IMS
Version 14

Commands, Volume 2: IMS Commands N - V

(March 1, 2018 edition)

IBM

IMS

Version 14

*Commands, Volume 2:
IMS Commands N - V*

(March 1, 2018 edition)

IBM

Contents

About this information

These topics provide command syntax and usage information for the IMS™ type-1 and type-2 commands /NRESTART through /VUNLOAD. The topics also describe the IMS command language and how to send commands to IMS in different environments. Information about all non-type-1 and non-type-2 IMS commands is in *IMS Version 14 Commands, Volume 3: IMS Component and z/OS Commands*.

This information is available in IBM® Knowledge Center.

Prerequisite knowledge

Before using this information, you should have knowledge of either IMS Database Manager (DB) or IMS Transaction Manager (TM). You should also understand basic z/OS® and IMS concepts, your installation's IMS system, and have general knowledge of the tasks involved in project planning.

Recommendation: Before using this information, you should be familiar with the following resources:

- *IMS Version 14 Operations and Automation*
- *z/OS JES2 Commands*
- *z/OS JES3 Commands*
- *z/OS MVS™ System Commands*

You can learn more about z/OS by visiting the "z/OS basic skills" topics in IBM Knowledge Center.

You can gain an understanding of basic IMS concepts by reading *An Introduction to IMS*, an IBM Press publication.

IBM offers a wide variety of classroom and self-study courses to help you learn IMS. For a complete list of courses available, go to the IBM Skills Gateway and search for IMS.

How new and changed information is identified

New and changed information in most IMS library PDF publications is denoted by a character (revision marker) in the left margin. The first edition (-00) of *Release Planning*, as well as the *Program Directory* and *Licensed Program Specifications*, do not include revision markers.

Revision markers follow these general conventions:

- Only technical changes are marked; style and grammatical changes are not marked.
- If part of an element, such as a paragraph, syntax diagram, list item, task step, or figure is changed, the entire element is marked with revision markers, even though only part of the element might have changed.
- If a topic is changed by more than 50%, the entire topic is marked with revision markers (so it might seem to be a new topic, even though it is not).

Revision markers do not necessarily indicate all the changes made to the information because deleted text and graphics cannot be marked with revision markers.

How to read syntax diagrams

The following rules apply to the syntax diagrams that are used in this information:

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following conventions are used:
  - The >>--- symbol indicates the beginning of a syntax diagram.
  - The ---> symbol indicates that the syntax diagram is continued on the next line.
  - The >--- symbol indicates that a syntax diagram is continued from the previous line.
  - The --->< symbol indicates the end of a syntax diagram.
- Required items appear on the horizontal line (the main path).

```
►►──required_item────────────────────────────────────────►◄
```
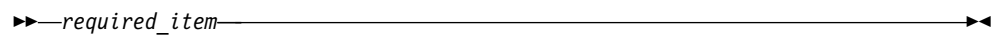
- Optional items appear below the main path.

```
►►──required_item──────────────────────────────────────────►◄
              └─optional_item─┘
```
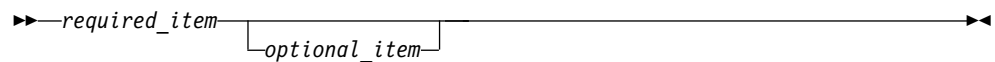
If an optional item appears above the main path, that item has no effect on the execution of the syntax element and is used only for readability.

```
              ┌─optional_item─┐
►►──required_item──────────────────────────────────────────►◄
```
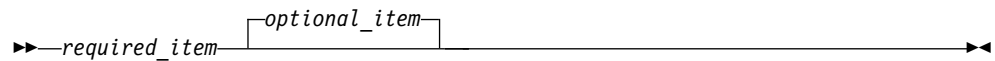
- If you can choose from two or more items, they appear vertically, in a stack.

  If you *must* choose one of the items, one item of the stack appears on the main path.

```
►►──required_item──┬─required_choice1─┬──────────────────────►◄
                   └─required_choice2─┘
```
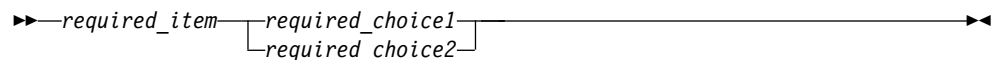
If choosing one of the items is optional, the entire stack appears below the main path.

```
►►──required_item──┬──────────────────┬──────────────────────►◄
                   ├─optional_choice1─┤
                   └─optional_choice2─┘
```
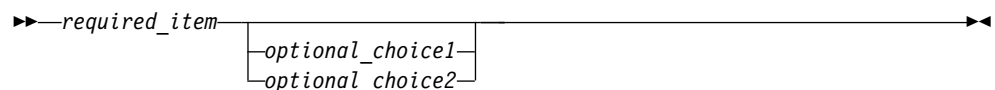
If one of the items is the default, it appears above the main path, and the remaining choices are shown below.

```
                   ┌─default_choice──┐
►►──required_item──┼─────────────────┼──────────────────────►◄
                   ├─optional_choice─┤
                   └─optional_choice─┘
```
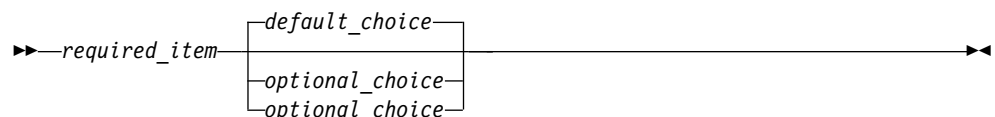
- An arrow returning to the left, above the main line, indicates an item that can be repeated.

```
►►─required_item──┬─repeatable_item─┬──────────────────────────►◄
                  └─────────────────┘
```
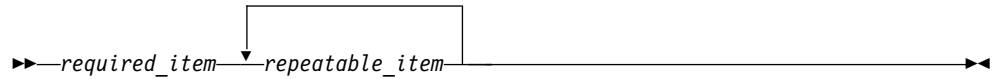
If the repeat arrow contains a comma, you must separate repeated items with a comma.

```
                  ┌─,───────────────┐
►►─required_item──┴─repeatable_item─┴──────────────────────────►◄
```

A repeat arrow above a stack indicates that you can repeat the items in the stack.

- Sometimes a diagram must be split into fragments. The syntax fragment is shown separately from the main syntax diagram, but the contents of the fragment should be read as if they are on the main path of the diagram.

```
►►─required_item─┤ fragment-name ├──────────────────────────────►◄
```

fragment-name:

```
├─required_item──┬────────────────┬──────────────────────────────┤
                 └─optional_item──┘
```
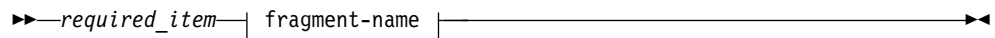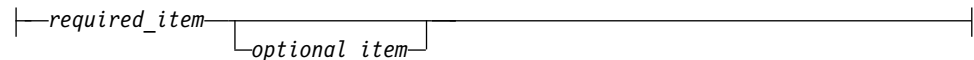
- In IMS, a b symbol indicates one blank position.
- Keywords, and their minimum abbreviations if applicable, appear in uppercase. They must be spelled exactly as shown. Variables appear in all lowercase italic letters (for example, *column-name*). They represent user-supplied names or values.
- Separate keywords and parameters by at least one space if no intervening punctuation is shown in the diagram.
- Enter punctuation marks, parentheses, arithmetic operators, and other symbols, exactly as shown in the diagram.
- Footnotes are shown by a number in parentheses, for example (1).

Accessibility features for IMS 14

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility features

The following list includes the major accessibility features in z/OS products, including IMS 14. These features support:

- Keyboard-only operation.
- Interfaces that are commonly used by screen readers and screen magnifiers.
- Customization of display attributes such as color, contrast, and font size.

Keyboard navigation

You can access IMS 14 ISPF panel functions by using a keyboard or keyboard shortcut keys.

For information about navigating the IMS 14 ISPF panels using TSO/E or ISPF, refer to the *z/OS TSO/E Primer*, the *z/OS TSO/E User's Guide*, and the *z/OS ISPF User's Guide Volume 1*. These guides describe how to navigate each interface, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Related accessibility information

Online documentation for IMS 14 is available in IBM Knowledge Center.

IBM and accessibility

See the *IBM Human Ability and Accessibility Center* at www.ibm.com/able for more information about the commitment that IBM has to accessibility.

How to send your comments

Your feedback is important in helping us provide the most accurate and highest quality information. If you have any comments about this or any other IMS information, you can take one of the following actions:

- Click the **Contact Us** tab at the bottom of any IBM Knowledge Center topic.
- Send an email to imspubs@us.ibm.com. Be sure to include the book title and the publication number.

To help us respond quickly and accurately, please include as much information as you can about the content you are commenting on, where we can find it, and what your suggestions for improvement might be.

Chapter 1. /NRESTART command

The /NRESTART command is used to cold start IMS or warm start IMS following an orderly termination accomplished with a /CHECKPOINT shutdown command.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 3
- "Usage notes" on page 6
- "Examples" on page 8

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 1. Valid environments for the /NRESTART command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/NRESTART	X	X	X
BUILDQ	X		X
CHKPT	X	X	X
CMDAUTH	X		X
CMDAUTHE	X		X
FORMAT	X	X	X
MSDBLOAD	X		
MULTSIGN	X		X
NOBUILDQ	X		X
NOCMDAUTH	X		X
NOCMDAUTHE	X		X
NOTRANAUTH	X		X
NOUSER	X		X
SNGLSIGN	X		X
TRANAUTH	X		X
USER	X		X

Syntax

Cold start with no previous shutdown
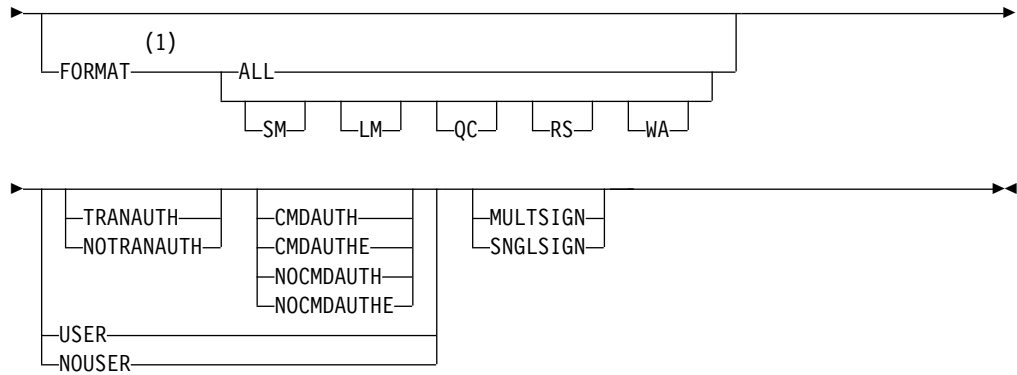
```
►►─┬─/NRESTART─┬──CHKPT──0──────────────────────────────────────────►
   └─/NRE──────┘
```

```
>>─┬────────────────────────────────────────────────────────┬─────────>
   │            (1)                                           │
   └─FORMAT─┬──ALL──────────────────────────────────────┬────┘
            └──┬────┬─┬────┬─┬────┬─┬────┬─┬────┬────────┘
               └─SM─┘ └─LM─┘ └─QC─┘ └─RS─┘ └─WA─┘

>─┬──────────────────┬─┬──────────────┬─┬───────────┬─────────────────><
  ├─TRANAUTH─────────┤ ├─CMDAUTH──────┤ ├─MULTSIGN──┤
  └─NOTRANAUTH───────┤ ├─CMDAUTHE─────┤ └─SNGLSIGN──┘
  │                  │ ├─NOCMDAUTH────┤
  ├─USER─────────────┤ └─NOCMDAUTHE───┤
  └─NOUSER───────────┘
```

Notes:

1 The FORMAT keyword must be followed by at least one of the SM, LM, QC, RS, WA, or ALL parameters.

Warm start after a /CHECKPOINT FREEZE command
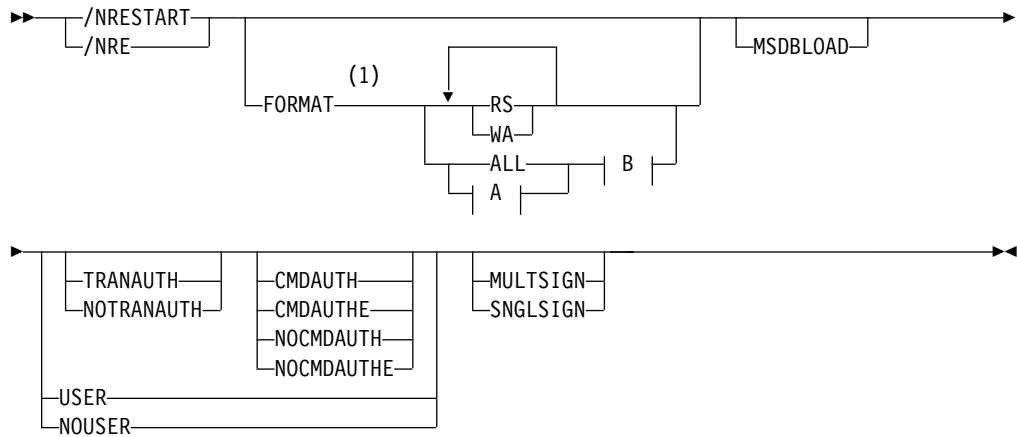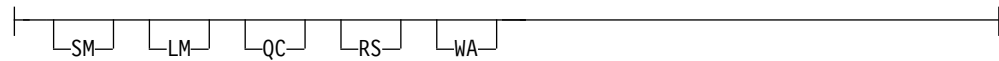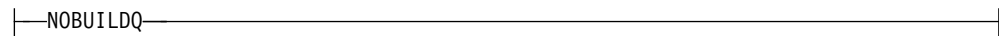
```
>>─┬─/NRESTART─┬─┬──────────────────────────────────┬─┬───────────┬──>
   └─/NRE──────┘ │           (1)  ┌─────────────┐   │ └─MSDBLOAD──┘
                 └─FORMAT──┬─┬─▼─┬─RS─┬──────┬───┴─┬─┘
                           │   └─┴─WA─┘      │     │
                           └──┬──ALL──┬─┤ B ├─┘
                              └──┤ A ├─┘

>─┬──────────────────┬─┬──────────────┬─┬───────────┬─────────────────><
  ├─TRANAUTH─────────┤ ├─CMDAUTH──────┤ ├─MULTSIGN──┤
  └─NOTRANAUTH───────┤ ├─CMDAUTHE─────┤ └─SNGLSIGN──┘
  │                  │ ├─NOCMDAUTH────┤
  ├─USER─────────────┤ └─NOCMDAUTHE───┤
  └─NOUSER───────────┘
```

A:

```
├─┬────┬─┬────┬─┬────┬─┬────┬─┬────┬─────────────────────────────────────┤
  └─SM─┘ └─LM─┘ └─QC─┘ └─RS─┘ └─WA─┘
```

B:

```
├─NOBUILDQ──────────────────────────────────────────────────────────────┤
```

Notes:

1 The FORMAT keyword must be followed by at least one of the SM, LM, QC, RS, WA, or ALL parameters.

Warm start after a /CHECKPOINT PURGE or /CHECKPOINT DUMPQ command
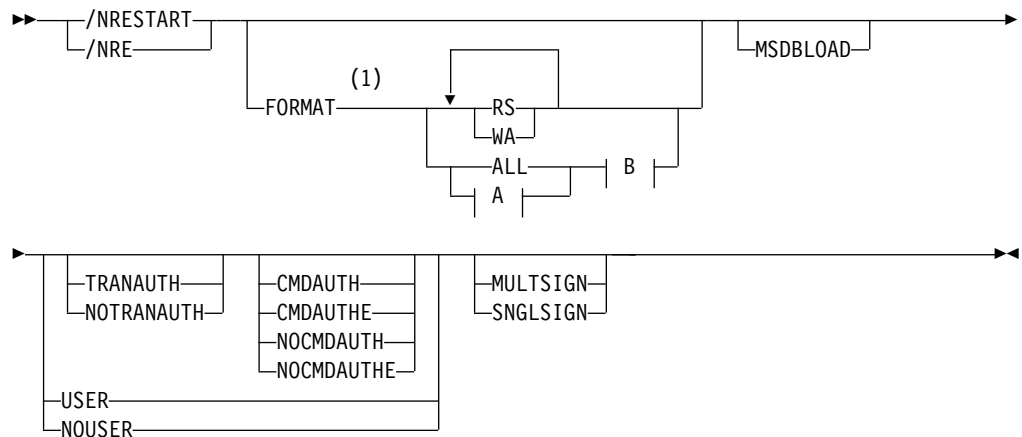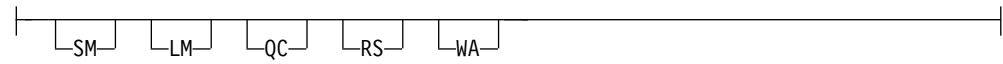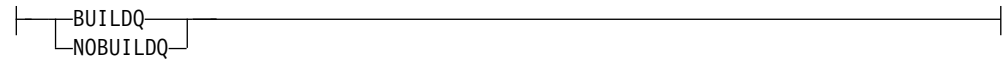
```
►►─┬─/NRESTART─┬──┬──────────────────────────────────────────┬──┬──────────┬──►
   └─/NRE──────┘  │                  (1)    ┌────────┐        │  └─MSDBLOAD─┘
                  └─FORMAT──────────┬─►──┬─RS─┬──────┴┬──────┬─┘
                                    │    └─WA─┘       │      │
                                    ├─ALL──────┬─ B ──┤
                                    └─A─────────┘      │

►──┬──────────────┬──┬────────────┬──┬───────────┬──────────────────────────►◄
   ├─TRANAUTH─────┤  ├─CMDAUTH────┤  ├─MULTSIGN──┤
   └─NOTRANAUTH───┤  ├─CMDAUTHE───┤  └─SNGLSIGN──┘
                  │  ├─NOCMDAUTH──┤
   ┌─USER────────┐│  └─NOCMDAUTHE─┘
   └─NOUSER──────┘
```

A:

```
├──┬────┬──┬────┬──┬────┬──┬────┬──┬────┬──┤
   └─SM─┘  └─LM─┘  └─QC─┘  └─RS─┘  └─WA─┘
```

B:

```
├──┬─BUILDQ───┬──────────────────────────────┤
   └─NOBUILDQ─┘
```

Notes:

1   The FORMAT keyword must be followed by at least one of the SM, LM, QC, RS, WA, or ALL parameters.

Keywords

The following keywords are valid for the /NRESTART command:

BUILDQ | NOBUILDQ

    BUILDQ requests that the message queues dumped on the log be loaded into the message queue data sets. BUILDQ is optional for a warm start after a /CHECKPOINT PURGE or /CHECKPOINT DUMPQ.

    The BUILDQ keyword must be included if the message queues are to be restored after being formatted. If the BUILDQ keyword is specified, the log from the last /CHECKPOINT DUMPQ or /CHECKPOINT PURGE is required, slowing down the restart process.

    If /NRESTART FORMAT is specified without the BUILDQ keyword, the NOBUILDQ keyword must be specified. Specifying NOBUILDQ reformats the queues in question and all messages are lost.

    If an /NRESTART BUILDQ command fails, and then the /ERESTART CHECKPOINT 0, /ERESTART COLDCOMM, or /ERESTART COLDSYS command is performed, the messages are lost. IBM IMS Queue Control Facility for z/OS can be used to recover the local message queues.

    In a shared-queues environment, the BUILDQ keyword is ignored because the message queue data sets are not used.

CHECKPOINT

Identifies the shutdown/restart sequence. `CHECKPOINT 0` must be specified for a cold start.

CMDAUTH

Specifies that both signon (user identification verification) and command authorization for static and ETO terminals are in effect at the end of the emergency restart. (Command authorization is same as specifying RCF=S on the startup parameter.)

To specify CMDAUTH, you must specify `RCF=A|Y|T|C|S` as an initialization EXEC parameter.

CMDAUTHE

Specifies that command authorization for ETO terminals (same as RCF=S on the startup parameter) is in effect at the end of the emergency restart. CMDAUTHE also resets command authorization for static terminals, if it was set.

To specify CMDAUTHE, you must specify `RCF=A|Y|T|C|S` as an initialization EXEC parameter.

FORMAT

Specifies which queues or data sets should be formatted as part of the restart process when:

- A message queue or data set I/O error occurs.
- The size of a message queue or data set is to be changed.
- A message queue or data set is to be reallocated.

Specify one or more of the following or ALL:

SM   Short-message queue

LM   Long-message queue

QC   Control record data set

RS   Restart data set

WA   Write-ahead data set

ALL   All message queues (SM and LM) and data sets (QC, WA, and RS)

When FORMAT ALL is specified, do not also specify SM, LM, QC, WA, or RS. FORMAT ALL is only required at IMS initialization (first time use of the system).

You can specify any combination of SM, LM, QC, WA, and RS; for example, FORMAT LM RS.

In a shared-queues environment, the LM, SM, and QC parameters are ignored because the message queue data sets are not used. If you specify ALL, IMS does not attempt to format the message queue data sets.

The following table shows the environments in which the parameters are valid.

Table 2. /NRESTART FORMAT command parameter environments

Parameter	DB/DC	DBCTL	DCCTL
SM	X		X
LM	X		X
QC	X		X
RS	X	X	X

Table 2. /NRESTART FORMAT command parameter environments (continued)

Parameter	DB/DC	DBCTL	DCCTL
WA	X	X	X
ALL	X	X [1]	X

Note:

1. Supports only RS and WA parameters.

MSDBLOAD
> Requests that the MSDBs be loaded from the z/OS sequential data set MSDBINIT instead of the MSDB checkpoint data set. Use the MSDBLOAD keyword only when an MSDB initial load is required; otherwise, omit it from the /NRESTART command. After you modify an MSDB DBD, you must specify MSDBLOAD on the next warm start of IMS in order for the changes to be effective.
>
> MSDBLOAD is not required:
> *   For warm starts when the MSDB checkpoint data set is used.
> *   For a cold start because the MSDBs are loaded from the z/OS sequential data set MSDBINIT and the MSDB checkpoint data sets are formatted.

MULTSIGN
> Permits multiple signons for each user ID.
>
> In an IMSplex with Resource Manager and a resource structure, if MULTSIGN conflicts with the single user signon definition for the IMSplex, a warning message will be issued.

NOCMDAUTH
> Resets command authorization on static and ETO terminals.

NOCMDAUTHE
> Resets command authorization for static and ETO terminals. The command authorization is reset for static terminals because the command authorization for static terminals cannot exist without the command authorization for ETO terminals.

NOTRANAUTH
> Turns off transaction authorization. NOTRANAUTH is not the opposite of TRANAUTH. TRANAUTH sets transaction authorization and also turns on signon (user identification verification).
>
> If you specify NOTRANAUTH, it is rejected with an error message if TRN=F was specified as a JCL EXEC parameter.

NOUSER
> Specifies that none of the following is in effect at the end of the emergency restart:
> *   transaction authorization
> *   user identification verification
> *   command authorization
>
> You can use /NRESTART NOUSER for a warm start, but if transaction or command authorization is set from the checkpoint data, NOUSER is ignored, and signon (user identification verification) is set on.

SNGLSIGN
> Permits a single signon for each user ID.

In an IMSplex with Resource Manager and a resource structure, if SNGLSIGN conflicts with the single user signon definition for the IMSplex, a warning message will be issued.

TRANAUTH

Specifies both transaction authorization and user identification verification, with or without RACF®.

To specify TRANAUTH, you must specify `RCF=A|Y|T|C|S` as an initialization EXEC parameter.

USER

Specifies user identification verification. User identification verification means that signon is required by static terminals. This keyword has no effect on ETO terminals, because they are always required to sign on. User identification verification can be forced on by the TRANAUTH or CMDAUTH keyword

To specify USER, you must specify `RCF=A|Y|T|C|S` as an initialization EXEC parameter.

Usage notes

This command can be issued to an IMSplex using the Batch SPOC utility.

The /NRESTART command has three forms. The selected form depends on the following conditions:

- Whether a cold start or warm start is required
- In the case of warm starts, whether the previous orderly shutdown was accomplished with:
  - /CHECKPOINT FREEZE
  - /CHECKPOINT PURGE or DUMPQ

Attention: A cold start performed after a processing failure could cause processing against uncommitted data. To ensure data integrity, be sure necessary backout or recovery operations have been performed before restarting.

When IMS initializes, the system parameters used for this initialization can come from the IMS system generation, from a PROCLIB member, or from EXEC statements that can override both the defaults and the PROCLIB members. Therefore, message `DFS1929I` is displayed showing the system parameters used for this particular initialization. The system parameters are also written to the job log.

For an IMS cold start, the base security definition is created from the IMS system definition and EXEC parameter specifications. For an /NRESTART warm restart, the base security definition is created from the IMS checkpoint data.

To override the base security definitions on a cold start, the security keywords of the /NRESTART command must be used.

The SGN=, TRN=, and RCF= startup parameters can be overridden by the /NRESTART command using the security keywords shown in the following table. A brief description of the keywords is also included.

Table 3. Security keywords and their startup parameter equivalents

Keyword	Description	Startup parameter
CMDAUTH	RACF command authorization on static and ETO terminals only.	RCF=S
CMDAUTHE	RACF command authorization on ETO terminals only.	RCF=C
MULTSIGN	Permits multiple signons for each user ID.	SGN=M
NOCMDAUTH	Resets the command authorization on static and ETO terminals.	Not RCF=S
NOCMDAUTHE	Resets the command authorization on ETO terminals only.	Not RCF=C
NOTRANAUTH	Resets the transaction authorization.	Not TRN=F or Y
NOUSER	Resets user identification verification, transaction authorization, and command authorization.	Not SGN=F or Y (G or Z becomes M) Not TRN=F or Y Not RCF=C or S
SNGLSIGN	Permits a single signon for each user ID.	SGN=F and Y Not SGN=M (G or Z becomes F or Y)
TRANAUTH	Transaction authorization.	TRN=F or Y
USER	Sets user identification verification.	SGN=Y

If IMS is enabled to use the IMSRSC repository, IMS calls Resource Manager (RM) during warm start to read the change list, if one exists, for the IMS. The IMS change list is maintained by RM and is built if the IMPORT command with the SCOPE(ALL) keyword is issued while the IMS is down.

If a change list exists for the IMS that is being restarted, the database, program, transaction, and routing code resources and descriptors in the IMS change list and that apply to the IMS environment are quiesced and are not available for use until the stored resource definitions are imported from the repository.

After the IMS log is processed, IMS imports the stored resource definitions from the repository for the database, program, transaction, and routing code resources and descriptors in the IMS change list. It then applies the changes to the runtime resource and descriptor definitions in the IMS.

For the resources or descriptors that are in the IMS change list and that do not exist in IMS, the runtime resource definitions are created from the stored resource definitions in the repository. For the resource or descriptors that exist in IMS, the runtime resource definitions are updated with the stored resource definitions from the repository. The change list for the IMS system is deleted at the end of the warm start.

During cold start, if IMS is enabled to use the repository, the stored resource definitions are read from the repository. The change list is not processed during cold start because all the resource definitions are imported as part of cold start processing. Any change list in RM for the IMS being restarted is deleted.

Examples

The following are examples of the /NRESTART command:

Example 1 for /NRESTART command

This is an example of a cold start with new message queue data sets.

Entry ET:

```
/NRESTART CHECKPOINT 0 FORMAT ALL
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS994I *CHKPT 82274/114447**SIMPLE*
```

Explanation: IMS is started at 114447 (time) on 82274 (Julian date). A simple checkpoint is written on the system log. All message queue data sets are formatted. 82274/114447 is the checkpoint number.

Example 2 for /NRESTART command

This is an example of a warm start from a FREEZE checkpoint.

Entry ET:

```
/NRESTART
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82273/180000
DFS994I *CHKPT 82274/082217**SIMPLE*
```

Explanation: The restart is being performed from checkpoint 82273/180000, which was written at the most recent IMS shutdown. IMS is restarted at 082217 (time) on 82274 (Julian date). A simple checkpoint is written on this system log. 82274/082217 is the checkpoint number.

Example 3 for /NRESTART command

This is an example of a warm start to format WADS.

Entry ET:

```
/NRESTART FORMAT WA
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82119/230000
DFS994I *CHKPT 82120/101318**SIMPLE*
```

Explanation: The restart is being performed from checkpoint 82119/230000, which was written at the most recent IMS shutdown. IMS is restarted at 101318 (time) on 82120 (Julian date). A simple checkpoint is written on the system log. 82120/101318 is the checkpoint number.

Example 4 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint.

Entry ET:

```
/NRESTART BUILDQ
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82080/214240
DFS994I *CHKPT 82081/060000**SIMPLE*
```

Explanation: IMS is restarted at 060000 (time) on 82081 (Julian date) from checkpoint 82080/214240, which was written at the most recent IMS shutdown. 82081/060000 is the checkpoint number.

Example 5 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint. The large and small message queue data sets have been reallocated.

Entry ET:

```
/NRESTART BUILDQ FORMAT SM LM
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82170/085236
DFS994I *CHKPT 82170/085820**SIMPLE*
```

Explanation: IMS is restarted at 085820 (time) on 82170 (Julian date) from checkpoint 82170/085236, which was written at the most recent IMS shutdown. The large and small message queue data sets are reformatted. 82170/085820 is the checkpoint number.

Example 6 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint. An initial set of MSDBs is needed.

Entry ET:

```
/NRESTART BUILDQ MSDBLOAD
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82068/180000
DFS2554 MSDB MSDBHJ01 LOADED
DFS2554 MSDB MSDBHJ02 LOADED
DFS2554 MSDB MSDBAK01 LOADED
DFS2554 MSDB MSDBAK02 LOADED
DFS2554 MSDB MSDBPS01 LOADED
DFS994I *CHKPT 82069/080000**SIMPLE*
```

Explanation: IMS is restarted at 080000 (time) on 82069 (Julian date) from checkpoint 82068/180000, which was written at the most recent IMS shutdown. A simple checkpoint is written on the system log. 82069/080000 is the checkpoint number. An initial set of MSDBs is loaded from the z/OS sequential data set MSDBINIT.

Example 7 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint with a request for transaction command security.

Entry ET:

```
/NRESTART TRANCMDS
```

Response ET:

```
DFS058I  (time stamp) NRESTART COMMAND IN PROGRESS
DFS680I USING CHKPT 82080/214240
DFS994I *CHKPT 82274/114447**SIMPLE*
```

Explanation: The master terminal operator is warm starting IMS and requesting that transaction command security be in effect at the completion of the normal restart.

Example 8 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint with a request to negate transaction command security.

Entry ET:

```
/NRESTART NOTRANCMDS
```

Response ET:

```
DFS2181I  CANNOT OVERRIDE FORCED COMMAND SECURITY
```

Explanation: The master terminal operator is warm starting IMS with a request to negate transaction command security. IMS system definition precludes the authority of the master terminal operator to enter the NOTRANCMDS keyword. IMS returns the DFS2181 message.

Example 9 for /NRESTART command

This is an example of a warm start from a PURGE or DUMPQ checkpoint. The data sets are formatted without rebuilding the message queues.

Entry ET:

```
/NRESTART FORMAT SM LM NOBUILDQ
```

Response ET:

```
DFS058 NRESTART COMMAND IN PROGRESS
```

Explanation: The master terminal operator is warm starting IMS with a request to reformat the data sets without rebuilding the message queues. IMS comes up, but all messages are lost.

Related reference:

➡ IMS Queue Control Facility overview

Chapter 2. /OPNDST command

The /OPNDST command is a multisegment command that causes IMS to initiate a session with either an ISC TCP/IP or a VTAM® terminal.

For VTAM terminals, if the USER keyword is specified, the user is signed on automatically to the terminal after successful session initiation.

For ISC TCP/IP terminals, the user is signed on automatically after successful session initiation.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 12
- "Usage notes" on page 15
- "Examples" on page 16

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 4. Valid environments for the /OPNDST command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/OPNDST	X		X
ID	X		X
LOGOND	X		X
MODE	X		X
NODE	X		X
Q	X		X
UDATA	X		X
USER	X		X
USERD	X		X

Syntax

The /OPNDST command without the USER keyword: Use this form of the command for all static and ETO terminals except ETO SLU P and Finance terminals, ETO output-only devices, and all ISC parallel sessions.

```
                    ┌─────────────────────────────────────────────────┐
►►──┬───────────────────────────┬──────────────────────────────────►◄
    └─LOGOND──logondname─┘
```
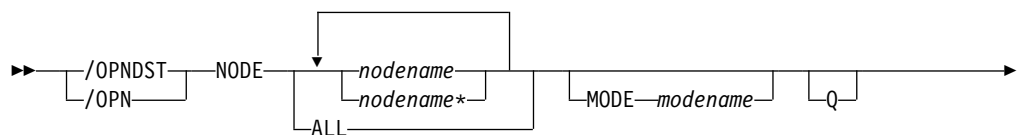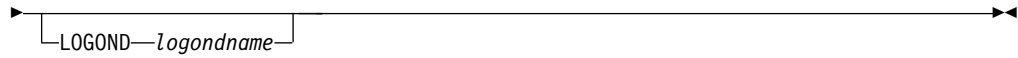
The /OPNDST command with the USER keyword for non-ISC: Use this form of
the command for:

- ETO SLU P and Finance terminals
- ETO output-only devices, for example, 3284, 3286, and SLU P1 with a single
  component of PRINTER1

```
►►──┬─/OPNDST─┬──NODE──nodename──USER──username──┬───────────────────┬──┬───┬──►
    └─/OPN────┘                                  └─MODE──modename─┘  └─Q─┘

►──┬───────────────────────┬──┬──────────────────────┬──┬────────────────────┬──►◄
   └─LOGOND──logondname─┘     └─USERD──userdname─┘      └─UDATA──userdata─┘
```
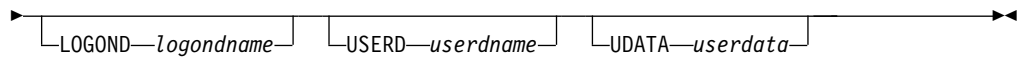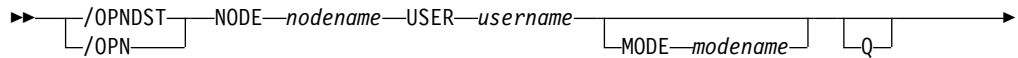
The /OPNDST command with the USER keyword for ISC TCP/IP: Use this form
of the command for all ISC parallel sessions, both static and ETO.

```
►►──┬─/OPNDST─┬──NODE──nodename──USER──username──┬──────────────┬──────────────►
    └─/OPN────┘                                  └─ID──idname─┘

►──┬───────────────────────┬──────────────────────────────────────────────────►◄
   └─LOGOND──logondname─┘
```
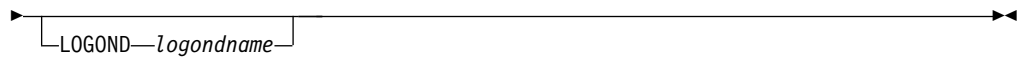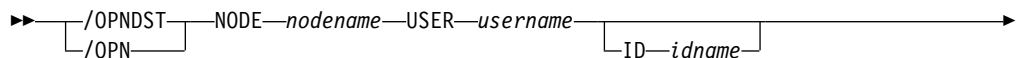
The /OPNDST command with the USER keyword for ISC VTAM: Use this form
of the command for all ISC parallel sessions, both static and ETO.

```
►►──┬─/OPNDST─┬──NODE──nodename──USER──username──┬──────────────┬──────────────►
    └─/OPN────┘                                  └─ID──idname─┘

►──┬───────────────────┬──┬───┬──────────────────────────────────────────────►◄
   └─MODE──modename─┘     └─Q─┘
```
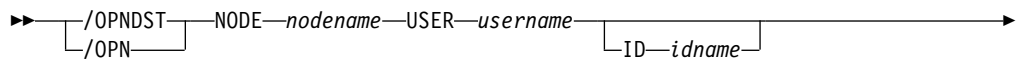
Keywords

The following keywords are valid for the /OPNDST command:

ID   Is applicable only if the USER keyword is specified and the node is a parallel
session ISC VTAM node. ID must not be specified for a single session ISC
node.

For ISC VTAM parallel sessions, ID identifies the other system half-session
qualifier. ID *idname* must be specified to open ISC nodes defined with users.
The ID *idname* is passed to the other half-session with the session initiation
request. If the other system is another IMS system, *idname* is the name of an
ISC user in that system.

For ISC TCP/IP parallel sessions, the ID keyword does not apply. If the ID
keyword is specified, it is ignored.

LOGOND

Indicates the logon descriptor used for session establishment. LOGOND is supported for both dynamic ISC TCP/IP nodes and dynamic non-ISC nodes.

If a logon descriptor name is provided by the Logon exit routine (DFSLGNX0), the name provided by the exit routine overrides the name specified on the LOGOND keyword.
For ISC TCP/IP nodes, the name specified on the LOGOND keyword must match a logon descriptor that specifies the LCLICON keyword for TCP/IP support. If the logon descriptor does not specify TCP/IP support, IMS attempts to open the session as a VTAM node.

MODE

Identifies the LOGON MODE table entry that VTAM must use and can determine operating characteristics for certain VTAM terminals. If a list of node names is given, the MODE keyword is applied to each of them. The command checks whether each node has been defined to accept IMS initiated connections. If the node was defined as NOPNDST, the /OPNDST command accepts all the nodes except the node defined as NOPNDST.

ISC TCP/IP sessions do not support the MODE keyword.

NODE

Specifies the node with which IMS initiates a session.

The /OPNDST NODE ALL command opens sessions for all static terminals except ISC parallel sessions. The command has considerable concurrent activity, both for IMS and for VTAM. Ensure that the system has sufficient pool sizes, buffer sizes, and number of concurrent IMS tasks defined.

If the USER keyword is not specified in the command, the NODE parameter can be generic or ALL, or a range of static nodes, or there can be multiple NODE parameters. If a generic, ALL, or range of node names is specified, any nodes defined with users are ignored and flagged with an error message.

The /OPNDST NODE USER command logs on and signs on a user at the same time, except for ISC terminals. On ISC terminals, the session is allocated.

The following list includes sample /OPNDST NODE USER commands:

- To log on and sign on to a static non-ISC terminal:

    ```
    /OPNDST NODE nodename USER username UDATA userdata
    ```

    This command marks a statically defined terminal as signed on by the user.

- To log on and sign on to an ETO non-ISC terminal:

    ```
    /OPNDST NODE nodename USER username MODE modename
    LOGOND logondname USERD userdname UDATA userdata
    ```

    This command creates the terminal and user structures, and allocates the newly created user structure to the terminal structure created to indicate signed on status.

- To log on and sign on to a static ISC terminal:

    ```
    /OPNDST NODE nodename USER username ID idname
    ```

    This command finds the subpool structure and allocates the subpool (user) structure to the statically defined ISC terminal. If the terminal is defined to use ISC TCP/IP, the ID keyword does not apply and is ignored by IMS.

- To log on and sign on to an ETO ISC VTAM terminal:

    ```
    /OPNDST NODE nodename USER username ID idname MODE modename
    LOGOND logondname USERD userdname UDATA userdata
    ```

This command creates the terminal and subpool (user) structure and allocates the newly created user structure to the terminal structure created. The command also signs on the user with the username (the username must be defined to RACF), and remains signed on until the user issues a /SIGN OFF command, or the session is terminated.

- To log on and sign on to an ETO ISC TCP/IP terminal:

  /OPNDST NODE *nodename* USER *username* LOGOND *logondname*

  This command creates the terminal and subpool (user) structure and allocates the newly created user structure to the terminal structure created. The command also signs on the user with the username (the username must be defined to RACF), and remains signed on until the user issues a /SIGN OFF command, or the session is terminated.

  For ISC TCP/IP parallel sessions, a logon descriptor must exist that specifies the LCLICON keyword for TCP/IP support. IMS uses the value of the NODE keyword to search for a logon descriptor only when the LOGOND keyword is not specified. If the logon descriptor does not exist or does not specify TCP/IP support, IMS attempts to open the session as a VTAM node.

To restart failing ETO sessions (for example, ETO only or printer sessions), use the /OPNDST NODE USER command.

In an IMSplex, /OPNDST NODE specifies a VTAM node with which IMS initiates a session. Specify ROUTE(*imsid*), if you want to log the node onto a particular IMS. If ROUTE(*imsid*) is not specified, and /OPNDST is routed to all the IMS systems, IMS processes the command only on the IMS system designated as the command master. If the ROUTE keyword specifies multiple IMS systems so that the /OPNDST command is routed to more than one IMS systems, IMS processes the command only on the IMS system designated as the command master. On the other IMS systems, the /OPNDST command is rejected.

Q   Causes IMS to request VTAM to queue SIMLOGON requests for VTAM/SNA-supported terminals.

The /OPNDST NODE Q command also allows IMS to request another subsystem to share a node (usually printers) with IMS. If the other subsystem is using the printer, VTAM queues the SIMLOGON request for IMS for the printer, schedules the owning subsystem's RELREQ VTAM exit, and acquires the printer for IMS after the current owning system releases the printer. Multiple requests for the same printer are queued by VTAM for processing.

The /DISPLAY NODE command is used to determine whether IMS has acquired the printer.

ISC TCP/IP sessions do not support the Q keyword.

UDATA
Indicates the user data used with the signon.

The UDATA keyword is valid only if the USER keyword and parameter are also specified. The UDATA keyword is valid for static and dynamic users. It is not valid for ISC nodes. The user data can be up to 256 bytes long. Passwords can be mixed case or lowercase depending on what is specified on the PSWDC keyword in the DFSPBxxx IMS.PROCLIB member.

ISC TCP/IP sessions do not support the UDATA keyword.

RACF password phrases are not supported with UDATA.

USER

Identifies the logical terminal user to be allocated to the half-session to be created for the ISC node that is specified on the NODE keyword.

For dynamic non-ISC users, it specifies the user ID to be signed on to the dynamic node *nodename*.

The USER keyword applies to ISC sessions when allocating a user to an ISC node, to dynamic users when signing a dynamic user on to a dynamic node, and to static user IDs when signing a user on to a static node.

USER username must be specified to open parallel session ISC nodes with users. It must not be specified for a single session ISC node.

Restrictions for using NODE and USER parameters together:
- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- The /OPNDST NODE USER commands are valid for ISC and non-ISC nodes and users.

USERD

Specifies the user descriptor to be used with the signon. It is valid only if the USER keyword and parameter are specified. USERD is only supported for dynamic users and is only valid for dynamic non-ISC nodes. The user descriptor can also be provided through the logon or signon exits.

ISC TCP/IP sessions do not support the USERD keyword.

Usage notes

All forms of logging-on a remote VTAM terminal to IMS, including the use of the /OPNDST command, do not work until the /START DC command has been entered and accepted by IMS.

All /OPNDST formats require an EOM indication to denote end-of-message. An EOS indication must be included for all segments that precede the last segment.

You can issue /OPNDST on the XRF alternate to restart a failed backup session for a class 1 ETO terminal. To do this, the node and the user structure must still exist and be coupled together, and an active session must exist on the active system. If the node and the user are not coupled, or an active session does not exist on the active system, the command is rejected.

This command can be issued to an IMSplex using the Batch SPOC utility.

A period is normally used as the delimiter at the end of IMS commands. When support for special characters is enabled in RACF, a period becomes a valid character in the RACF password. Therefore, when a password is specified at the end of the /OPNDST command, you must insert a space before the period that you are using as the end-of-command delimiter. If a space is not added before the period that is intended to be the end-of-command delimiter and support for special characters is enabled in RACF, the period is treated as part of the password and not as a delimiter.

One or more equal signs (=) can normally be used as a valid delimiter between a keyword and the keyword value on the /OPNDST command. However, when

support for special characters is enabled in RACF, an equal sign becomes a valid character in the RACF password. If you use the equal sign as a delimiter when support for special characters is enabled in RACF, you can specify only one delimiter immediately preceding the password on the **UDATA** keyword. Any equal sign that follows the delimiter is interpreted as the first character of the password.

For IMS 14 to support special characters in RACF passwords when special characters is also enabled in RACF, install APAR/PTFs PI48111/UI34376/UI34377. Also, ensure that all IMS systems that require support for RACF special characters are migrated to IMS 14.

Examples

The following are examples of the /OPNDST command:

Example 1 for /OPNDST command

Entry ET:

```
/DIS NODE L3270*
```

Response ET:

```
NODE-USR TYPE  CID       RECD ENQCT DEQCT  QCT  SENT
L3270A   3277  00000000    0    0     0     0     0 IDLE C1INOP STATIC
L3270B   3277  00000000    0    0     0     0     0 IDLE C1INOP STATIC
L3270C   3277  08000002   44   45    45     0    80 CON STATIC
L3270D   3277  00000000    2    0     0     0     7 IDLE STATIC
*94307/145048*
```

Entry ET:

```
/OPNDST NODE L3270*
```

Response ET:

```
DFS058I  OPNDST COMMAND COMPLETED
```

Entry ET:

```
/DIS NODE L3270*
```

Response ET:

```
NODE-USR TYPE  CID       RECD ENQCT DEQCT  QCT  SENT
L3270A   3277  06000004    0    0     0     0     1 IDLE CON STATIC
L3270B   3277  04000005    0    0     0     0     1 IDLE CON STATIC
L3270C   3277  08000002   46   46    46     0    82 CON STATIC
L3270D   3277  04000006    2    0     0     0     8 IDLE CON STATIC
*94307/145750*
```

```
Explanation:  The nodes L3270A through L3270D are logged on to IMS.
```

Example 2 for /OPNDST command

Entry ET:

```
/OPNDST NODE DT327002 USER IMSUS01 MODE LU032NT4 USERD DFSUSER
UDATA= IMSPW01.
```

Response ET:

```
DFS058I 11:07:48 OPNDST COMMAND COMPLETED
```

Explanation: A session with dynamic node DT327002 is established by using mode table LU032NT4. User IMSUS01 is signed on to the node using user descriptor DFSUSER, username IMSUS01, and password IMSPW01.

Example 3 for /OPNDST command

Entry ET:
```
/OPNDST NODE DTSLU201 USER IMSUS01 MODE SLU2MOD1 USERD DFSUSER
LOGOND DFSSLU2 UDATA=IMSPW01.
```

Response ET:
```
DFS058I 11:07:48 OPNDST COMMAND COMPLETED
```

Explanation: DTSLU201 is logged on. A session with dynamic node DTSLU201 is established by using logon descriptor DFSSLU2 (type SLU2), and mode table SLU2MOD1. Dynamic user IMSUS01 is signed on to the node using user descriptor DFSUSER and password IMSPW01.

Example 4 for /OPNDST command

Entry ET:
```
/OPNDST NODE WEST-EAST
```

Response ET:
```
DFS058I  OPNDST COMMAND COMPLETED
```

Response RT:
```
DFS3650 TERMINAL CONNECTED TO IMS XXXXXXXX
```

Explanation: The nodes, WEST through EAST, are logged on to IMS.

Example 5 for /OPNDST command

The following set of commands illustrate the use of the MODE keyword on the /OPNDST command.

Entry ET:
```
/DIS NODE LUTYPEP1 MODE
```

Response ET:
```
 NODE-USR TYPE   DEF MODETBL  ACT MODETBL
  LUTYPEP1 SLUP    DEFRESP
  *90179/100206*
```

Explanation: DEFRESP is the mode table name defined for node LUTYPEP1 at system definition. The session is not active so the ACT MODETBL field is blank.

Entry ET:
```
/OPN NODE LUTYPEP1.
```

Response ET:
```
DFS058I OPNDST COMMAND COMPLETED
```

Entry ET:
```
/DIS NODE LUTYPEP1 MODE
```

Response ET:
```
NODE-USR TYPE   DEF MODETBL   ACT MODETBL
 LUTYPEP1 SLUP   DEFRESP        DEFRESP
 *90179/100508*
```

Explanation: A mode table name was not specified with the /OPNDST command so the default value defined at system definition was used to initiate the session.

Entry ET:
```
/CLS NODE LUTYPEP1
```

Response ET:
```
DFS058I CLSDST COMMAND COMPLETED
```

Entry ET:
```
/DIS NODE LUTYPEP1 MODE
```

Response ET:
```
NODE-USR TYPE   DEF MODETBL   ACT MODETBL
 LUTYPEP1 SLUP    DEFRESP
 *90179/100630*
```

Explanation: Active mode table name displays as blank at normal session termination.

Entry ET:
```
/OPN NODE LUTYPEP1 MODE ALPHA.
```

Response ET:
```
DFS058I OPNDST COMMAND COMPLETED
```

Entry ET:
```
/DIS NODE LUTYPEP1 MODE
```

Response ET:
```
NODE-USR TYPE   DEF MODETBL   ACT MODETBL
 LUTYPEP1 SLUP   DEFRESP        ALPHA
 *90179/100805*
```

Explanation: The mode table name specified with the /OPNDST command (ALPHA) is used to initiate the session. The default value specified at system definition (DEFRESP) is overridden by the /OPNDST command.

Example 6 for /OPNDST command

In the following example, the /OPNDST NODE command is issued for an ISC TCP/IP parallel session.

Entry ET:
```
/OPN NODE CICS USER CICD.
```

Response ET:
```
DFS058I OPNDST COMMAND COMPLETED
```

Entry ET:

```
/DISPLAY NODE CICS USER CICD
```

Response ET:

```
NODE-USR TYPE  CID      RECD ENQCT DEQCT  QCT  SENT
CICS     TCP
-CICD         ********    0    0     0     0    0 SIGN(CICD   )
                                                 IDLE CON
*13175/200817*
```

Chapter 3. /PSTOP command

Use the /PSTOP command to stop the sending and receiving of messages to a particular communication line, terminal, or logical link.

You can also use this command to stop scheduling messages that contain specific transaction codes, to allow the queuing of output messages and input messages to continue, to halt a stalled synchronous program switch request, and to perform validity checks of all parameters entered by the terminal operator.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 22
- "Usage notes" on page 25
- "Equivalent IMS type-2 commands" on page 26
- "Examples" on page 26

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 5. Valid environments for the /PSTOP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/PSTOP	X	X	X
AOITOKEN	X	X	X
CLASS	X		X
FORCE	X		X
JOBNAME	X	X	X
LINE	X		X
LINK	X		X
LTERM	X		X
MSPLINK	X		X
PTERM	X		X
PURGE	X		X
REGION	X	X	X
SYNC	X		X
TRAN	X		X

Syntax

```
►►─┬─/PSTOP─┬─┬─LINE──┬──────────◄─line#◄─────────────────────────┬─────────────────────►◄
   └─/PST───┘ │       │                                           │
             │        ├─line#──PTERM──┬──◄─pterm#─┬───────────────┤
             │        │               └─ALL───────┘               │
             │        └─ALL───────────────────────────────────────┘
             │
             ├─LINK──┬─link #─────────┬──────────────────────────────
             │       ├─ALL────────────┤
             │       ├─link #─PURGE────┤
             │       └─link #─FORCE────┘
             │
             ├─LTERM──┬──◄─┬─ltermname──┬──────────────────────────
             │        │    └─ltermname*─┘
             │        └─ALL─────────────────
             │
             ├─MSPLINK──┬──◄─msplinkname─┬──────────────────────────
             │          └─ALL────────────┘
             │
             ├─REGION──┬─reg#──┬─AOITOKEN─aoitokenname──────────────
             │         │       ├─SYNC──┬─tpipename─┬────────────────
             │         │       │       └─tranname──┘
             │         │       └─TRAN─tranname──────────────────────
             │         └─JOBNAME─jobname──┬─AOITOKEN─aoitokenname────
             │                            ├─SYNC──┬─tpipename─┬──────
             │                            │       └─tranname──┘
             │                            └─TRAN─tranname────────────
             │
             └─TRAN──┬──◄─┬─tranname──┬─────────────────────────────
                     │    └─tranname*─┘
                     └─ALL──────────────────────────────
                          └─CLASS──┬──◄─cls#─┬───
                                   └─ALL─────┘
```
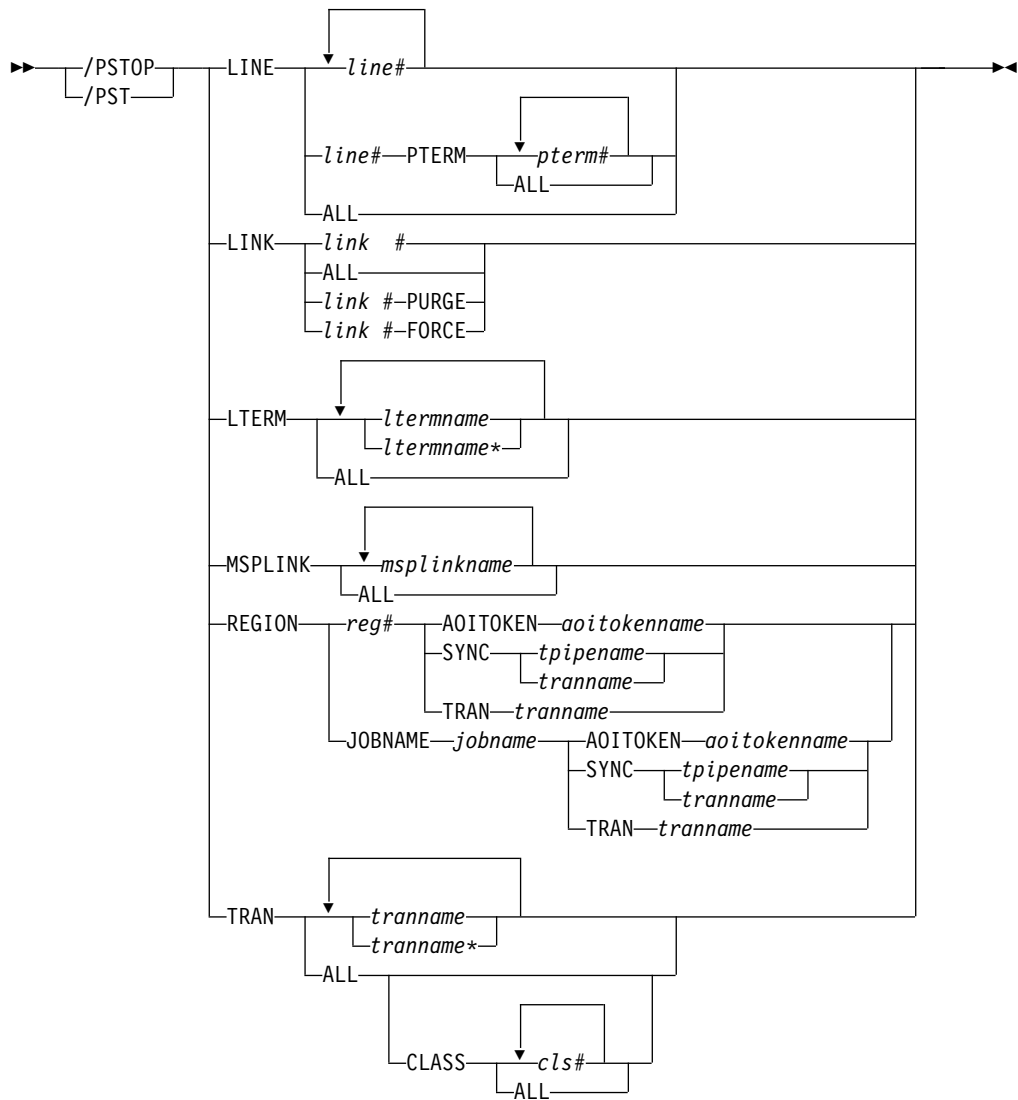
Keywords

The following keywords are valid for the /PSTOP command:

LINE

Specifies the IMS communication line to stop.

LINK

Specifies the link to be stopped; the partner link in another IMS system stops itself and notifies the master terminal operator of that system..

> **PURGE**
>
> PURGE can be used only for one logical link whose physical link is channel-to-channel. PURGE must be used when the partner link is in a system that failed. Otherwise, the link will not become idle after it is stopped.

> **FORCE**
>
> The FORCE keyword is for TCP/IP and VTAM links and is intended for use when a link does not clean up and assume a PSTOPPED IDLE status during normal PSTOP processing, even though the session is terminated.

After /PSTOP processing is completed on one side of the link, the other side is displayed. If the other side is not in a PSTOPPED IDLE state, the operation must be repeated in the partner IMS system. When both sides are in the PSTOPPED IDLE state, the /RSTART LINK command can be issued to restart the link.

For TCP/IP links, the FORCE option is useful for shutting down an MSC TCP/IP link that does not shut down normally after the link was shut down in the partner IMS system. A link that does not shut down normally might have a NOTIDLE-C status or might otherwise fail to assume a PSTOPPED, IDLE status.

For TCP/IP links, you can issue /PSTOP with the FORCE option at any time. You are not required to shut down a link normally before using the FORCE option.

IMS performs the following actions when processing the FORCE option for a TCP/IP link:

- Shuts down the link in the IMS where the command is issued
- Notifies the local IMS Connect instance to clean up the send socket
- Issues error message DFS3177E MSC DETECTED AN ERROR RETCODE = 00000000, RSNCODE = 00000070, LOSTSESS = FORCESTO
- Issues informational message DFS2169I DISCONNECTION COMPLETED ON LINK
- Places the link in PSTOPPED ERE IDLE status

Note: To determine if a TCP/IP link session is still active in IMS Connect, issue the WTOR command VIEWMSC *lclplkid* on the local IMS Connect instance. You can also use the z/OS MODIFY command QUERY MSC or the IMS type-2 command QUERY IMSCON TYPE(MSC).

For VTAM links, the FORCE option can be used with some VTAM commands to idle and clean up the VTAM link within IMS.

IMS performs the following actions when processing the FORCE option for VTAM links:

1. Determines if the link started PSTOP processing. If PSTOP processing started, the link displays as PSTOPPED NOTIDLE.
2. Tests if the hang condition is due to an outstanding VTAM request. If so, IMS issues an inquire request to VTAM to determine if the session is inactive. In that case, IMS simulates the completion of the VTAM request to enable the link to complete PSTOP processing.

You can determine if a session is still active to VTAM by issuing the command DISPLAY NET,SESSIONS,LU1=applid1,LU2=applid2,SCOPE=ALL,LIST=ALL. If it is active, note the SID of the session.

If the VTAM session is not active and it has a PSTOPPED NOTIDLE status to IMS, then issue /PSTOP LINK x FORCE.

If the session is still active to VTAM, then issue VARY NET,TERM,SID=x,NOTIFY=YES,SCOPE=ALL,TYPE=FORCE to VTAM to terminate the session.

Under normal conditions, the VTAM VARY NET,TERM command terminates the session in VTAM and cause IMS to PSTOP and IDLE the link associated with the VTAM session. If the VTAM VARY command does terminate the VTAM session but does not PSTOP and IDLE the link, then

the IMS /PSTOP LINK FORCE command can be used to complete the PSTOP and cleanup processing within IMS.

IMS replies with `DFS058 /PSTOP LINK COMPLETED EXCEPT LINK x`, if the /PSTOP LINK x FORCE command cannot be executed because:

- The session is still active to VTAM.
- Normal PSTOP processing is not started (issue /PSTOP without the FORCE keyword in this case).
- PSTOP processing is not completing because of some reason other than an incomplete VTAM request.

LTERM

Specifies the logical terminal that is to be stopped from sending and receiving messages.

The /PSTOP LTERM command has no effect on an LTERM that is in QLOCK state, or is a remote logical terminal. The LTERM parameter can be generic, where the generic parameter specifies existing LTERMs.

The /PSTOP LTERM command is valid only for LTERMs that belong to nodes that are logged on.

If time-controlled operations (TCO) is active, the /PSTOP LTERM command is valid for the DFSTCF TCO LTERM, even if the DFSTCF LTERM is not allocated to a node.

MSPLINK

Stops logons to an MSC TCP/IP or VTAM physical link and enables the operator to issue the /MSASSIGN command to reassign logical links to the physical link. Any links in sessions that were not stopped by the /PSTOP command are not affected by an /MSASSIGN command.

After the /PSTOP command is issued, the status of the link is either PSTOPPED or, for links in a TCP/IP generic resource group, PSTOPGEN.

After the logical link assignments are complete, issue the /RSTART command to permit logons to the physical link.

The /PSTOP MSPLINK command does not apply to CTC or MTM links.

REGION

If the TRAN keyword is specified, the message region is not stopped. A QC status (no more messages) is returned to the application program currently active in the specified region. The scheduler continues to schedule available transactions in the referenced region.

The /PSTOP REGION command is ignored unless both of the following occur:

- An active transaction type is specified.
- The referenced message region is processing transactions with the wait-for-input option, or the region is an MPP.

If the AOITOKEN keyword is specified, the AO application in wait AOI token state is posted and receives AIB return code X'00000004' and reason code X'0000004C'.

If JOBNAME keyword is specified, the job name for the dependent region must be 1-8 alphanumeric or national ($,#,@) characters. The first character of the job name must be either alphabetic or national.

If the SYNC keyword is specified, the user application in wait synchronous callout state is posted and receives AIB return code X'00000100' with reason code X'0000010C'.

The X'00000100' return code with the X'0000010C' reason code is also returned if the SYNC keyword is used while a user application is waiting for a reply to a synchronous program switch request.

TRAN

Stops the scheduling of transactions; however, the transactions will continue to be processed until the limit count is reached. If the limit count is large, the processing interval will be long. The /DISPLAY command ascertains the status of the transaction; the /ASSIGN command alters the status of the transaction.

If a region is scheduled against a process stopped transaction and there are no more messages available for that transaction, the region does not wait for the next message (wait-for-input-mode). Instead, a QC status (no more messages) is returned to the application. If the region is scheduled and waiting for the next message when the command is entered, the region is notified and a QC status is returned to the application.

A batch message processing region (BMP) scheduled against wait-for-input (WFI) transactions returns a QC status code (no more messages) for /PSTOP REGION, /DBD, /DBR, or /STA commands only.

The /PSTOP command cannot stop the scheduling of Fast Path exclusive transactions but can be used to stop Fast Path potential transactions.

The /PSTOP TRAN command cannot be used for Fast Path exclusive transactions or CPI Communications driven transaction programs.

The TRAN parameter can be generic where the generic parameter specifies transactions that already exist.

In a shared-queues environment, the /PSTOP TRAN command will result in IMS deregistering interest for the transaction, which indicates that the transaction cannot be scheduled at that IMS.

The /PSTOP TRAN command is not allowed for a transaction that is marked bad with the `NOTINIT-15-REPOCHGLIST` reason code because the IMS change list processing is not complete for the transaction or the change list processing failed.

Usage notes

If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and the /PSTOP command processes the rest of the parameters.

The /PSTOP command can be used to reset conditions previously established with the /START, /RSTART, /PURGE, or /MONITOR command.

In a single IMS system, or in the local system in a multiple system configuration, IMS system messages such as broadcast text and terminal status messages (`DFS059 TERMINAL STARTED`) are not affected by the /PSTOP command. In a multiple system configuration, broadcast messages are queued but not sent across stopped links.

This command can be issued to an IMSplex using the Batch SPOC utility.

If a synchronous program switch request (a DL/I ICAL request to an IMS transaction) stalls for a long period of time, you can use the /DISPLAY ACTIVE REGION command to determine which region contains the stalled application. Then, issue /PSTOP to stop the application and halt the request.

Equivalent IMS type-2 commands

The following table shows variations of the /PSTOP command and the IMS type-2 commands that perform similar functions.

Table 6. Type-2 equivalents for the /PSTOP command

Task	/PSTOP command	Similar IMS type-2 command
Stops the scheduling of transactions.	/PSTOP TRAN *tranname*	UPDATE TRAN(*tranname*) START(Q) STOP(SCHD)
Stops logons to the physical link (only for MSC VTAM links).	/PSTOP MSPLINK *mplinkname* | ALL	UPDATE MSPLINK NAME(*msplinkname* | *) STOP(LOGON)
Posts the regions in which the program is scheduled so that a new refreshed copy of the program can be obtained on the next schedule of the program.	/PSTOP REGION TRAN *tranname*	UPDATE PGM START(REFRESH)

Examples

The following are examples of the /PSTOP command:

Example 1 for /PSTOP command

Entry ET:
```
/PSTOP LINE 4 PTERM 1
```

Response ET:
```
DFS058I  PSTOP COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL PSTOPPED
```

Explanation: LINE 4 PTERM 1 is not sent application program or message switch output and is not allowed to send input. Output messages for the terminal continue to be queued.

Example 2 for /PSTOP command

Entry ET:
```
/PSTOP LINE 4 6 200
```

Response ET:
```
DFS058I  PSTOP COMMAND COMPLETED EXCEPT LINE 200
```

Explanation: LINE 4 and LINE 6 are not allowed to send or receive messages. Message queuing continues. Line 200 is an invalid line number.

Example 3 for /PSTOP command

Entry ET:

```
/PSTOP LINK 2 3 4
```

Response ET:

```
DFS058I  PSTOP COMMAND COMPLETED
```

Response ET:

```
DFS2169I  DISCONNECTION COMPLETED ON LINK 2
```

Explanation: Logical link 2 is disconnected. This message is received for each logical link that is disconnected.

Response Remote MT:

```
DFS2161I  LINK 2 STOPPED BY PARTNER
DFS2161I  LINK 3 STOPPED BY PARTNER
DFS2161I  LINK 4 STOPPED BY PARTNER
```

Explanation: Logical links 2, 3, and 4 stop processing messages. Output queuing continues.

Response ET:

```
DFS2169I  DISCONNECTION COMPLETED ON LINK 3
```

Explanation: Logical link 3 is disconnected. This message is also received when logical link 4 disconnects.

Example 4 for /PSTOP command

Entry ET:

```
/PSTOP LINK ALL
```

Response ET:

```
DFS058I  PSTOP COMMAND COMPLETED
```

Response Remote MT:

```
A DFS2161 LINK n STOPPED BY PARTNER message
is received for each logical link that was
operational when /PSTOP was entered.
```

Explanation: Output to all logical links stops. Output queuing continues. Input is not allowed.

Response ET:

```
DFS2169I  DISCONNECTION COMPLETED ON LINK XXX
```

Explanation: As each logical link is disconnected, this message is received.

Example 5 for /PSTOP command

Entry ET:

```
/PSTOP LINK 1 2 3 PURGE
```

Response ET:

```
DFS2272I PURGE KEYWORD INVALID, ONLY ONE
CTC LINK ALLOWED
```

Explanation: Only one link can be specified with the PURGE keyword.

Example 6 for /PSTOP command

Entry ET:
```
/PSTOP LINK 2 PURGE
```

Response ET:
```
DFS2273I PURGE KEYWORD REJECTED, CURRENT STATUS
OF LINK IS NORMAL
```

Explanation: The partner system has not failed and the link appears to be working.

Example 7 for /PSTOP command

Entry ET:
```
/PSTOP LTERM APPLE, TREE
```

Response ET:
```
DFS058I  PSTOP COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL PSTOPPED
```

Explanation: The physical terminals associated with logical terminals APPLE and TREE are not sent output that is destined for logical terminals APPLE or TREE, or allowed to enter input. Output queuing continues.

Example 8 for /PSTOP command

Entry ET:
```
/PSTOP MSPLINK ALL
```

Response ET:
```
DFS058I  COMMAND COMPLETED
```

Explanation: All the VTAM physical links are stopped from receiving logons. Any links in session are not affected.

Example 9 for /PSTOP command

Entry ET:
```
/PSTOP TRAN SEED
```

Response ET:
```
DFS058I  PSTOP COMMAND COMPLETED
```

Explanation: Transaction code SEED can no longer be scheduled. Queuing of the transaction continues.

Example 10 for /PSTOP command

Entry ET:
```
/PSTOP TRAN ALL CLASS 3
```

Response ET:

```
  DFS058I  PSTOP COMMAND COMPLETED
```

Explanation: All transactions associated with class 3 can no longer be scheduled. Queuing of the transactions continues.

Example 11 for /PSTOP command

Entry ET:

```
  /PSTOP REGION 1 TRAN XYZ
```

Response ET:

```
  DFS058I  PSTOP COMMAND IN PROGRESS
```

Response ET:

```
  DFS0569I PSTOP COMPLETE FOR REGION 1 TRAN XYZ
  DFS0566I PSTOP NOT VALID FOR TRAN XYZ
```

Explanation: If the DFS0569I message prints, processing of the transaction type, xyz, is stopped in message region 1. If the DFS0566I message prints, the command was ignored because the two required conditions were not satisfied.

Example 12 for /PSTOP command

Entry ET:

```
  /PSTOP REGION 2 AOITOKEN AOITOK2
```

Response ET:

```
  DFS058I PSTOP COMMAND IN PROGRESS
```

Response MT:

```
  DFS0569I PSTOP OR STOP COMPLETE FOR REGION 2 AOIT AOITOK2.
```

Explanation: If the DFS0569I message prints, the AO application in region 2 waiting for a message for AOI token AOITOK2 is posted; the application receives AIB return code X'00000004' and reason code X'0000004C'.

Example 13 for /PSTOP command

Entry ET:

```
  /PSTOP REGION 2 AOITOKEN  AOITOK2
```

Response ET:

```
   DFS058I PSTOP COMMAND IN PROGRESS
```

Response MTO:

```
   DFS1190I REGION 2 NOT WAITING ON AOITOKEN AOITOK2
```

Explanation: If the DFS1190I message prints, the command is ignored because region 2 was not waiting on AOI token AOITOK2.

Example 14 for /PSTOP command

Entry ET:

```
                             /DIS ACTIVE REG
```

Response ET:

```
 REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
     1 MPP610C   TP    NQF1      PMVAPZ12 ACTIVE            1, 2, 3, 4
       BATCHREG  BMP   NONE
       FPRGN     FP    NONE
       DBTRGN    DBT   NONE
       DBRECTA9  DBRC
       DLIECTA9  DLS
       *96081/150611*
```

Explanation: Message processing program PMVAPZ12 is processing transaction NQF1. The job name of the region is MPP610C

Entry ET:

```
 /PSTOP REGION JOBNAME MPP610C TRAN NQF1
```

Response ET:

```
 DFS058I PSTOP COMMAND IN PROGRESS
 DFS0569I PSTOP OR STOP COMPLETE FOR REGION 00001 TRAN NQF1
```

Response ET:

```
 DFS058I PSTOP COMMAND IN PROGRESS
 DFS0566I PSTOP NOT VALID FOR TRAN NQF1
```

Explanation: If the DFS0569I message is displayed, processing if the transaction type, NQF1, is stopped in message region 1. If the DFS0566I message is displayed, the command was ignored because the two required conditions were not satisfied.

Example 15 for /PSTOP command

Entry ET:

```
 /DIS ACTIVE REG
```

Response ET:

```
 REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
     1 MPP1A     TP    APOL11    APOL1    WAIT-CALLOUT     1
                 TMEM:   HWS1              TPIPE:  TPIPE1
       JMPRGN    JMP   NONE
       JBPRGN    JBP   NONE
       BATCHREG  BMP   NONE
       FPRGN     FP    NONE
       DBTRGN    DBT   NONE
       DBRZCSAJ  DBRC
       DLIZCSAJ  DLS
 *08235/173441*
```

Explanation: Message processing program APOL1 is processing transaction APOL11, which is waiting for a response to a synchronous callout request (WAIT-CALLOUT) from transaction pipe TPIPE1.

Entry ET:

```
 /PSTOP REGION 1 SYNC TPIPE1
```

Response ET:

```
 DFS058I PSTOP COMMAND IN PROGRESS
 DFS0569I PSTOP OR STOP COMPLETE FOR REGION 00001 SYNC TPIPE1
```

Response ET:

```
DFS058I PSTOP COMMAND IN PROGRESS
DFS1190I REGION 00001 NOT WAITING ON SYNTPIPE TPIPE1
```

Explanation: If the DFS0569I message prints, the user application in wait synchronous callout state is posted; the application receives AIB return code X'00000100' and reason code X'0000010C'. If the DFS1190I message prints, the command is ignored because region 1 was not waiting for a response to a synchronous callout request from transaction pipe TPIPE1.

Example 16 for /PSTOP command

This example demonstrates a scenario for using PSTOP to halt a stalled synchronous program switch request. The following command is issued to determine the status of a waiting program:

```
/DIS ACTIVE REG
```

The following response is returned:

```
REGID JOBNAME   TYPE TRAN/STEP PROGRAM STATUS       CLASS
    1 MPP1A     TP   APOL11    APOL1   WAIT-CALLOUT 1
                     TRAN:SKS1                 END TIME: 2010.172 12:45:04
      JMPRGN    JMP  NONE
      JBPRGN    JBP  NONE
      BATCHREG  BMP  NONE
      FPRGN     FP   NONE
      DBTRGN    DBT  NONE
      DBRZCSAJ  DBRC
      DLIZCSAJ  DLS
*10172/120101*
*
```

The message processing program APOL1 is processing transaction APOL12. The transaction is waiting for a synchronous program switch response from transaction SKS2. The following command is issued to end the wait state:

```
/PSTOP REGION 1 SYNC SKS2
```

The command will receive one of the two possible responses. The following response indicates that the PSTOP command successfully ended the wait state:

```
DFS058I PSTOP COMMAND IN PROGRESS
DFS0569I PSTOP OR STOP COMPLETE FOR REGION 00001 SYNC SKS2
```

You will receive a different response if the PSTOP command executes successfully but the specified region is not waiting for the specified transaction to respond:

```
DFS058I PSTOP COMMAND IN PROGRESS
DFS1190I REGION 00001 NOT WAITING ON TRAN SKS2
```

Related reference:

Chapter 16, "/RSTART command," on page 795

"UPDATE TRAN command" on page 1326

Chapter 4. /PURGE command

The /PURGE command stops input for a particular communication line, terminal, or logical link path, or stops input messages destined for a particular transaction code.

Messages can be sent to the specified communication line or terminal, and transactions can still be scheduled.

Any APPC requests that are queued in 64-bit storage as a result of APPC/IMS flood control are rejected with sense code `TP_Not_Available_No_Retry`.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 34
- "Usage notes" on page 35
- "Equivalent IMS type-2 commands" on page 35
- "Examples" on page 36

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 7. Valid environments for the /PURGE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/PURGE	X		X
APPC	X		X
CLASS	X		X
FPPROG	X		X
FPRGN	X		X
LINE	X		X
LTERM	X		X
MSNAME	X		X
PTERM	X		X
TRAN	X		X

Syntax

```
>>──┬─/PURGE─┬──┬─APPC──────────────────────────────────────────────┬──><
    └─/PUR───┘  │              ┌──────────┐                          │
               │              ▼          │                          │
               ├─FPPROG──┬──── pgmname ──┴──┐                        │
               │         └─ALL─────────────┘                        │
               │              ┌───────────┐                         │
               │              ▼           │                         │
               ├─FPRGN───┬──── fprgnname ─┴─┐                       │
               │         └─ALL──────────────┘                       │
               │              ┌─────────┐                           │
               │              ▼         │                           │
               ├─LINE────┬──── line# ───┴───────────────┐           │
               │         │              ┌──────────┐    │           │
               │         │              ▼          │    │           │
               │         ├─ line# ─PTERM──── pterm# ┴───┤           │
               │         └─ALL──────────────────────────┘           │
               │              ┌───────────────┐                     │
               │              ▼               │                     │
               ├─LTERM───┬──┬─ ltermname ──┬──┴──┐                  │
               │         │  └─ ltermname* ─┘     │                  │
               │         └─ALL───────────────────┘                  │
               │              ┌─────────────┐                       │
               │              ▼             │                       │
               ├─MSNAME──┬──┬─ msname ──┬───┴──┐                    │
               │         │  └─ msname* ─┘      │                    │
               │         └─ALL─────────────────┘                    │
               │              ┌──────────────┐                      │
               │              ▼              │                      │
               └─TRAN────┬──┬─ tranname ──┬──┴─────────────────────┤
                         │  └─ tranname* ─┘                         │
                         └─ALL───┬──────────────────────────────────┘
                                 │          ┌───────┐
                                 │          ▼       │
                                 └─CLASS──┬─ cls# ──┴─┐
                                          └─ALL───────┘
```
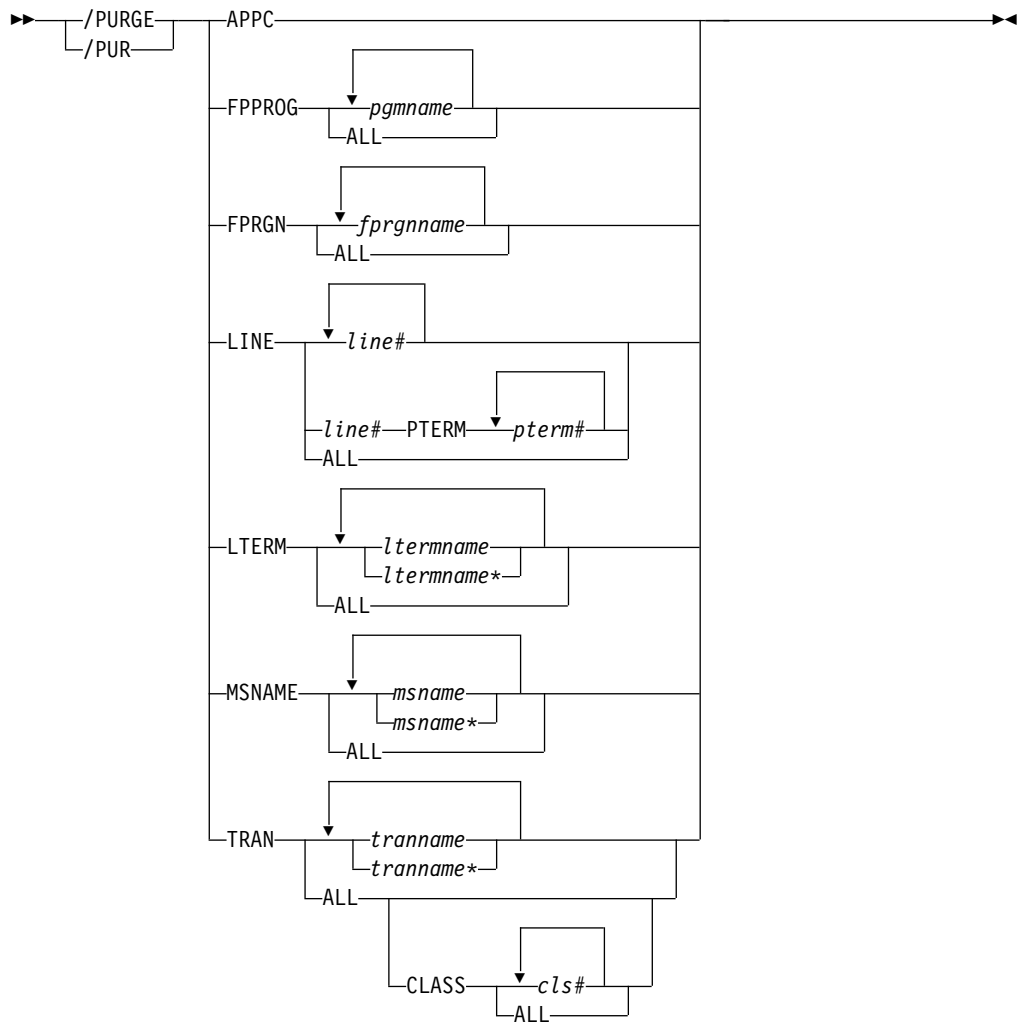
Keywords

The following keywords are valid for the /PURGE command:

APPC

> Is used to purge incoming transactions. All new requests by APPC/z/OS to schedule a transaction in IMS are rejected with `TP_Not_Available_No_Retry`. Transactions that IMS has already received are processed normally. Sending of output to LU 6.2 devices proceeds normally. Because /PURGE APPC does not call to APPC/z/OS, the rejection of transaction scheduling is done by the schedule exit of IMS only.
>
> **Note:** The sense code returned to the LU 6.2 remote device for an incoming ATTACH to a purged APPC/IMS system is determined by APPC/z/OS, and it might differ from release to release. In general, the remote LU 6.2 application should wait for a period of time after rejection before any attempts to reestablish a session with IMS.
>
> The /PURGE APPC command sets the PURGING status and resets conditions previously set by the /START APPC command. The command is rejected if the APPC is already in DISABLED, FAILED, STOPPED, or CANCEL state.

FPPROG

Specifies the PSB name of the message-driven program to be terminated.

/PURGE takes message-driven programs out of wait-for-input mode and terminates them as soon as their load balancing group message queue is empty.

Use the FPPROG keyword with caution, because all Fast Path message-driven programs that are using a PSB with the same name will be terminated.

FPRGN

Specifies the region identifier of the message-driven program to be terminated.

LINE

Specifies the communication line for which input is to be stopped.

LTERM

Specifies the logical terminal for which input is to be stopped.

The /PURGE LTERM command is rejected for LTERMs in QLOCK state. (QLOCK indicates that the LTERM is locked from sending any further output or from receiving input that can create additional output for the same LTERM until the state is reset by a specific request received on the session.) /PURGE LTERM is also rejected for remote logical terminals. The LTERM supports generic parameters where the generic parameter specifies LTERMs that already exist.

The /PURGE LTERM command is valid only for LTERMs that belong to nodes that are logged on.

MSNAME

Specifies the logical link path in a multiple systems configuration for which input is to be stopped. The MSNAME keyword supports generic parameters.

TRAN

Specifies the transaction code for which input messages are to be stopped.

The TRAN parameter can be generic where the generic parameter specifies transactions that already exist.

The /PURGE TRAN command is not allowed for a transaction that is marked bad with the `NOTINIT-15-REPOCHGLIST` reason code because the IMS change list processing is not complete for the transaction or the change list processing failed.

Usage notes

The /PURGE command validity checks all parameters entered by the terminal operator. If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and the /PURGE command processes the rest of the parameters.

The /PURGE command can be used to reset conditions previously set by the /START, /RSTART, /STOP, /PSTOP, or /MONITOR command.

This command can be issued to an IMSplex using the Batch SPOC utility.

Equivalent IMS type-2 commands

The following table shows variations of the /PURGE command and the IMS type-2 commands that perform similar functions.

Table 8. Type-2 equivalents for the /PURGE command

Task	/PURGE command	Similar IMS type-2 command
Stops input messages for a particular transaction code.	/PURGE TRAN *tranname*	UPDATE TRAN NAME(*tranname*) START(SCHD) STOP(Q)

Examples

The following are examples of the /PURGE command:

Example 1 for /PURGE command

Entry ET:
```
/PURGE FPPROG ALL
```

Response ET:
```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: All message-driven programs are taken out of wait-for-input mode and terminated by PSB name as soon as their load balancing group message queue is empty.

Example 2 for /PURGE command

Entry ET:
```
/PURGE FPRGN ALL
```

Response ET:
```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: All message-driven programs are taken out of wait-for-input mode and terminated by region identifier as soon as their load balancing group message queue is empty.

Example 3 for /PURGE command

Entry ET:
```
/PURGE LINE 4
```

Response ET:
```
DFS058I PURGE COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL PURGING
```

Explanation: All physical terminals associated with line 4 can receive output sent to them but are not allowed to enter input.

Example 4 for /PURGE command

Entry ET:
```
/PURGE LINE 5 7 400
```

Response ET:

```
DFS058I  PURGE COMMAND COMPLETED EXCEPT LINE 400
```

Explanation: All physical terminals associated with line 5 and line 7 can receive output but are not allowed to enter input. Line 400 is an invalid line number.

Example 5 for /PURGE command

Entry ET:
```
/PURGE MSNAME BOSTON
```

Response ET:
```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: All messages from a terminal (primary requests), except messages continuing a conversation, will not be queued for the destinations represented by MSNAME BOSTON. This includes all messages destined for remote transactions with the SYSID of the MSNAME, and for remote logical terminals associated with this MSNAME.

Example 6 for /PURGE command

Entry ET:
```
/PURGE TRAN PIT, SEED
```

Response ET:
```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: Transactions PIT and SEED can still be scheduled but input for these transactions cannot be queued unless the input originates as output from an application program.

Example 7 for /PURGE command

Entry ET:
```
/PURGE TRAN ALL CLASS 2
```

Response ET:
```
DFS058I  PURGE COMMAND COMPLETED
```

Explanation: All transactions associated with class 2 are marked as purged. No further transactions are queued from terminals.

Related reference:
"UPDATE TRAN command" on page 1326
"UPDATE MSNAME command" on page 1211

Chapter 5. QUERY commands

Use the IMS QUERY commands to display information about IMS resources.

The QUERY commands return information based on the keyword specified. All of the QUERY commands are type-2 commands and can be issued from the OM API.

These commands can be issued through TSO SPOC, IBM Management Console for IMS and DB2® for z/OS, the Manage Resources options in the IMS Applications menu, or other user-written programs that issue commands through OM API. These commands can also be issued to an IMSplex by using the Batch SPOC utility.

The value shown in the QCNT column of the command output has different meanings for different commands, as described in individual QUERY command topics.

QUERY commands are:

- "QUERY USERID command" on page 721

Related concepts:

➡ CSL RM, IMS, and Repository Server termination (System Administration)

➡ IMSRSC repository administration (System Administration)

➡ Resource lists for the IMSRSC repository (System Definition)

QUERY AREA command

Use the QUERY AREA command, which is a type-2 command, to display information about DEDB areas and area data set information.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 41
- "Usage notes" on page 43
- "Equivalent IMS type-1 commands" on page 45
- "Output fields" on page 45
- "Return, reason, and completion codes" on page 47
- "Examples" on page 48

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY AREA command and keywords can be issued.

Table 9. Valid environments for QUERY AREA command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUERY AREA	X	X	
NAME	X	X	
OPTION	X	X	
SHOW	X	X	
STATUS	X	X	

Syntax

```
►►─┬─QUERY─┬─AREA─┬────────NAME(*)────────┬──┬──────────────────┬──►
   └─QRY───┘      │         ┌──,─────────┐ │  └─OPTION(REFRESH)─┘
                  │         ▼            │ │
                  └─NAME(─┬───areaname───┴─┬─)─┘
                         └─areaname*──────┘
```

```
                ,                            ,
    |-SHOW(-+--ACCTYPE--+-)-|    |-STATUS(-+--EEQE--------+-)-|
           +--ALL-------+               +--IC----------+
           +--CI--------+               +--MAS---------+
           +--LOCAL-----+               +--NOTOPEN-----+
           +--GLOBAL----+               +--OFR---------+
           +--STATUS----+               +--OPEN--------+
           +--UTILITY---+               +--PRELOAD-----+
                                        +--PREOPEN-----+
                                        +--QUIESCED----+
                                        +--QUIESCING---+
                                        +--RECALL------+
                                        +--RECOVERY----+
                                        +--RECOVNEEDED-+
                                        +--SAS---------+
                                        +--SHARED------+
                                        +--STOPPED-----+
                                        +--UTIL--------+
                                        +--VSO---------+
```
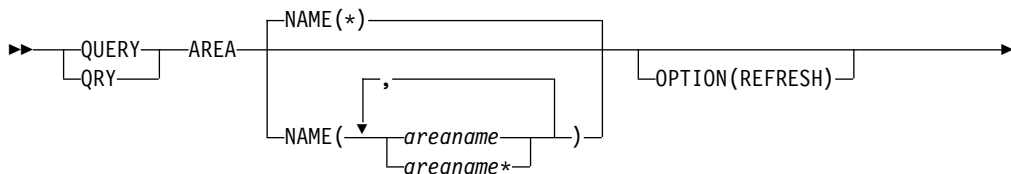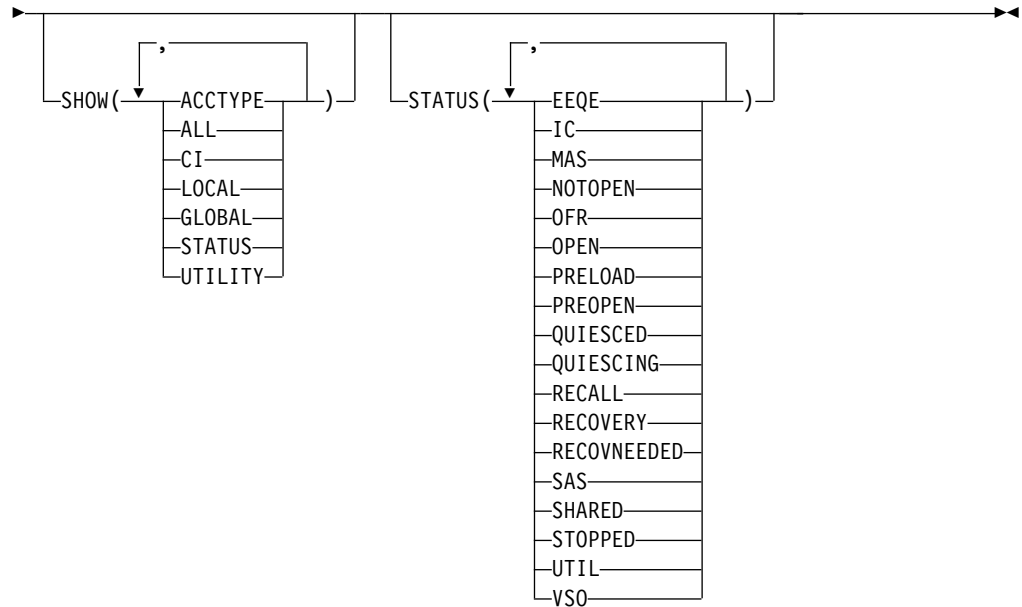
Keywords

The following keywords are valid for the QUERY AREA command:

NAME()
>   Specifies the names of the specific areas that are to be processed or the group of areas whose names match the generic or wildcard parameter specified.
>
>   If the STATUS filter is not specified, all the area names that match the NAME parameter are returned. The NAME keyword is optional and the default is NAME(*).

OPTION(REFRESH)
>   Specifies the additional functions to be performed.
>
>   **REFRESH**
>   >   Refreshes the control interval information for the sequential dependent space and the independent overflow part of the direct addressable space for the area. OPTION(REFRESH) returns the CI information even if SHOW(CI) is not specified.
>   >   OPTION(REFRESH) results in an I/O request that is performed only at the command master IMS if the area is open at the master. If the command is routed to multiple IMS systems in the IMSplex, all non-master IMS systems return local CI information. The age of the local information shown on the non-master IMS systems is at most as old as the value set on the IOVFI= IMS control region startup parameter.
>   >
>   >   The CI information is returned only if the area is open at the IMS. If the area is not open, blanks will be returned in the output CI columns SDAT, SDAU, LDAT, and LDAU. If the area is not open at the command master IMS, no refresh of the control intervals is performed. The QRY AREA NAME(areaname) SHOW(CI) OPTION(REFRESH) must be routed to the IMS where the area is open to get the current SDEP and IOVF CI information.

If the default NAME(*) is used with the REFRESH keyword, or if large numbers of areas are processed with the REFRESH keyword, performance may be affected depending on the size and number of areas involved. If large areas, numerous areas, or both are involved, the control regions processing the command may appear stopped.

SHOW()

Specifies the area output fields to be returned. The area name, the DEDB name, and the area data set information are always returned along with the name of the IMS that created the output for the area and the completion code.

The filters supported with the SHOW keyword are:

ACCTYPE

Returns DEDB area access when the area access has been changed explicitly by an UPDATE AREA START(ACCESS) SET(ACCTYPE()) command. If an area access is not changed explicitly by an UPDATE AREA command, the area access is blank. If none of the areas specified in the NAME() keyword of the QUERY AREA NAME() SHOW(ACCTYPE|ALL) command has an explicit area access value, then the area access (LAcc) output field column is not displayed.

Type of access to the DEDB area, which can be one of the following:

- BRWS - Read only
- EXCL - Exclusive
- READ - Read
- UPD - Update

ALL Returns all the output fields.

If global area status is maintained, the QUERY AREA command will return global information from the RM resource structure. The command master IMS returns the status on a separate response line. The area does not have to be defined at the command master.

CI Control intervals.

Returns the total and unused control intervals defined for the sequential dependent space and the total and unused control intervals for the independent overflow part of the direct addressable space.

LOCAL

For output fields that have both local and global values, this option returns only the local values. If used with another SHOW keyword to request a specific output field, this option requests that only the local value of the specified output field is returned. Local output is returned by each IMS that processes the command.

GLOBAL

For output fields that have both local and global values, this option returns only the global values. If used with another SHOW keyword to request a specific output field, this OPTION requests that only the global value of the specified output field is returned. IMS retrieves global information from RM. Global output is returned only by the command master IMS.

If global area status is maintained, the QUERY AREA command will return global information from the RM resource structure. The command master IMS returns the status on a separate response line. The area does not have to be defined at the command master.

STATUS
Local and global area status.

Global status is returned if global area status is maintained in RM. Global status is returned only by command master IMS and is returned on a separate response line.

If the area has an EEQE status, the count of I/O errors or write error EEQE for the area are also returned. If an area's status is OPEN, that status is not maintained in the RM resource structure.

If global area status is maintained, the QUERY AREA command will return global information from the RM resource structure. The command master IMS returns the status on a separate response line. The area does not have to be defined at the command master.

UTILITY
Returns the utility information about the utility that has the area open. The utility name, the total and available buffers in the private pool, and the utility UOW are returned.

STATUS()
Selects areas for display that match the NAME parameter and possess at least one of the specified area status locally.

The status filter allows for additional filtering by area status. The output returned when the STATUS filter is specified includes the status of the area that caused the area name to be displayed even if the SHOW(STATUS) is not specified.

Status parameters are the same as the values displayed in the QUERY AREA local status output column. See Table 11 on page 44.

Usage notes

The command can only be specified through the Operations Manager (OM) API and can only be processed by DB/DC and DBCTL environments.

The QUERY AREA command is valid on the XRF alternate as well as the RSR tracker.

The command syntax for this command is defined in XML and is available to automation programs which communicate with OM.

How the SHOW keyword on QUERY AREA determines the type of output

The following table provides some examples of how the SHOW keyword determines the type of output returned on the QUERY AREA command.

Table 10. How the SHOW keyword on QUERY AREA determines the type of output

Form of SHOW keyword used	Type of output returned
SHOW(LOCAL)	Only those fields that are local to an IMS system. SHOW(ALL,LOCAL) provides the same output.

Table 10. How the SHOW keyword on QUERY AREA determines the type of output (continued)

Form of SHOW keyword used	Type of output returned
SHOW(GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM. SHOW(ALL,GLOBAL) provides the same output.
SHOW(ALL)	All of the output fields for those fields that have both local and global data. Both values are returned in the output.
SHOW(STATUS,GLOBAL)	Only global STATUS values.
SHOW(STATUS,LOCAL)	Only local STATUS values.
SHOW(STATUS)	Both local and global STATUS values.
SHOW(ALL,GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM. SHOW(GLOBAL) provides the same output.
SHOW(ALL,LOCAL)	Only those output fields that are local to an IMS system. SHOW(LOCAL) provides the same output.

QUERY AREA status

The following table lists the local and global area status conditions that may be returned when SHOW(STATUS) is specified.

Table 11. Status conditions for QUERY AREA

Status	Meaning
EEQE	Area has EEQEs.
IC	Area image copy is active.
MAS	Area is on a multi-area structure.
NONE	The area has no global status in the RM resource structure.
NOTOPEN	Area is not open.
OFR	Area has online forward recovery in progress to bring it up to current tracking level.
OPEN	Area is open. Not maintained in the RM resource structure.
PRELOAD	Area is defined to be preloaded.
PREOPEN	Area is defined to be preopened.
QUIESCED	The DEDB area named on the command is currently quiesced by a previous UPDATE DB START(QUIESCE) or UPDATE AREA START(QUIESCE) command.
QUIESCING	The DEDB area named on the command is currently undergoing quiesce by a previous UPDATE DB START(QUIESCE) or UPDATE AREA START(QUIESCE) command.
RECALL	Area is in recall.
RECOVERY	Area recovery in progress.
RECOVNEEDED	Area needs recovery.
SAS	Area is on a single area structure.
SHARED	Area is shared.
STA	The area is started globally.
STOACC	The area is stopped for access globally and is offline.
STOPPED	The area is stopped locally or globally.

Table 11. Status conditions for QUERY AREA (continued)

Status	Meaning
UTIL	Area is open by a utility.
VSO	Area is a VSO area.

The following table lists the ADS status conditions that can be returned for an ADS associated with an AREA when SHOW(STATUS) is specified.

Table 12. ADS status conditions for QUERY AREA

Status	Meaning
COPY-PHASE	The CREATE utility is active on this ADS and is in the COPY phase. The CREATE utility must complete before any action can be processed for the ADS.
FORMAT-PHASE	The CREATE utility is active on this ADS and is in the FORMAT phase. The CREATE utility must complete before any action can be processed for the ADS.
LONGBUSY	Area in long busy state or long busy recovery mode.
PREOPEN-FAIL	XRF PREOPEN failed for this ADS.
SEVERE-ERROR	The ADS had a severe I/O error (write error to 2nd CI).
UNAVAIL	The ADS is marked unavailable because of I/O errors.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY AREA command and the type-1 IMS commands that perform similar functions.

Table 13. Type-1 equivalents for the QUERY AREA command

QUERY AREA command	Similar IMS type-1 command
QUERY AREA	/DIS AREA area1...arean	ALL, /DIS STATUS AREA

Output fields

The following table shows the QUERY AREA output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning
> Provides a brief description of the output field.

Table 14. Output fields for QUERY AREA command

Short label	Long label	Keyword	Scope	Meaning
ADS	ADSName	N/A	N/A	ADS name. The Area data set name or names associated with the AREA.
AREA	AreaName	N/A	N/A	Area name. The Area name is always returned.
CC	CC	N/A	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned. See the return, reason, and completion codes table for QUERY AREA.
CCTXT	CCText	*error*	LCL	The completion code text that briefly explains the meaning of the completion code. The completion code text can be up to 32 bytes long.
DB	DBName	N/A	N/A	DEDB name. The DEDB name associated with the Area.
LDAT	Dir-T	CI	LCL	Local value of the total control intervals for the independent overflow part of the direct addressable space. This value only appears if the area is open.
LDAU	Dir-U	CI	LCL	Local value of the unused control intervals for the independent overflow part of the direct addressable space. This value only appears if the area is open and the IOVF count ITASK was not disabled when IOVFI=1 on the IMS Control Region startup procedure was specified. When the command is processed on an RSR tracking IMS system, no information will be returned. This value is refreshed during command processing if OPTION(IOVF) was entered on the QUERY AREA command. If OPTION(IOVF) was not specified, the value reflects the updated value from the last IOVF count ITASK.
LEQ	EQCnt	STATUS	LCL	Local value of the total control intervals for the independent overflow part of the direct addressable space. This value only appears if the area is open.
LPBA	PBuf-A	UTILITY	LCL	Available number of private buffers in private pool.
LPBT	PBuf-T	UTILITY	LCL	Total number of private buffers in private pool.
LSDT	SDep-T	CI	LCL	Local value of the total control intervals defined for the sequential dependent space. This value only appears if the area is open and SDEPs have been defined.
LSDU	SDep-U	CI	LCL	Local value of the unused control intervals defined for the sequential dependent space. This value only appears if the area is open, SDEPs have been defined, and unused SDEPs are available.

Table 14. Output fields for QUERY AREA command (continued)

Short label	Long label	Keyword	Scope	Meaning
LSTT	LclStat	STATUS	LCL	Local area status. All area status conditions that apply are returned. Area status can be one or more of the status conditions described in Table 11 on page 44.
LUOW	UOW	UTILITY	LCL	The current utility UOW for HSREORG and HSSP, otherwise it is blank.
LUTIL	UtilName	UTILITY	LCL	Utility name that has area OPEN.
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. IMS identifier of the IMS that built the output. The IMS identifier is always returned.
STT	Status	STATUS, GLOBAL	GBL	Global area status. Status can be: **STA** Area has a global status of started. **STO** Area has a global status of stopped. **STOACC** Area has a global status of stopped for access.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY AREA command. The OM return and reason codes that may be returned as a result of the QUERY AREA command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 15. Return and reason code for the QUERY AREA command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY AREA command completed successfully.
X'00000004'	X'00001010'	No resources were found to be returned. The resource name(s) specified may be invalid, or there were no resources that match the filter specified.
X'00000008'	X'0000200C'	The QUERY AREA command is not processed because no resources matched any status specified on the STATUS( ) keyword.
X'00000008'	X'00002014'	The QUERY AREA command is not processed because in invalid character is found in the area name parameter.
X'00000008'	X'00002040'	More than one filter or keyword value is specified on the QUERY AREA command. Either more than one keyword or an invalid combination of filters was specified. Check the input command and reenter the correct combinations.

Table 15. Return and reason code for the QUERY AREA command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003000'	The QUERY AREA command was successful for at least one resource name. The QUERY AREA command was not successful for one or more resource names. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY AREA command are listed in Table 16.
X'0000000C'	X'00003004'	The QUERY AREA command was not successful for all the resource name(s) specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY AREA command are listed in Table 16.
X'00000010'	X'00004024'	The QUERY AREA command cannot be processed on a non-Fast Path system.
X'00000010'	X'00004025'	The QUERY AREA command is rejected because no Fast Path areas are defined.
X'00000014'	X'00005004'	The QUERY AREA command processing terminated as a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The QUERY AREA command processing terminated because of an internal error.

The following table includes an explanation of the completion codes. Errors unique to the processing of QUERY AREA command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 16. Completion codes for the QUERY AREA command

Completion code	Meaning
0	The QUERY AREA command completed successfully for the resource.
10	Resource not found. The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. Confirm that the correct spelling of the resource name is specified on the command.

Examples

The following are examples of the QUERY AREA command:

Example 1 for QUERY AREA command

In this example, the command returns the CI and STATUS information for the area, DB21AR0, from all the IMS systems in the IMSPlex. Any ADS information, if available, is also returned by each IMS.

TSO SPOC input:
```
QRY AREA NAME(DB21AR10) SHOW(STATUS,CI)
```

TSO SPOC output:

```
AreaName ADSName MbrName  DBName     CC  SDep-T  SDep-U  Dir-T Dir-U EQCnt LclStat
DB21AR0          IMS2     DEDBJN21   0                                     PREOPEN,NOTOPEN
DB21AR0          SYS3     DEDBJN21   0   1303    1302    74    74          PREOPEN,OPEN,SHARED
DB21AR0  DB21AR01 SYS3               0                               10
DB21AR0  DB21AR02 SYS3               0                               10
```

OM API input:

```
CMD(QRY AREA NAME(DB21AR10) SHOW(STATUS,CI))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1     </xmlvsn>
<statime>2003.132 16:10:52.861123</statime>
<stotime>2003.132 16:10:52.862301</stotime>
<staseq>B968A1B61BEC302F</staseq>
<stoseq>B968A1B61C35D38E</stoseq>
<rqsttkn1>USRT005 10091052</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>AREA           </kwd>
<input>QRY AREA NAME(DB21AR0) SHOW(CI,STATUS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="ADS" llbl="ADSName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="n" key="0" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="LSDT" llbl="SDep-T" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="INT" align="right" skipb="no" />
<hdr slbl="LSDU" llbl="SDep-U" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="INT" align="right" skipb="no" />
<hdr slbl="LDAT" llbl="Dir-T" scope="LCL" sort="n" key="0" scroll="yes"
 len="7" dtype="INT" align="right" skipb="no" />
<hdr slbl="LDAU" llbl="Dir-U" scope="LCL" sort="n" key="0" scroll="yes"
 len="7" dtype="INT" align="right" skipb="no" />
<hdr slbl="LEQ" llbl="EQCnt" scope="LCL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>AREA(DB21AR0 ) MBR(IMS2    ) DB(DEDBJN21) CC(   0) LSDT( ) LSDU( )
 LDAT( ) LDAU( ) LSTT(PREOPEN,NOTOPEN) </rsp>
<rsp>AREA(DB21AR0 ) MBR(SYS3    ) DB(DEDBJN21) CC(   0) LSDT(   1303)
 LSDU(   1302) LDAT(    74) LDAU(    74) LSTT(PREOPEN,OPEN,SHARED)
</rsp>
<rsp>AREA(DB21AR0 ) ADS(DB21AR01) MBR(SYS3    ) DB( ) CC(   0) LEQ(  10) </rsp>
<rsp>AREA(DB21AR0 ) ADS(DB21AR02) MBR(SYS3    ) DB( ) CC(   0) LEQ(  10) </rsp>
</cmdrspdata>
</imsout>
```

Example 2 for QUERY AREA command

In this example, the command returns all the areas that match the wildcard name and have a status of SHARED. The status is also returned. The ADS information for the AREAs is also returned if it is available. Command response lines are not returned from IMS2 because no AREAs match the status specified. A return and reason code is returned from IMS2.

TSO SPOC input:
```
QRY AREA NAME(DB21AR1*) STATUS(SHARED)
```

TSO SPOC output:
```
Log for . . : QRY AREA NAME(DB21AR1*) STATUS(SHARED)
IMSplex . . . . . . : PLEX1
Routing . . . . . . :
Start time. . . . : 2003.132 09:13:37.93
Stop time . . . . : 2003.132 09:13:37.94
Return code . . . : 0200000C
Reason code . . . : 00003000
Command master. . : IMS2
              Return    Reason
MbrName     Code      Code
--------    ------    ------
IMS2        00000008  0000200C
AreaName ADSName  MbrName  DBName    CC EQCnt LclStat
DB21AR10          SYS3     DEDBJN21   0       PREOPEN,OPEN,SHARED
DB21AR10 DB21AR10 SYS3               0   10
DB21AR11          SYS3     DEDBJN21   0       PREOPEN,OPEN,SHARED
DB21AR11 DB21AR11 SYS3               0   10
```

OM API input:
```
CMD(QRY AREA NAME(DB21AR1*) STATUS(SHARED))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2003.132 16:13:37.940282</statime>
<stotime>2003.132 16:13:37.940959</stotime>
<staseq>B968A2538A73A707</staseq>
<stoseq>B968A2538A9DF027</stoseq>
<rqsttkn1>USRT005 10091337</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS2    ">
<typ>IMS    </typ>
<styp>DBDC    </styp>
<rc>00000008</rc>
<rsn>0000200C</rsn>
<rsntext>No resources found</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMS2    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>AREA          </kwd>
<input>QRY AREA NAME(DB21AR1*) STATUS(SHARED) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="1"
```

```
    scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="ADS" llbl="ADSName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="n" key="0" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="LEQ" llbl="EQCnt" scope="LCL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>AREA(DB21AR10) MBR(SYS3    ) DB(DEDBJN21) CC(    0)
 LSTT(PREOPEN,OPEN,SHARED) </rsp>
<rsp>AREA(DB21AR10) ADS(DB21AR10) MBR(SYS3    ) DB( ) CC(    0) LEQ( 10) </rsp>
<rsp>AREA(DB21AR11) MBR(SYS3    ) DB(DEDBJN21) CC(    0)
 LSTT(PREOPEN,OPEN,SHARED) </rsp>
<rsp>AREA(DB21AR11) ADS(DB21AR11) MBR(SYS3    ) DB( ) CC(    0) LEQ( 10) </rsp>
</cmdrspdata>
</imsout>
```

Example 3 for QUERY AREA command

In this example, the QUERY AREA SHOW(STATUS) command also returns the global status for the area. The global status is returned by the master IMS on a separate global response line.

TSO SPOC input:
```
QRY AREA NAME(DB21AR0,DB21AR1) SHOW(STATUS)
```

TSO SPOC output:

AreaName	ADSName	MbrName	DBName	CC	Status	LclStat
DB21AR0		IMS1	DEDBJN21	0	STOPPED	
DB21AR0		IMS1	DEDBJN21	0		STOPPED,NOTOPEN
DB21AR0		IMS2	DEDBJN21	0		STOPPED,NOTOPEN
DB21AR1		IMS1	DEDBJN21	0	STA	
DB21AR1		IMS1	DEDBJN21	0	OPEN	
DB21AR1	DB21AR1	IMS1		0		
DB21AR1		IMS2	DEDBJN21	0	OPEN	
DB21AR1	DB21AR1	IMS2		0		

Example 4 for QUERY AREA command

In this example, the QUERY AREA SHOW(STATUS) command also returns the global status for the area. The global status is returned by the master IMS on a separate global response line.

TSO SPOC input:
```
QRY AREA NAME(DB21AR0,DB21AR1) SHOW(STATUS)
```

TSO SPOC output:

AreaName	ADSName	MbrName	DBName	CC	Status	LclStat
DB21AR0		IMS1	DEDBJN21	0	STOPPED	
DB21AR0		IMS1	DEDBJN21	0		STOPPED,NOTOPEN
DB21AR0		IMS2	DEDBJN21	0		STOPPED,NOTOPEN
DB21AR1		IMS1	DEDBJN21	0	STA	

```
DB21AR1               IMS1    DEDBJN21  0                OPEN
DB21AR1 DB21AR1       IMS1              0
DB21AR1               IMS2    DEDBJN21  0                OPEN
DB21AR1 DB21AR1       IMS2              0
```

Example 5 for QUERY AREA command

The following example is of a query of the databases that have a status of quiesced.

TSO SPOC input:

```
QRY AREA NAME(*) SHOW(STATUS) STATUS(QUIESCED)
```

TSO SPOC output:

```
AreaName MbrName  DBName     CC LclStat
AXZY01   IM01     FPDBXYZ     0 OPEN,QUIESCED
AXZY01   IM02     FPDBXYZ     0 OPEN,QUIESCED
AXZY01   IM03     FPDBXYZ     0 OPEN,QUIESCED
```

Example 6 for QUERY AREA command

TSO SPOC input:

```
QUERY AREA NAME(DB21AR1*) SHOW(ACCTYPE)
```

TSO SPOC output:

```
AreaName ADSName  MbrName  DBName     CC  LAcc
DB21AR1           IMS1     DEDBJN21    0  READ
DB21AR1  DB21AR1  IMS1                 0
DB21AR1           IMS2     DEDBJN21    0  READ
DB21AR1  DB21AR1  IMS2                 0
DB21AR10          IMS1     DEDBJN21    0
DB21AR10          IMS2     DEDBJN21    0
DB21AR11          IMS1     DEDBJN21    0
DB21AR11          IMS2     DEDBJN21    0
```

Explanation: Database access for DEDB DEDBJN21 is UPD (Update). Area access for area DB21AR1 is READ. Area access for areas DB21AR10 and DB21AR11 are UPD, which they inherit from DEDB DEDBJN21 implicitly. Area access for areas DB21AR10 and DB21AR11 are not displayed because their area access is not changed by the UPDATE AREA START(ACCESS) SET(ACCTYPE()) command explicitly.

Example 7 for QUERY AREA command

TSO SPOC input:

```
QUERY AREA NAME(DB21AR2) SHOW(ACCTYPE)
```

TSO SPOC output:

```
AreaName ADSName  MbrName  DBName     CC
DB21AR2           IMS1     DEDBJN21    0
DB21AR2  DB21AR2  IMS1                 0
DB21AR2           IMS2     DEDBJN21    0
DB21AR2  DB21AR2  IMS2                 0
```

Explanation: Database access for DEDB DEDBJN21 is UPD (Update). Area access for area DB21AR2 is UPD, which is inherited from DEDB DEDBJN21 implicitly. Area access for area DB21AR2 is not displayed because its area access is not changed by the UPDATE AREA START(ACCESS) SET(ACCTYPE()) command

explicitly. Because area DB21AR2 is the only area in the NAME() keyword and its area access has not been changed by the UPDATE AREA START(ACCESS) SET(ACCTYPE()) command, the output field column "LAcc" is not displayed.

Related concepts:

⬛▶ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

⬛▶ /DISPLAY AREA command (Commands)

⬛▶ Command keywords and their synonyms (Commands)

QUERY DB command

Use the QUERY DB command, which is a type-2 command, to display information about databases.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 54
- "Usage notes" on page 62
- "Equivalent IMS type-1 commands" on page 63
- "Output fields" on page 63
- "Return, reason, and completion codes" on page 72
- "Examples" on page 75

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY DB command and keywords can be issued.

Table 17. Valid environments for QUERY DB command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUERY DB	X	X	
NAME	X	X	
SHOW	X	X	
STATUS	X	X	
TYPE	X	X	

Syntax

```
         NAME(*)
►►─┬─QUERY─┬─DB─┬──────────────────────────────┬──────────────────────────────►
   └─QRY───┘    │              ,               │
               │         ┌───────────┐        │
               └─NAME(─▼─┬─name──┬─)─┘
                         └─name*─┘

                                    ,
                              ┌────────────────┐
                         ┌─SHOW(─▼─┬─ACCTYPE───┬─)─┐
                         │         ├─ALL───────┤   │
                         │         ├─DEFN──────┤   │
                         │         ├─DEFNTYPE──┤   │
                         │         ├─GLOBAL────┤   │
                         │         ├─IMSID─────┤   │
                         │         ├─LOCAL─────┤   │
                         │         ├─MODEL─────┤   │
                         │         ├─RESIDENT──┤   │
                         │         ├─STATUS────┤   │
                         │         └─TIMESTAMP─┘   │
                         │    ┌─SHOW(─┬─PGM─────┬─)─┤
                         │    │       ├─SNDX────┤   │
                         │    │       ├─VERSION─┤   │
                         │    │       └─WORK────┘   │
                         └─SHOW(EXPORTNEEDED)───────┘


                      ,                              ,
               ┌─────────────┐               ┌────────────┐
►─┬────────────────────────────┬─┬──────────────────────────┬─────────────────►◄
  │         ┌─ALLOCF─────┐      │ │        ┌─DEDB─────┐       │
  └─STATUS(─▼─┬─ALLOCF───┬─)─┘  └─TYPE(─▼─┬─DEDB─────┬─)─┘
             ├─ALLOCS────┤              ├─DHISNDX──┤
             ├─BACKOUT───┤              ├─DLI──────┤
             ├─EEQE──────┤              ├─MSDB─────┤
             ├─LOCK──────┤              ├─PART─────┤
             ├─NOTINIT───┤              ├─PHDAM────┤
             ├─NOTOPEN───┤              ├─PHIDAM───┤
             ├─OFR───────┤              └─PSINDEX──┘
             ├─OLR───────┤
             ├─OPEN──────┤
             ├─QUIESCED──┤
             ├─QUIESCING─┤
             ├─RECALL────┤
             ├─RECOV─────┤
             ├─RNL───────┤
             ├─STOACC────┤
             ├─STOSCHD───┤
             └─STOUPDS───┘
```
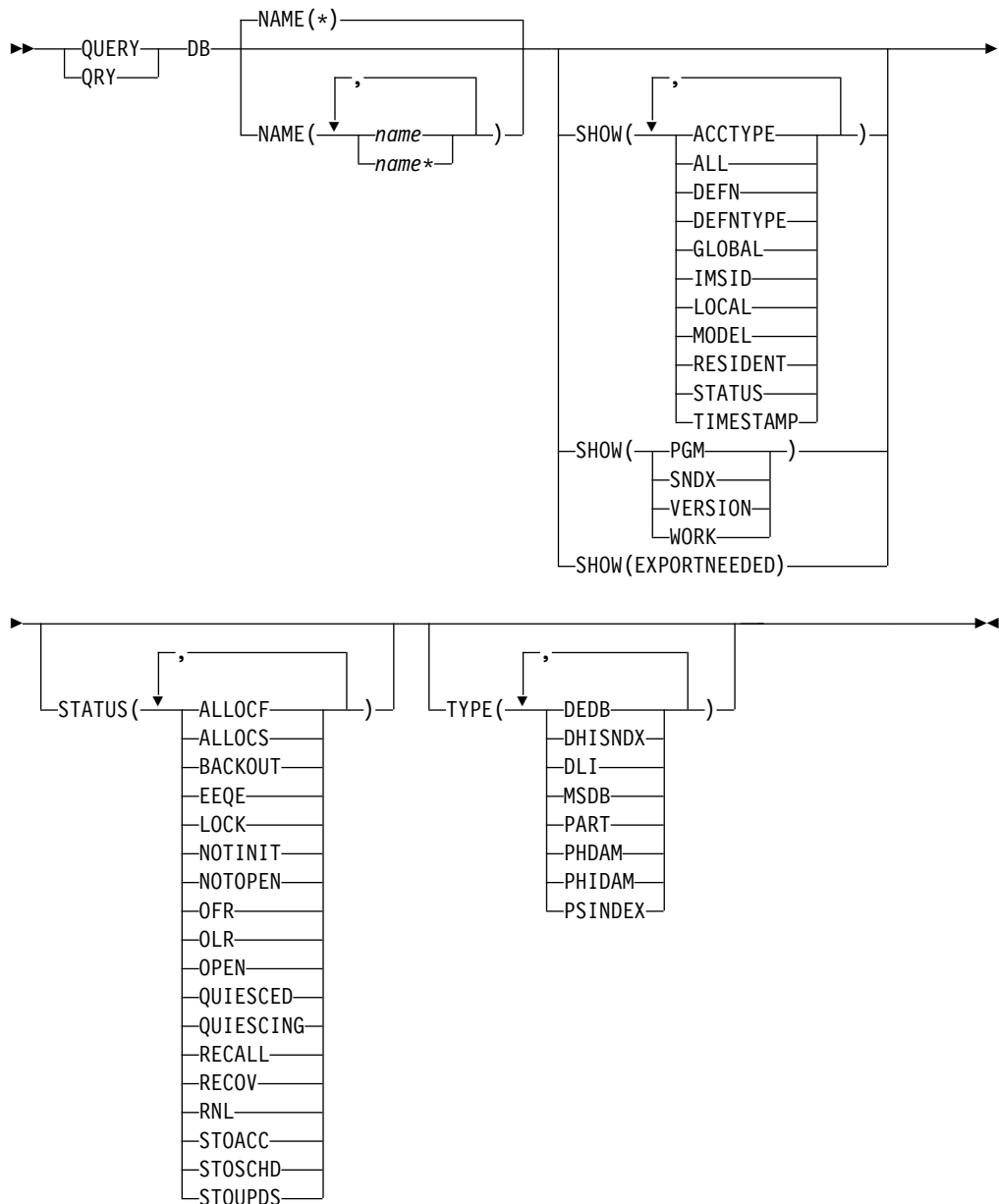
Keywords

The following keywords are valid for the QUERY DB command:

NAME()

Specifies the 1- to 8-character name of the database (DBD name). Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all database resources.

If the STATUS filter is not specified, all the database names that match the NAME parameter are returned.

The database names that match the generic or wildcard parameters are processed. Response lines are returned for all the databases names that are processed.

The database name specified can be a HALDB master or a HALDB partition. If the database name is the HALDB master, response lines are returned for the HALDB master and all of its partitions. If the database name is the HALDB partition, response lines are returned for the HALDB master and the partition name if the HALDB master has not been taken offline.

If the database name specified is a DEDB name, response lines are returned for the DEDB name and all the DEDB areas.

DEDB area information is not returned if SHOW(DEFN) is specified.

SHOW()
Specifies the database output fields to be returned. The database name and type are always returned, along with the name of the IMS that created the output for the database and the completion code. The filters supported with the SHOW keyword are:

ACCTYPE
Returns the type of access to the database or the area.

If SHOW(ACCTYPE) is specified, and IMS is not using the IMSRSC repository and global DB status is maintained, the global access type from the resource structure is returned.

If SHOW(ACCTYPE) is specified, the command master IMS is using the repository, and global status is also enabled, the global access type from the resource structure is not returned. The access type from the repository is returned.

The access type from the repository is returned if SHOW(DEFN), SHOW(DEFN,ACCTYPE), SHOW(DEFN,GLOBAL), or SHOW(DEFN,ACCTYPE,GLOBAL) is specified.

ALL
Returns all information about the database itself. Other SHOW keywords can be specified to return information about resources related to the database.

The command master IMS returns global information from the RM resource structure on a separate line, even if the database is not defined at the command master.

Restriction: The `ExportNeeded` column, which is returned when SHOW(EXPORTNEEDED) is specified, is not returned with SHOW(ALL).

DEFN
Specifies that the resource definitions are to be returned.

The database attributes that can be returned are ACCTYPE, RESIDENT, the repository create and update time stamps, and the IMS runtime create, update, import, and access time stamps.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes, including those defined globally in the repository and those defined locally in the IMS system, are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition obtained from the repository. This response line

displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- You cannot specify SHOW(DEFN) with DEFNTYPE, MODEL, STATUS, WORK, or PGM.
- When querying database information from the repository, the SHOW(DEFN) filter is not supported when used with either the TYPE or STATUS filter. The runtime filters of TYPE and STATUS are not valid with SHOW(DEFN), SHOW(DEFN,GLOBAL), SHOW(DEFN,LOCAL), SHOW(DEFN,IMSID), SHOW(DEFN,IMSID,GLOBAL) or SHOW(DEFN,IMSID,LOCAL).
- The AreaName, LclStat, LModelName, LModelType, and LDefnType columns, which are returned on the QUERY DB SHOW(ALL) command, are not returned with SHOW(DEFN).
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition. There will be some performance overhead for SHOW(DEFN,IMSID) because a list of the IMS systems that have the resource defined is also to be obtained, along with the resource definition.

DEFNTYPE
Returns the definition type. This is how the resource was defined.

EXPORTNEEDED
Specifies that only the database resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the resources that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each resource name

specified on the command. If the specified resource does not need to be exported to the repository, an N is returned in the ExportNeeded column.

SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords and is valid only when IMS is enabled with the IMSRSC repository.

The QUERY DB SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources and descriptors that have not been exported to the IMSRSC repository, you can issue the EXPORT DEFN TARGET(REPO) OPTION(CHANGESONLY) command to write the resource and descriptor definitions to the repository so that they are available for the next IMS cold start.

GLOBAL
For output fields that have both local and global values, this option returns only the global values. If used with another SHOW keyword to request a specific output field, this option requests that only the global value of the specified output field is returned. IMS retrieves global status information from RM. Global output is returned only by the command master IMS.

The command master IMS returns global status information from the RM resource structure on a separate line, even if the database is not defined at the command master.

If the database name is a HALDB master, global information is returned only for the master database, not for the HALDB partitions associated with the master. If the database name specified is a DEDB, global information is returned only for the DEDB, not the DEDB areas associated to the DEDB.

If SHOW(GLOBAL) is specified without SHOW(DEFN), the repository information is not returned. If GSTSDB=Y is set in IMS, SHOW(GLOBAL) returns global status information from the RM resource structure.

If SHOW(GLOBAL,DEFN) is specified, the global resource definitions from the repository are returned by the command master IMS. SHOW(GLOBAL,DEFN) is valid only when the command master IMS is enabled to use the repository.

IMSID
Returns the IMSIDs of the IMS systems whose resource lists contain the resource name specified.

SHOW(IMSID) is processed only by the command master IMS and is valid only when command master IMS is enabled to use the repository.

If SHOW(IMSID) is specified with the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

If SHOW(IMSID) is specified without the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

SHOW(IMSID) cannot be specified with any other SHOW filters other than DEFN and GLOBAL. If SHOW(IMSID,GLOBAL) is specified, GLOBAL is ignored; that is, SHOW(IMSID,GLOBAL) is treated as SHOW(IMSID). SHOW(DEFN,IMSID,LOCAL) is treated as SHOW(DEFN,LOCAL).

LOCAL

For output fields that have both local and global values, this option returns only the local values. If used with another SHOW keyword to request a specific output field, this option requests that only the local value of the specified output field is returned. Local output is returned by each IMS that processes the command.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

MODEL

The model name and model type used to create this resource. The model name and model type are blanks for databases generated in the MODBLKS data set. The CREATE command specified without the LIKE keyword creates a resource using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSDB1 or user-defined. The CREATE command specified with the LIKE keyword creates a resource using a model. The resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

PGM

The names of the programs that reference the specified database.

The QRY DB SHOW(PGM) command will not show the area names if the DB is a DEDB or the partition names if the DB is a HALDB master.

The QRY DB SHOW(PGM) command will not show dynamic (DOPT) PSBs and database names for which the DOPT PSB has intent.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(PGM) individually.

RESIDENT

Returns the local runtime value for the resident option. The value returned is always RESIDENT(Y) for Fast Path DEDBs. The resident option definition is also shown, if it is different from the runtime value. The RESIDENT(Y) option takes effect at the next restart, unless the database was created or updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart.

SNDX

Returns the names of the associated Fast Path secondary index databases for a DEDB database.

Restriction: You cannot specify this filter with other SHOW filters. You must specify SHOW(SNDX) individually.

STATUS

Returns the local and global database status.

If global status is maintained, the command master IMS returns global information from the RM resource structure on a separate line. The global status information is returned even if the database is not defined at the command master. For a description of the possible database status returned, see the STATUS keyword in the Output fields table under "Output fields" on page 63.

TIMESTAMP

The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS) time, and import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

VERSION

Returns the version number of the database that is currently active in the online IMS system. The active version number of a database is stored in the database control blocks that are loaded by the online IMS system. The control blocks define the actual physical structure of the database to the online IMS system.

The possible version numbers range from 0 - 2147483647.

You cannot specify this filter with other SHOW filters; you must specify SHOW(VERSION) individually.

SHOW(VERSION) is valid only for the following database access types:
- HDAM
- HIDAM
- PHDAM
- PHIDAM
- DEDB

If SHOW(VERSION) is specified for an unsupported database access type, completion code BD is returned in the CC column of the output. This completion code indicates that the query is invalid for the access type of the database.

If SHOW(VERSION) is specified for a HALDB master, the output includes response lines for the HALDB master name and for each of its partitions. Each partition of the HALDB inherits the version number from the HALDB master, so each response line displays the same version number.

If SHOW(VERSION) is specified for a HALDB partition, the output lists a response line for just that partition. The version number that is displayed is that of the HALDB master. If the partition is disconnected from the HALDB master, such as might happen when the /DBR command is in effect on the database, completion code 10 is returned in the CC column of the output to indicate that no resource was found.

If SHOW(VERSION) is specified for a DEDB, the output lists a response line for only the DEDB. Area information is not listed in the output response.

WORK

Work is in progress for the database that is specified on the NAME parameter and its associated resources. The QRY DB SHOW(WORK) command can be issued before a DELETE, IMPORT or UPDATE command to check for any work in progress for the specified database and any of its associated resources. Any work in progress might cause the subsequent DELETE, IMPORT or UPDATE command to fail. The QRY DB SHOW(WORK) command returns the status for the work in progress for the database that is specified on the NAME parameter.

Specifying SHOW(WORK) with NAME(*) might take a long time to process.

The QRY DB SHOW(WORK) command for a HALDB master database will show the master name if the master has work in progress. The HALDB partition names are shown only if the partitions have work in progress.

Notes:

1. You cannot specify this filter with other SHOW filters; you must specify SHOW(WORK) individually.
2. The QRY DB SHOW(WORK) command is not valid on an XRF alternate.

STATUS()

Selects databases for display that match the NAME parameter and possess at least one of the specified database statuses. This selection allows for additional filtering by database status.

The output returned when the STATUS filter is specified includes the status of the database that caused the database name to be displayed even if SHOW(STATUS) is not specified.

ALLOCF

Sets the STATUS() filter to return information about databases that have allocation failure.

ALLOCS

Sets the STATUS() filter to return information about databases that are allocated successfully.

BACKOUT

Sets the STATUS() filter to return information about databases for which incomplete backout exists that prevents the use of the databases.

EEQE

Sets the STATUS() filter to return information about databases for which one or more extended error queue elements exist.

LOCK

Sets the STATUS() filter to return information about databases that are locked.

NOTINIT

Sets the STATUS() filter to return information about databases that are not initialized and therefore cannot be used.

NOTOPEN

Sets the STATUS() filter to return information about databases that are not open.

OFR

Sets the STATUS() filter to return information about databases that have online forward recovery in progress to bring them to the current tracking level.

OLR

The QUERY DB STATUS(OLR) command displays the status of all 'authorized' partitions with HALDB OLR in progress. The rate information is not returned. Instead, the rate information is returned on the QUERY OLREORG command output. A partition can be authorized by issuing a /START DB *partname* OPEN or UPDATE DB NAME(*partname*) OPTION(OPEN) command or by accessing the partition from an application. The SSYS record from a RECON listing can be used to determine whether a partition is authorized and to which subsystem.

OPEN

Sets the STATUS() filter to return information about databases that are open.

QUIESCED

Sets the STATUS() filter to return information about databases that are currently quiesced by a previous UPDATE DB START(QUIESCE) command.

QUIESCING

Sets the STATUS() filter to return information about databases that are undergoing quiesce by a previous UPDATE DB START(QUIESCE) command.

RECALL

Sets the STATUS() filter to return information about databases for which database recalls are in progress.

RECOV

Sets the STATUS() filter to return information about databases for which database recovery is in progress.

RNL

Sets the STATUS() filter to return information about DEDB databases for which randomizers are not loaded.

STOACC

Sets the STATUS() filter to return information about databases that are stopped for access locally or globally and are offline. The SHOW(STATUS) keyword on the QUERY DB command returns a status of STOACC for a DEDB. A status of STOACC indicates that the DEDB is stopped from further access because of a prior /DBR DB or UPDATE DB STOP(ACCESS) command.

STOSCHD

Sets the STATUS() filter to return information about databases that are stopped locally.

STOUPDS

Sets the STATUS() filter to return information about databases that are stopped for updates locally.

TYPE()

Selects databases for display that match the NAME parameter and specific TYPE filter. The supported TYPE filters are:

- DEDB
- DHISNDX (refers to Fast Path secondary index databases)
- DLI
- MSDB
- PART
- PHDAM
- PHIDAM
- PSINDEX

TYPE can be specified with the STATUS filter. If both TYPE and STATUS filters are specified, a response line is returned for each database that matches the NAME parameter and the TYPE and STATUS filters specified.

Usage notes

The command can be specified only through the OM API and can be processed only in DB/DC and DBCTL environments.

The QUERY DB command is valid on the XRF alternate as well as the RSR tracker.

The command syntax for QUERY DB is defined in XML and is available to automation programs that communicate with OM.

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB except for the /START DB command and the UPDATE DB START(ACCESS) command. A HALDB partition is not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

For HALDB databases, IMS tracks partition statuses and master database statuses separately. For example, a partition can be stopped, but its master database can be started. Alternatively, the partition can be started, but its master database can be stopped. Before opening, authorizing, or scheduling a partition, IMS always examines the status of the partition and the master database. If either the partition or the master database has a status that prevents the action, IMS does not perform the action.

Each partition has the access limitations of both itself and its master database. For example, if the master database has an access intent of read (READ) and one of its partitions has an access intent of update (UPD), the partition cannot be updated. Alternatively, if the master database has an access intent of update (UPD) and one of its partitions has an access intent of read (READ), the partition cannot be updated. Similar considerations apply to other statuses that affect access limitations, such as being stopped or locked.

Exception: If the HALDB master database has update access (UPD), the partitions can have an access intent of exclusive (EXCL), exceeding the access of the master.

Commands that are issued with a partition name affect only the status of the partition. Commands that are issued against the master database affect only the status of the master database. Therefore, a start of a master database does not update the status of its partitions. If the partitions are stopped, they remain stopped. When a HALDB partition is explicitly stopped, it must be explicitly started again. The type-1 commands with the keyword ALL, type-2 commands with NAME(*), and commands against a HALDB master do not change the STOPPED (shown as STOACC, STOSCHD, or STOUPDS on QUERY DB) and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is UPDATE DB STOP(ACCESS) on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. However, the stopped status is only set in the master database. If a QUERY DB command is issued, only the HALDB master displays a status of STOACC (each HALDB partition does not display STOACC unless it was itself stopped). If a UPDATE DB STOP(ACCESS) command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

If you want to display information about resource definitions, specify SHOW(DEFN). If you want to know which IMS systems have the resource defined and also know the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource defined, specify SHOW(IMSID).

Equivalent IMS type-1 commands

The following table shows variations of the QUERY DB command and the type-1 IMS commands that perform similar functions.

Table 18. Type-1 equivalents for the QUERY DB command

QUERY DB command	Similar IMS type-1 commands
QUERY DB	/DIS DB dbname1...dbnamen	ALL, /DIS STATUS DB

Output fields

The following table shows the QUERY DB output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 19. Output fields for QUERY DB command

Short label	Long label	Keyword	Scope	Meaning
ACC	Acc	ACCTYPE, GLOBAL	GBL	Global access type information from RM, which can be one of the following values: **BRWS** Database has global access type of BRWS. **EXCL** Database has global access type of EXCL. **READ** Database has global access type of READ. **UPD** Database has global access type of UPD.
AREA	AreaName	N/A	N/A	Area name. The area name is returned if there are one or more response lines for DEDB areas in the output.
CC	CC	N/A	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned. Refer to the return, reason, and completion codes for QUERY DB.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DB	DBName	DB	N/A	Database name.
DFNT	LDefnType	DEFNTYPE	N/A	Definition type, which can be one of the following values: **CREATE** Defined by a CREATE command. **DDLCRE** Defined by the DDL CREATE DATABASE command. **IMPORT** Defined by the IMPORT command. **IMS** Defined by IMS. **MODBLKS** Defined by system definition in the MODBLKS data set. **UPDATE** Defined by system definition in the MODBLKS data set, but changed into a dynamic resource by the UPDATE command.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the database has been exported to the IMSRSC repository. The value can be Y or N.
IMSID	IMSid	IMSID	GBL	Returns from the repository the IMSIDs that have resource defined.

Table 19. Output fields for QUERY DB command (continued)

Short label	Long label	Keyword	Scope	Meaning
LACC	LAcc	ACCTYPE	LCL	Type of access to database or area, which can be one of the following values: **BRWS** The database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data since a sharing IMS subsystem could be updating the database. The database is opened for read-only processing. **EXCL** The named database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC. **READ** The database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. Access type READ differs from access type BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing. **UPD** The database is available for update as well as read processing in the IMS subsystem.
LRSDNT	LRsdnt	ALL, RESIDENT	LCL	Local runtime value of the resident option. Indicates whether the database DMB resides in local storage. **N** The DMB associated with the named database resource is not made resident in storage. If a database is defined as resident but encounters an error during IMS restart, N is set. The DMB is loaded at scheduling time. **Y** The DMB associated with the named database resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the DMB and initializes it. A resident database is accessed from local storage, which eliminates I/O to the ACBLIB.
LSTT	LclStat	STATUS	LCL	Local database status. All database status conditions that apply are returned. Database status can be one or more of the status conditions described in Table 20 on page 69.
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. The IMS identifier of the IMS that built the output. The IMS identifier is always returned.

Table 19. Output fields for QUERY DB command (continued)

Short label	Long label	Keyword	Scope	Meaning
MDLN	LModelName	MODEL	N/A	Model name. Name of the resource or descriptor that is used as a model to create this resource. DFSDSDB1 is the IMS descriptor name for databases.
MDLT	LModelType	MODEL	N/A	Model type, either RSC or DESC. RSC means that the resource was created by using another resource as a model. DESC means that the resource was created by using a descriptor as a model.
PART	PartName	N/A	N/A	HALDB partition name. The partition name is returned if there is one or more response lines for HALDB partitions in the output.
PGM	LPgmName	PGM	LCL	Program name, from the local IMS, that references the database.
RACC	Acc	DEFN	GBL	Access type information from the repository if SHOW(DEFN) is specified and the repository is enabled.
REPO	Repo	DEFN	GBL	Indicates whether the information about the line includes stored resource definitions. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
RRSDNT	Rsdnt	DEFN, RESIDENT	GBL	Resident value from the repository.
RSDNT	LDRsdnt	ALL, RESIDENT	LCL	Local deferred resident value that takes effect at the next IMS restart. A value of Y is shown if a database was defined as resident but could not be made resident at IMS restart time because no DMB existed for it in ACBLIB. This database can become resident during the next IMS restart only if there is a DMB for it in the ACBLIB.
RTMCR	TimeCreate	DEFN	GBL	Create time from the repository. This is the time the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN	GBL	Update time from the repository. This is the time the resource was last updated in the repository.
SNDX	SndxName	SNDX	LCL	Name of the Fast Path secondary index database for a DEDB database.

Table 19. Output fields for QUERY DB command (continued)

Short label	Long label	Keyword	Scope	Meaning
STT	Status	STATUS, GLOBAL	GBL	Global status information from RM, which can be one of the following values: **ALLOC** Database has a global status of allocated. **LOCKED** Database is a global status of locked. **OPEN** Database has a global status of open. **STA** Database has a global status of started. **STOACC** Database has a global status of stopped for access. **STOSCHD** Database has a global status of stopped. **STOUPDS** Database has a global status of stopped for updates.
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the resource was last accessed. The last access time is retained across warm start, emergency restart, export, and import. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. This access time stamp is obtained from the local IMS. For a database resource, the following actions update the last access time: • Database is accessed by an application program. • CREATE command references the resource as the model. For HALDB and DEDB databases, the last access time stamp is returned in the HALDB partition entry or the AREA entry. The last access time stamp is left blank in the HALDB master and the DEDB entries.
TMCR	LTimeCreate	TIMESTAMP	LCL	The time the resource was created. This is the result of a CREATE DB command, IMPORT command that creates the database, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT. This create time stamp is obtained from the local IMS.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the resource was last imported, if applicable. The import time is retained across warm start and emergency restart. This import time stamp is obtained from the local IMS.

Table 19. Output fields for QUERY DB command (continued)

Short label	Long label	Keyword	Scope	Meaning
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated as a result of the UPDATE DB, a type-1 command, or the IMPORT command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
TYP	Type	N/A	N/A	The type of the database. • AREA - indicates the response line is for a DEDB area • blank - if database status is NOTINIT • DEDB - indicates the database is a DEDB • DHISNDX - indicates the database is a DEDB HISAM or SHISAM secondary index database • DL/I - indicates the database is a full function non-partitioned database • MSNR - indicates the database is an MSDB non-related database • MSRD - indicates the database is an MSDB-related dynamic database • MSRF - indicates the database is an MSDB-related fixed database • PART - indicates the database is a HALDB partition • PHDAM - indicates the database is the master of a partitioned HDAM database • PHIDAM - indicates the database is the master of a Partitioned HIDAM database • PSINDEX - indicates the database is the master of a partitioned secondary index database
VER	Version	VERSION	LCL	Version number of the database that is currently active on the IMS system.

WRK	Work	WORK		Work is in progress for the database or one of its associated resources. Work in progress can be one of the following values:
				
				AREA OPEN
				    Area associated with the FP DEDB is open.
				
				DB STOP ACCESS IN PROGRESS
				    A /DBRECOVERY or UPDATE DB STOP(ACCESS) command to stop database access is in progress for a database. This takes the database offline.
				
				DB STOP UPDATES IN PROGRESS
				    A /DBDUMP or UPDATE DB STOP(UPDATES) command to stop database updates is in progress for the database.
				
				IN USE
				    Database is in use.
				
				RECOVER CMD ACTIVE
				    A /RECOVER START command is in progress to recover one or more databases with the database recovery services.

QUERY DB status

The following table lists local area status conditions that can be returned when SHOW(STATUS) is specified.

Table 20. Local status conditions for QUERY DB

Status	Meaning
ALLOCF	Database has an allocation failure.
ALLOCS	Database is allocated successfully.
BACKOUT	Incomplete backout exists for the database that prevents the use of the database.
EEQE	One or more extended error queue elements exist for the database.
IC	Area image copy is active.
LOCK	Database is locked locally.
NONE	Database has no global status.
NOTINIT-*xx-reason*	Database is not initialized and therefore cannot be used. NOTINIT is displayed in the format NOTINIT-*xx-reason*, where *xx* is the reason code that identifies the unique location in one module where this reason code is set. NOTINIT-00 indicates that the reason is unknown. Action: 1.
	DFSDDIR MACRO defines each reason code that might be set in the database bad reason code (field DDIRBADR) and identifies the module that sets it. *reason* explains the reason code *xx* in abbreviated text format up to 13 characters.
NOTOPEN	Area or database is not open.
OFR	Area or database has online forward recovery in progress to bring it up to current tracking level.
OLR	Database partition has an online reorganization in progress.
	The QUERY DB STATUS(OLR) command displays the status of all databases with HALDB OLR in progress. Rate information is not returned. Instead, the rate information is returned on the QUERY OLREORG command output.
OPEN	Area or database is open. Not maintained in the RM resource structure.
PRELOAD	Area is defined to be preloaded.
PREOPEN	Area is defined to be preopened.
QUIESCED	Area or database is currently quiesced by a previous UPDATE DB START(QUIESCE) or UPDATE AREA START(QUIESCE) command.
QUIESCING	Area or database is currently undergoing quiesce by a previous UPDATE DB START(QUIESCE) or UPDATE AREA START(QUIESCE) command.
RECALL	Area or database recall is in progress.
RECOVINP	Area or database recovery is in progress.
RECOVNEEDED	Area needs recovery.
RNL	Randomizer is not loaded for the DEDB database.

Table 20. Local status conditions for QUERY DB (continued)

Status	Meaning
STOACC	Database is stopped for access locally and is offline. The SHOW(STATUS) keyword on the QUERY DB command returns a status of STOACC for a DEDB. A status of STOACC indicates that the DEDB is stopped from further access because of a prior /DBR DB or UPDATE DB STOP(ACCESS) command.
STOPPED	Area is stopped locally or globally.
STOSCHD	Database is stopped locally.
STOUPDS	Database is stopped for updates locally.
VSO	Area is a VSO area.

Table 21. Reason information for NOTINIT-xx-reason status

Reason	Meaning
ALIAS	Alias name error.
BLDL	BLDL miscellaneous error trying to build ACBLIB directory.
DBINIT	Database initialization failed.
DFLTCATALIASN	Default catalog alias not used.
DMBINCOMPTBL	DMB incompatibility.
DMBLEVEL	The IMS release level at which this DMB is generated using ACBGEN does not match the IMS release level of this IMS. Perform a DBDGEN, PSBGEN, ACBGEN, and ACBLIB online change as needed to generate this DMB at the correct IMS release level. Action: 4.
DMBNAME	DMB name missing.
DMBPOOL	DMB pool shortage.
DMBPOOLDELETE	DMB delete from DMB pool failed.
DUPLICATEAREA	Duplicate area name found in this DEDB. Action: 4.
EOD	EOD marker found before DMB.
FPDB	The database is a Fast Path MSDB or DEDB that is defined in a non-Fast Path (FP=N) system. The database cannot be used.
FPRESTART	A Fast Path error occurred during restart. Action: 1.
LOADCOMPRESS	An error occurred loading the compression routine.
LOADRANDOMIZE	An error occurred loading the randomizer routine.
MAXDBEXCEEDED	Database exceeded maximum database limit of 32767. The database cannot be used. The database can be recovered at IMS cold start from RDDS auto import if the total number of databases in a local system is less than 32767.
MOLCCOMMIT	Member OLC COMMIT MEMBER failed.
MSDBCHANGE	MSDB added or changed by online change, which is not allowed. Action: 4.
MSDBLEVEL	MSDB level of MSDB in ACBLIB is incorrect. Action: 4.
NODB	No database DDIR control block exists. Action: 5.
NODMB	No database DMB exists in ACBLIB. For a Fast Path database, this could be an MSDB or a DEDB. Action: 2.
NOMSDB	No MSDBs defined in system.

Table 21. Reason information for NOTINIT-xx-reason status (continued)

Reason	Meaning
NOSHRINDXDDIR	No shared index database DDIR control block. Action: 5.
NOSHRINDXDMB	No DMB in ACBLIB for shared index. Action: 2.
NOTDMB	Not a DMB. A PSB by the same name as the database is defined in ACBLIB instead of a DMB. If this resource should be a program, create the program with a CREATE PGM command. If this resource should be a database, perform a DBDGEN, PSBGEN, ACBGEN, and ACBLIB online change to define this resource as a DMB instead of a PSB. Action: 4.
PARTBUILD	Partition build failed.
PSBINCOMPTBL	PSB incompatibility.
REPOCHGLIST	The database name exists in the change list in the IMSRSC repository for this IMS, and the change list is being processed or the change list processing failed because of an error. This database cannot be used, scheduled, updated, exported, or referenced as a model until it is successfully imported so that the runtime database definition and the stored definition in the repository are synchronized. Action: 6.
RSCNEEDOLC	A Fast Path DEDB database is created with the IMPORT DEFN command. The DEDB database cannot be brought online for use with the IMPORT DEFN command. Perform ACBLIB online change to bring Fast Path DEDB database online. Action 2.
RSCNEEDSTADB	A full-function HALDB is created with the IMPORT DEFN command. HALDB master is created. To create HALDB partition databases, issue either the /START DB command or the UPDATE DB command on the HALDB master database.
SEGMENTNUM	Segment number error. ACBGEN error, more than 127 segments. Action: 4.
WRONGDMB	Wrong DMB.
WRONGPSB	Wrong PSB.

Note: Actions that can be taken to initialize the database are:
1. Call IBM.
2. Perform ACBLIB online change to add the DMB to ACBLIB.
3. Perform ACBLIB online change to add the PSB to ACBLIB.
4. Perform ACBLIB online change to correct PSBs or DMBs.
5. Perform MODBLKS online change or issue the CREATE DB command to create the database.
6. If IMS change list processing failed and message DFS4411E was issued, correct the problem and take one of the following actions:
   - Shut down IMS and restart it in order to try processing the change list again.
   - Issue the IMPORT DEFN SOURCE(REPO) NAME(*dbname*) TYPE(DB) command to import the database, where *dbname* is the database name.

How the SHOW keyword on QUERY DB determines the type of output

Some examples of how the SHOW keyword determines the type of output returned on the QUERY DB command are provided in the following table.

Table 22. How the SHOW keyword on QUERY DB determines the type of output

Form of SHOW keyword used	Type of output returned
SHOW(LOCAL)	Only those output fields that are local to an IMS system. SHOW(ALL,LOCAL) provides the same output.
SHOW(GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM. SHOW(ALL,GLOBAL) provides the same output.
SHOW(ALL)	All of the output fields for those fields that have both local and global data. Both values are returned in the output.
SHOW(STATUS,GLOBAL)	Only global STATUS values.
SHOW(STATUS,LOCAL)	Only local STATUS values.
SHOW(STATUS)	Both local and global STATUS values.
SHOW(ALL,GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM. SHOW(GLOBAL) provides the same output.
SHOW(ALL,LOCAL)	Only those output fields that are local to an IMS system. SHOW(LOCAL) provides the same output.
SHOW(DEFN)	The runtime definitions from IMS and the stored definitions from the repository are returned.
SHOW(DEFN,IMSID)	The runtime definitions from IMS and the stored definitions from the repository are returned. A response line is returned along with the definitional attribute for each IMS that has the resource defined in the repository.
SHOW(DEFN,LOCAL)	The runtime definitions from IMS are returned.
SHOW(DEFN,GLOBAL)	The stored definitions from the repository are returned.
SHOW(DEFN,IMSID,LOCAL)	The runtime definitions from IMS are returned. SHOW(DEFN,IMSID,LOCAL) is the same as SHOW(DEFN,LOCAL).
SHOW(DEFN,IMSID,GLOBAL)	The stored definitions from the repository are returned. A response line is returned along with the definitional attribute for each IMS that has the resource defined in the repository.
SHOW(EXPORTNEEDED)	Only those runtime resources whose definitions has not been exported to the repository.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY DB command. The OM return and reason codes that might be returned as a result of the QUERY DB command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 23. Return and reason codes for the QUERY DB command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY DB command completed successfully.
X'00000004'	X'00001010'	No resources were found to be returned. The resource names specified might be invalid, or: • There were no resources that match the filter specified. • There were no resources that had work to display for the SHOW(WORK) specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'00000008'	X'00002014'	The QUERY DB command is not processed because an invalid character is found in the database name parameter.
X'00000008'	X'00002040'	More than one filter or keyword value is specified on the QUERY DB command. Either more than one keyword or an invalid combination of filters was specified. Check the input command and reenter the correct combinations.
X'0000000C'	X'00003000'	The QUERY DB command was successful for some resources but failed for others. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY DB command are listed in the QUERY DB completion code table.
X'0000000C'	X'00003004'	The QUERY DB command was not successful for any of the resource names specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY DB command are listed in the QUERY DB completion code table.
X'00000010'	X'00004004'	No CQS address space.
X'00000010'	X'00004018'	No resource structure. Or, the resource structure is not available.
X'00000010'	X'00004100'	Resource structure is full.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx is defined with MODBLKS=OLC, or MODBLKS is not defined).
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled to use the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.

Table 23. Return and reason codes for the QUERY DB command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Repository access was denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server (RS) is being shut down.
X'00000010'	X'0000450B'	RS is not available.
X'00000010'	X'0000450C'	RS is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields related to the IMSRSC repository.
X'00000014'	X'00005004'	The QUERY DB command processing terminated as a DFSOCMD response buffer could not be obtained.
X'00000014'	X'0000501C'	IMODULE GETMAIN error.
X'00000014'	X'00005100'	RM request error.
X'00000014'	X'00005104'	CQS error.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY DB command is rejected because the resource list is unusable. Contact IBM Software Support. Send in the output from the command in error, and a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the IMSRSC repository RID and RMD data sets.
X'00000014'	X'00005FFF'	The QUERY DB command processing terminated because of an internal error.

The following table includes an explanation of the completion codes. Errors unique to the processing of QUERY DB command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 24. Completion codes for the QUERY DB command

Completion code	Completion code text	Meaning
0		The QUERY DB command completed successfully for the resource.
10	RESOURCE NOT FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. Confirm the correct spelling of the resource name specified on the command.

Table 24. Completion codes for the QUERY DB command (continued)

Completion code	Completion code text	Meaning
193	NOT A DEDB	The SHOW(SNDX) keyword was specified, but the database resource is not a DEDB.
194	NO SECONDARY INDEX DEFINED	The SHOW(SNDX) keyword was specified, but the database resource has no Fast Path secondary index database defined.
195	INVALID DB ACCESS TYPE	Database versioning for this database access type is not supported.

Examples

The following are examples of the QUERY DB command:

Example 1 for QUERY DB command

TSO SPOC input:

```
QUERY DB NAME(BANKATMS,DEDBJN21,DBHDOK01) SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
DBName    AreaName MbrName   CC TYPE     LAcc LRsdnt LclStat
BANKATMS           IMS1      0           EXCL N      NOTINIT-1E-NODMB,NOTOPEN
DBHDOK01           IMS1      0 DL/I      UPD  N      NOTOPEN
DEDBJN21           IMS1      0 DEDB      UPD  Y      NOTOPEN
DEDBJN21 DB21AR0  IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR1  IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR10 IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR11 IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR2  IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR3  IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR4  IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR5  IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR6  IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR7  IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR8  IMS1       0 AREA             NOTOPEN
DEDBJN21 DB21AR9  IMS1       0 AREA             NOTOPEN

(scrolled right to screen 2)
DBName    AreaName MbrName   LModelName LModelType LTimeCreate
BANKATMS           IMS1                            2011.181 10:22:15.10
DBHDOK01           IMS1                            2011.181 10:22:15.10
DEDBJN21           IMS1                            2011.181 10:22:15.10
DEDBJN21 DB21AR0  IMS1
DEDBJN21 DB21AR1  IMS1
DEDBJN21 DB21AR10 IMS1
DEDBJN21 DB21AR11 IMS1
DEDBJN21 DB21AR2  IMS1
DEDBJN21 DB21AR3  IMS1
DEDBJN21 DB21AR4  IMS1
DEDBJN21 DB21AR5  IMS1
DEDBJN21 DB21AR6  IMS1
DEDBJN21 DB21AR7  IMS1
DEDBJN21 DB21AR8  IMS1
DEDBJN21 DB21AR9  IMS1

(scrolled right to screen 3)
DBName    AreaName MbrName  LTimeUpdate LTimeAccess         LTimeImport
BANKATMS           IMS1
DBHDOK01           IMS1
DEDBJN21           IMS1
DEDBJN21 DB21AR0  IMS1
DEDBJN21 DB21AR1  IMS1
DEDBJN21 DB21AR10 IMS1
```

```
DEDBJN21 DB21AR11 IMS1
DEDBJN21 DB21AR2  IMS1
DEDBJN21 DB21AR3  IMS1
DEDBJN21 DB21AR4  IMS1
DEDBJN21 DB21AR5  IMS1
DEDBJN21 DB21AR6  IMS1
DEDBJN21 DB21AR7  IMS1
DEDBJN21 DB21AR8  IMS1
DEDBJN21 DB21AR9  IMS1

(scrolled right to screen 4)
DBName    AreaName MbrName  LDefnType
BANKATMS           IMS1     MODBLKS
DBHDOK01           IMS1     MODBLKS
DEDBJN21           IMS1     MODBLKS
DEDBJN21 DB21AR0  IMS1
DEDBJN21 DB21AR1  IMS1
DEDBJN21 DB21AR10 IMS1
DEDBJN21 DB21AR11 IMS1
DEDBJN21 DB21AR2  IMS1
DEDBJN21 DB21AR3  IMS1
DEDBJN21 DB21AR4  IMS1
DEDBJN21 DB21AR5  IMS1
DEDBJN21 DB21AR6  IMS1
DEDBJN21 DB21AR7  IMS1
DEDBJN21 DB21AR8  IMS1
DEDBJN21 DB21AR9  IMS1
```

OM API input:

```
CMD(QUERY DB NAME(BANKATMS,DEDBJN21,DBHDOK01) SHOW(ALL)  )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.181 18:44:59.208162</statime>
<stotime>2011.181 18:44:59.209123</stotime>
<staseq>C80029508EDE234E</staseq>
<stoseq>C80029508F1A3B8E</stoseq>
<rqsttkn1>USRT005 10114459</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DB             </kwd>
<input>QUERY DB NAME(BANKATMS,DEDBJN21,DBHDOK01) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RSDNT" llbl="LDRsdnt" scope="LCL" sort="n" key="0"
```

```
scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LRSDNT" llbl="LRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MDLN" llbl="LModelName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MDLT" llbl="LModelType" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(BANKATMS) MBR(IMS1   ) CC(   0) TYP(      ) LACC(EXCL)
 LSTT(NOTINIT-1E-NODMB,NOTOPEN) DFNT(MODBLKS) LRSDNT(N) TMCR(2011.181
 10:22:15.10) TMUP(                ) TMIM(              )
 TMAC(                ) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1   ) CC(   0) TYP(DEDB  ) LACC(UPD)
 LSTT(NOTOPEN) DFNT(MODBLKS) LRSDNT(Y) TMCR(2011.181 10:22:15.10) TMUP(
                 ) TMIM(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR0 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR1 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR2 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR3 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR4 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR5 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR6 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR7 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR8 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR9 ) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR10) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DEDBJN21) AREA(DB21AR11) MBR(IMS1   ) CC(   0) TYP(AREA   )
 LSTT(NOTOPEN) TMAC(              ) </rsp>
<rsp>DB(DBHDOK01) MBR(IMS1   ) CC(   0) TYP(DL/I  ) LACC(UPD)
 LSTT(NOTOPEN) DFNT(MODBLKS) LRSDNT(N) TMCR(2011.181 10:22:15.10) TMUP(
                 ) TMIM(              ) TMAC(
   ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY DB command is specified with the SHOW keyword to display the resident attribute and the database status. All definition and status information is returned for databases BANKATMS, DEDBJN21, and DBHDOK01 from IMS1. All of the database output fields do not fit on one screen, so you must scroll to the right for additional output fields. The database name, area name, and member name that built the line of output are displayed on every screen. Database BANKATMS has a status of NOTINIT-1E-NODMB, which means there is no DMB

in ACBLIB for BANKATMS and BANKATMS cannot be used. DEDBJN21 is a
DEDB and the area information is also returned, along with the DEDB information.
No model information is returned because the database was loaded from
MODBLKS and not created from a model. Time stamps for the DEDB areas are not
maintained and so they are not returned.

Example 2 for QUERY DB command

TSO SPOC input:

```
QUERY DB NAME(BE3PARTS,CUSTDB) SHOW(PGM)
```

TSO SPOC output:

```
DBName    MbrName   CC TYPE     LPgmName
BE3PARTS IMS1        0 DL/I     PE4CODEL
BE3PARTS IMS1        0 DL/I     PE4COINQ
BE3PARTS IMS1        0 DL/I     PE4CORDR
BE3PARTS IMS1        0 DL/I     PE4CPINV
BE3PARTS IMS1        0 DL/I     PE4CPPUR
CUSTDB   IMS1        0 DEDB     PSBBA
CUSTDB   IMS1        0 DEDB     PSBNO
CUSTDB   IMS1        0 DEDB     PSBOS
CUSTDB   IMS1        0 DEDB     PSBPA
```

OM API input:

```
CMD(QUERY DB NAME(BE3PARTS,CUSTDB) SHOW(PGM))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.181 18:53:06.988049</statime>
<stotime>2011.181 18:53:06.989553</stotime>
<staseq>C8002B21BDC11F98</staseq>
<stoseq>C8002B21BE1F14D8</stoseq>
<rqsttkn1>USRT005 10115306</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DB             </kwd>
<input>QUERY DB NAME(BE3PARTS,CUSTDB) SHOW(PGM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="PGM" llbl="LPgmName" scope="LCL" sort="n" key="0"
```

```
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) TYP(DL/I    ) PGM(PE4CODEL)
</rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) TYP(DL/I    ) PGM(PE4COINQ)
</rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) TYP(DL/I    ) PGM(PE4CORDR)
</rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) TYP(DL/I    ) PGM(PE4CPINV)
</rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) TYP(DL/I    ) PGM(PE4CPPUR)
</rsp>
<rsp>DB(CUSTDB  ) MBR(IMS1    ) CC(   0) TYP(DEDB    ) PGM(PSBBA  )
</rsp>
<rsp>DB(CUSTDB  ) MBR(IMS1    ) CC(   0) TYP(DEDB    ) PGM(PSBNO  )
</rsp>
<rsp>DB(CUSTDB  ) MBR(IMS1    ) CC(   0) TYP(DEDB    ) PGM(PSBOS  )
</rsp>
<rsp>DB(CUSTDB  ) MBR(IMS1    ) CC(   0) TYP(DEDB    ) PGM(PSBPA  )
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: Database BE3PARTS has programs BE3PARTS, PE4COINQ, and
PE4CORDR referring to it. Database CUSTDB has programs CUSTDB, PSBOS, and
PSBPA referring to it.

Example 3 for QUERY DB command

TSO SPOC input:

```
QRY DB NAME(DEDBRN01,DX41M401,DB000001) SHOW(RESIDENT,STATUS)
```

TSO SPOC output:

```
DBName     AreaName MbrName    CC TYPE      LDRsdnt LRsdnt LclStat
DB000001            IMS1       0            Y       N      NOTINIT-34-NODMB,NOTOPEN
DEDBRN01            IMS1       0 DEDB               Y      NOTOPEN
DEDBRN01 DEDB01D1 IMS1         0 AREA                      NOTOPEN
DX41M401            IMS1       0 DL/I               N      NOTOPEN
```

OM API input:

```
CMD(QRY DB NAME(DEDBRN01,DX41M401,DB000001) SHOW(RESIDENT,STATUS)      )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.181 18:59:00.325804</statime>
<stotime>2011.181 18:59:00.326647</stotime>
<staseq>C8002C72B5DAC610</staseq>
<stoseq>C8002C72B60F7B50</stoseq>
<rqsttkn1>USRT005 10115900</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DB             </kwd>
<input>QRY DB NAME(DEDBRN01,DX41M401,DB000001) SHOW(RESIDENT,STATUS)
```

```
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RSDNT" llbl="LDRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LRSDNT" llbl="LRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(DEDBRN01) MBR(IMS1    ) CC(    0) TYP(DEDB     ) LSTT(NOTOPEN)
 LRSDNT(Y) </rsp>
<rsp>DB(DEDBRN01) AREA(DEDB01D1) MBR(IMS1    ) CC(    0) TYP(AREA     )
 LSTT(NOTOPEN) </rsp>
<rsp>DB(DX41M401) MBR(IMS1    ) CC(    0) TYP(DL/I    ) LSTT(NOTOPEN)
 LRSDNT(N) </rsp>
<rsp>DB(DB000001) MBR(IMS1    ) CC(    0) TYP(         )
 LSTT(NOTINIT-34-NODMB,NOTOPEN) RSDNT(Y) LRSDNT(N) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY DB command is specified with SHOW keywords to
display the resident attribute and the database status for a few different types of
databases. DEDBRN01 is a DEDB. DEDBs are always resident and defined with
RESIDENT(Y). DX41M401 is a DLI database. DX41M401 is defined as Resident(N)
and its local runtime resident value is RESIDENT(N). Because Rsdnt and LRsdnt
values of database DX41M401 are the same, only LRsdnt value is displayed.
DB000001 is a new database created by a CREATE DB command. DB000001 shows
a status of NOTINIT-34-NODMB, which means it has no DMB defined in ACBLIB
and cannot be used. Since there is no DMB, the database type is shown as blanks,
because IMS does not know what type of database DB000001 is intended to be.
DB000001 is defined as RESIDENT(Y), but because the resident option will not take
effect until the next IMS restart, the local resident value is N. If all of the databases
being displayed have a LRsdnt (Local Resident) value that is the same as the Rsdnt
(Definitional Resident) value, the QRY DB output displays the LRsdnt header and
value, but not the Rsdnt header and value. Since database DB000001 is defined
with RESIDENT(Y) but its local runtime resident value is N, both Rsdnt and
LRsdnt columns are displayed in the QUERY DB output.

Example 4 for QUERY DB command

TSO SPOC input:
```
QRY DB NAME(*) SHOW(STATUS) STATUS(QUIESCING)
```

TSO SPOC output:

```
DBName      MbrName     CC TYPE      LclStat
DBXYZ       IM03         0 DLI       ALLOCS,OPEN,QUIESCING
KCIRDBAZ    IM03         0 DLI       ALLOCS,OPEN,QUIESCING
GONLXYZ     IM03         0 DLI       ALLOCS,OPEN,QUIESCING
```

Explanation: This example is a query of the databases that have a status of
QUIESCING.

Example 5 for QUERY DB command

TSO SPOC input:
```
QUERY DB NAME(DEDBJ001) SHOW(ACCTYPE,STATUS)
```

TSO SPOC output:
```
DBName     AreaName MbrName     CC TYPE      LAcc LclStat
DEDBJ001            IMS1         0 DEDB      UPD  OPEN
DEDBJ001 D0010001 IMS1           0 AREA      READ OPEN,PREOPEN
DEDBJ001 D0010002 IMS1           0 AREA      BRWS OPEN,PREOPEN
DEDBJ001 D0010003 IMS1           0 AREA           OPEN,PREOPEN
DEDBJ001 D0010004 IMS1           0 AREA      UPD  OPEN,PREOPEN
DEDBJ001 D0010005 IMS1           0 AREA           OPEN,PREOPEN
```

OM API input:
```
CMD(QUERY DB NAME(DEDBJ001) SHOW(ACCTYPE,STATUS))
```

OM API output:
```
<imsout>
 <ctl>
 <omname>OM1OM    </omname>
 <omvsn>1.5.0</omvsn>
 <xmlvsn>20   </xmlvsn>
 <statime>2011.188 23:38:28.734794</statime>
 <stotime>2011.188 23:38:28.735444</stotime>
<staseq>C80937F889B4AC94</staseq>
<stoseq>C80937F889DD4554</stoseq>
<rqsttkn1>USRT011 10163828</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1     </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>DB              </kwd>
<input>QRY DB NAME(DEDBJ001) SHOW(ACCTYPE,STATUS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
```

```
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(DEDBJ001) MBR(IMS1    ) CC(   0) TYP(DEDB    ) LACC(UPD)
 LSTT(OPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010001 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LACC(READ)LSTT(OPEN,PREOPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010002 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LACC(BRWS) LSTT(OPEN,PREOPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010003 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(OPEN,PREOPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010004 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LACC(UPD)LSTT(OPEN,PREOPEN) </rsp>
<rsp>DB(DEDBJ001) AREA(D0010005 ) MBR(IMS1    ) CC(   0) TYP(AREA    )
 LSTT(OPEN,PREOPEN) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: DEDB DEDBJ001 has database access of UPD (update). Area access for areas D0010001, D00010002, and D0010004 has been changed with the command; that is, area D0010001 has area access of READ (read), area D0010002 has area access of BRWS (read only), and area D0010004 has area access of UPD (update). Areas D0010003 and D0010005 inherit DEDBJ001 database access of UPD (update).

Example 6 for QUERY DB command

TSO SPOC input:
```
QUERY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1) SHOW(DEFN,ACCTYPE,RESIDENT)
```

TSO SPOC output:

(screen 1)

DBName	MbrName	CC	CCText	Repo	IMSid	TYPE
BANKATMS	IMS1	0		Y		
BANKATMS	IMS1	0			IMS1	
BANKATMS	IMS2	0			IMS2	
BANKATMS	IMS3	0			IMS3	
BE3PARTS	IMS1	0		Y		
BE3PARTS	IMS1	0		Y	IMS2	
BE3PARTS	IMS1	0		Y	IMS3	
BE3PARTS	IMS1	0			IMS1	DL/I
BE3PARTS	IMS2	0			IMS2	DL/I
BE3PARTS	IMS3	0			IMS3	DL/I
DEDBJN21	IMS1	0		Y		
DEDBJN21	IMS1	0			IMS1	DEDB
DEDBJN21	IMS2	0			IMS2	DEDB
DEDBJN21	IMS3	0			IMS3	DEDB
NEWDB1	IMS1	1D3	REPOSITORY MEMBER NOT FOUND	Y		
NEWDB1	IMS1	0			IMS1	
NEWDB1	IMS2	10	NO RESOURCES FOUND			
NEWDB1	IMS3	10	NO RESOURCES FOUND			

(scrolled right to screen 2)

DBName	MbrName	Repo	IMSid	TYPE	Acc	LAcc	Rsdnt	LDRsdnt	LRsdnt
BANKATMS	IMS1	Y			EXCL		N		
BANKATMS	IMS1		IMS1			EXCL			N
BANKATMS	IMS2		IMS2			EXCL			N
BANKATMS	IMS3		IMS3			EXCL			N
BE3PARTS	IMS1	Y			EXCL		N		
BE3PARTS	IMS1	Y	IMS2		UPD		N		
BE3PARTS	IMS1	Y	IMS3		EXCL		Y		
BE3PARTS	IMS1		IMS1	DL/I		EXCL			N

```
BE3PARTS IMS2            IMS2      DL/I          UPD              N
BE3PARTS IMS3            IMS3      DL/I          EXCL      Y      N
DEDBJN21 IMS1     Y                              UPD       Y
DEDBJN21 IMS1            IMS1      DEDB          UPD                   Y
DEDBJN21 IMS2            IMS2      DEDB          UPD                   Y
DEDBJN21 IMS3            IMS3      DEDB          UPD                   Y
NEWDB1   IMS1     Y
NEWDB1   IMS1            IMS1                    UPD              N
NEWDB1   IMS2
NEWDB1   IMS3
```

OM API input:

```
CMD(QUERY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1) SHOW(DEFN,ACCTYPE,RESIDENT))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.181 19:51:43.997358</statime>
<stotime>2011.181 19:51:44.021469</stotime>
<staseq>C800383BD29AEE9C</staseq>
<stoseq>C800383BD87DDE80</stoseq>
<rqsttkn1>USRT005 10125143</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
<mbr name="IMS3    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
<mbr name="IMS2    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DB            </kwd>
<input>QUERY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1)
               SHOW(DEFN,ACCTYPE,RESIDENT) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
```

```
                    <hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
                     scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
                     len="8" dtype="CHAR" align="left" skipb="no" />
                    <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
                     len="4" dtype="INT" align="right" skipb="no" />
                    <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
                     scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
                     len="1" dtype="CHAR" align="left" />
                    <hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
                     scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
                     len="8" dtype="CHAR" align="left" skipb="no" />
                    <hdr slbl="RACC" llbl="Acc" scope="GBL" sort="n" key="0" scroll="yes"
                     len="*" dtype="CHAR" align="left" skipb="no" />
                    <hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
                     len="*" dtype="CHAR" align="left" skipb="no" />
                    <hdr slbl="RRSDNT" llbl="Rsdnt" scope="GBL" sort="n" key="0"
                     scroll="yes" len="1" dtype="CHAR" align="left" />
                    <hdr slbl="RSDNT" llbl="LDRsdnt" scope="LCL" sort="n" key="0"
                     scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="LRSDNT" llbl="LRsdnt" scope="LCL" sort="n" key="0"
                     scroll="yes" len="1" dtype="CHAR" align="left" />
                    </cmdrsphdr>
                    <cmdrspdata>
                    <rsp>DB(BANKATMS) MBR(IMS1   ) CC(   0) TYP(        ) LACC(EXCL)
                     IMSID(IMS1    ) LRSDNT(N) </rsp>
                    <rsp>DB(DEDBJN21) MBR(IMS1   ) CC(   0) TYP(DEDB    ) LACC(UPD)
                     IMSID(IMS1    ) LRSDNT(Y) </rsp>
                    <rsp>DB(BE3PARTS) MBR(IMS1   ) CC(   0) TYP(DL/I    ) LACC(EXCL)
                     IMSID(IMS1    ) LRSDNT(N) </rsp>
                    <rsp>DB(NEWDB1  ) MBR(IMS1   ) CC(   0) TYP(        ) LACC(UPD)
                     IMSID(IMS1    ) LRSDNT(N) </rsp>
                    <rsp>DB(BANKATMS) MBR(IMS1   ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
                    </rsp>
                    <rsp>DB(DEDBJN21) MBR(IMS1   ) CC(   0) REPO(Y) RACC(UPD ) RRSDNT(Y)
                    </rsp>
                    <rsp>DB(BE3PARTS) MBR(IMS1   ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
                    </rsp>
                    <rsp>DB(BE3PARTS) MBR(IMS1   ) CC(   0) REPO(Y) IMSID(IMS2    )
                     RACC(UPD ) RRSDNT(N) </rsp>
                    <rsp>DB(BE3PARTS) MBR(IMS1   ) CC(   0) REPO(Y) IMSID(IMS3    )
                     RACC(EXCL) RRSDNT(Y) </rsp>
                    <rsp>DB(NEWDB1  ) MBR(IMS1   ) CC( 1D3) CCTXT(REPOSITORY MEMBER NOT
                     FOUND) REPO(Y) </rsp>
                    <rsp>DB(BANKATMS) MBR(IMS3   ) CC(   0) TYP(        ) LACC(EXCL)
                     IMSID(IMS3    ) LRSDNT(N) </rsp>
                    <rsp>DB(DEDBJN21) MBR(IMS3   ) CC(   0) TYP(DEDB    ) LACC(UPD)
                     IMSID(IMS3    ) LRSDNT(Y) </rsp>
                    <rsp>DB(BE3PARTS) MBR(IMS3   ) CC(   0) TYP(DL/I    ) LACC(EXCL)
                     IMSID(IMS3    ) RSDNT(Y) LRSDNT(N) </rsp>
                    <rsp>DB(NEWDB1  ) MBR(IMS3   ) CC(  10) CCTXT(NO RESOURCES FOUND)
                    </rsp>
                    <rsp>DB(BANKATMS) MBR(IMS2   ) CC(   0) TYP(        ) LACC(EXCL)
                     IMSID(IMS2    ) LRSDNT(N) </rsp>
                    <rsp>DB(DEDBJN21) MBR(IMS2   ) CC(   0) TYP(DEDB    ) LACC(UPD)
                     IMSID(IMS2    ) LRSDNT(Y) </rsp>
                    <rsp>DB(BE3PARTS) MBR(IMS2   ) CC(   0) TYP(DL/I    ) LACC(UPD)
                     IMSID(IMS2    ) LRSDNT(N) </rsp>
                    <rsp>DB(NEWDB1  ) MBR(IMS2   ) CC(  10) CCTXT(NO RESOURCES FOUND)
                    </rsp>
                    </cmdrspdata>
                    </imsout>
```

Explanation: This example shows the result of query for the runtime definitions and the definitions stored in the repository for databases BANKATMS, BE3PARTS, DEDBJN21, and NEWDB1, for an IMSplex that contains IMS1, IMS2, and IMS3. IMS1 is the command master IMS.

- In the TSO SPOC output, the response lines 1-4 show the repository and local information for DB BANKATMS.
  - Line 1 indicates that database BANKATMS is defined to the repository, the line was built by command master IMS1, and that the database has a global access type of EXCL and a global resident value of N. The Repo column shows a value of Y, which indicates the global values for database BANKATMS in the repository. The IMSid column is blank, which indicates that this is the global definition of BANKATMS for all IMS systems that it is defined to.
  - Lines 2, 3, and 4 indicate that database BANKATMS is defined locally to IMS1, IMS2, and IMS3 with local access type and resident runtime values that are the same as the repository values. The MbrName column is the IMSID of the IMS that built the response line. The IMSid column shows the IMSID of the IMS to which the resource or descriptor is defined. The command master IMS1 already built repository information for BANKATMS. The output does not indicate whether database BANKATMS is defined in the resource lists for IMS1, IMS2, or IMS3, because SHOW(IMSID) was not specified.
- The response lines 5-10 show the repository and local information for DB BE3PARTS.
  - Line 5 indicates that database BE3PARTS is defined to the repository, the line was built by command master IMS1, and that the database has a global access type of EXCL and a global resident value of N. The Repo column shows a value of Y, which indicates the global values for database BE3PARTS in the repository. The IMSid column is blank, which indicates that this is the global definition of BE3PARTS for all IMS systems that it is defined to and that do not have their own specific definitions.
  - Line 6 indicates that database BE3PARTS is defined to the repository with unique values for IMS2 (the access type is UPD instead of EXCL), different from the global values. When IMS2 next imports database BE3PARTS from the repository, it will get its unique access type value of UPD, but the global resident value of N. The resource list for IMS2 contains database BE3PARTS because it must, in order for its unique attributes, be stored in the repository.
  - Line 7 indicates that database BE3PARTS is defined to the repository with unique values for IMS3 (the resident value is Y instead of N), different from the global values. When IMS3 next imports database BE3PARTS from the repository, it will get its unique resident value of Y, but the global access type of EXCL. The resource list for IMS3 contains database BE3PARTS because it must, in order for its unique attributes, be stored in the repository.
  - Lines 8-10 indicate that database BE3PARTS is defined locally to IMS1, IMS2, and IMS3 with various local access type and resident runtime values. The output does not indicate whether database BE3PARTS is in the resource lists for IMS1, IMS2, or IMS3, because SHOW(IMSID) was not specified. These IMS systems do not access repository information; the command master IMS1 already accessed it. Runtime values for IMS1, IMS2, and IMS3 match the stored values in the repository, so their definitions are synchronized with the repository and no import is needed.
- The response lines 11-14 show the repository and local information for database DEDBJN21.

- Line 11 indicates that database DEDBJN21 is defined to the repository, the line was built by command master IMS1, and that the database has a global access type of UPD and a global resident value of Y. The Repo column shows a value of Y and indicates the global values for database DEDBJN21 in the repository.
- Lines 12-14 indicate that database DEDBJN21 is defined locally to IMS1, IMS2, and IMS3 with the local access type UPD and resident Y runtime values. The command master IMS1 already built repository information for DEDBJN21. Runtime values for IMS1, IMS2, and IMS3 match the stored values in the repository, so their definitions are synchronized with the repository, and no import is needed. The output does not indicate whether database DEDBJN21 is defined in the resource lists for IMS1, IMS2, or IMS3, because SHOW(IMSID) was not specified.
- The response lines 15-18 shows the information for DB NEWDB1.
  - Line 15 indicates that the database NEWDB1 is not defined to the repository.
  - Line 16 indicates that database NEWDB1 is defined locally to IMS1.
  - Line 17-18 indicate that the database NEWDB1 is not defined locally to IMS2 and IMS3.

Example 7 for QUERY DB command

TSO SPOC input:

```
QUERY DB NAME(BA*) SHOW(DEFN)
```

TSO SPOC output:

```
(screen 1)
DBName   MbrName   CC Repo IMSid    TYPE      Acc  LAcc Rsdnt LRsdnt
BANKATMS IMS1       0 Y                       EXCL      N
BANKATMS IMS1       0      IMS1                    EXCL      N
BANKATMS IMS2       0      IMS2                    EXCL      N
BANKFNCL IMS1       0 Y                       EXCL      N
BANKFNCL IMS1       0      IMS1                    EXCL      N
BANKFNCL IMS2       0      IMS2                    EXCL      N
BANKLDGR IMS1       0 Y                       EXCL      N
BANKLDGR IMS1       0      IMS1                    EXCL      N
BANKLDGR IMS2       0      IMS2                    EXCL      N
BANKTERM IMS1       0 Y                       EXCL      N
BANKTERM IMS1       0      IMS1                    EXCL      N
BANKTERM IMS2       0      IMS2                    EXCL      N
 (scrolled right to screen 2)
DBName   MbrName  Repo TimeCreate          LTimeCreate
BANKATMS IMS1      Y    2011.181 10:22:15.10
BANKATMS IMS1                               2011.181 10:22:15.10
BANKATMS IMS2                               2011.181 10:20:47.82
BANKFNCL IMS1      Y    2011.181 10:22:15.10
BANKFNCL IMS1                               2011.181 10:22:15.10
BANKFNCL IMS2                               2011.181 10:20:47.82
BANKLDGR IMS1      Y    2011.181 10:22:15.10
BANKLDGR IMS1                               2011.181 10:22:15.10
BANKLDGR IMS2                               2011.181 10:20:47.82
BANKTERM IMS1      Y    2011.181 10:22:15.10
BANKTERM IMS1                               2011.181 10:22:15.10
BANKTERM IMS2                               2011.181 10:20:47.82
(scrolled right to screen 3)
DBName   MbrName  Repo LTimeUpdate LTimeAccess        LTimeImport
BANKATMS IMS1      Y
BANKATMS IMS1
BANKATMS IMS2
BANKFNCL IMS1      Y
BANKFNCL IMS1
```

```
BANKFNCL IMS2
BANKLDGR IMS1     Y
BANKLDGR IMS1
BANKLDGR IMS2
BANKTERM IMS1     Y
BANKTERM IMS1
BANKTERM IMS2
```

OM API input:

```
CMD(QUERY DB NAME(BA*) SHOW(DEFN))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.181 20:08:12.066851</statime>
<stotime>2011.181 20:08:12.152198</stotime>
<staseq>C8003BEA1E823B0C</staseq>
<stoseq>C8003BEA335864C2</stoseq>
<rqsttkn1>USRT005 10130812</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DB             </kwd>
<input>QRY DB NAME(BA*) SHOW(DEFN) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RACC" llbl="Acc" scope="GBL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RRSDNT" llbl="Rsdnt" scope="GBL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="RSDNT" llbl="LDRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LRSDNT" llbl="LRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="RTMCR" llbl="TimeCreate" scope="GBL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="RTMUP" llbl="TimeUpdate" scope="GBL" sort="n" key="0"
```

```
  scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(BANKATMS) MBR(IMS1   ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS1   ) LRSDNT(N) TMCR(2011.181 10:22:15.10) TMUP(
        ) TMIM(                  ) TMAC(                    ) </rsp>
<rsp>DB(BANKFNCL) MBR(IMS1   ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS1   ) LRSDNT(N) TMCR(2011.181 10:22:15.10) TMUP(
        ) TMIM(                  ) TMAC(                    ) </rsp>
<rsp>DB(BANKLDGR) MBR(IMS1   ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS1   ) LRSDNT(N) TMCR(2011.181 10:22:15.10) TMUP(
        ) TMIM(                  ) TMAC(                    ) </rsp>
<rsp>DB(BANKTERM) MBR(IMS1   ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS1   ) LRSDNT(N) TMCR(2011.181 10:22:15.10) TMUP(
        ) TMIM(                  ) TMAC(                    ) </rsp>
<rsp>DB(BANKATMS) MBR(IMS1   ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
RTMCR(2011.181 10:22:15.10) </rsp>
<rsp>DB(BANKFNCL) MBR(IMS1   ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
 RTMCR(2011.181 10:22:15.10) </rsp>
<rsp>DB(BANKLDGR) MBR(IMS1   ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
RTMCR(2011.181 10:22:15.10) </rsp>
<rsp>DB(BANKTERM) MBR(IMS1   ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
 RTMCR(2011.181 10:22:15.10) </rsp>
<rsp>DB(BANKATMS) MBR(IMS2   ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS2   ) LRSDNT(N) TMCR(2011.181 10:20:47.82) TMUP(
        ) TMIM(                  ) TMAC(                    ) </rsp>
<rsp>DB(BANKFNCL) MBR(IMS2   ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS2   ) LRSDNT(N) TMCR(2011.181 10:20:47.82) TMUP(
        ) TMIM(                  ) TMAC(                    ) </rsp>
<rsp>DB(BANKLDGR) MBR(IMS2   ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS2   ) LRSDNT(N) TMCR(2011.181 10:20:47.82) TMUP(
        ) TMIM(                  ) TMAC(                    ) </rsp>
<rsp>DB(BANKTERM) MBR(IMS2   ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS2   ) LRSDNT(N) TMCR(2011.181 10:20:47.82) TMUP(
        ) TMIM(                  ) TMAC(                    ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: A line is returned for each resource that matches the wildcard name.
The resource definitions from each IMS that has the resource defined and the
global repository definition are returned. The repository information is returned by
the command master IMS. There are no IMS-specific sections in the repository for
each resource name that matches the wildcard name.

Example 8 for QUERY DB command

TSO SPOC input:
```
QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1)
SHOW(DEFN,IMSID,ACCTYPE,RESIDENT)
```

TSO SPOC output:

(screen 1)

DBName	MbrName	CC	CCText	Repo	IMSid	TYPE	Acc
BANKATMS	IMS1	0		Y			EXCL
BANKATMS	IMS1	0		Y	IMS1		EXCL
BANKATMS	IMS1	0		Y	IMS2		EXCL
BANKATMS	IMS1	0		Y	IMS3		EXCL
BANKATMS	IMS1	0		Y	IMS4		EXCL

```
BANKATMS IMS1      0                                    IMS1
BANKATMS IMS2      0                                    IMS2
BANKATMS IMS3      0                                    IMS3
BE3PARTS IMS1      0                          Y                      EXCL
BE3PARTS IMS1      0                          Y    IMS1             EXCL
BE3PARTS IMS1      0                          Y    IMS2             UPD
BE3PARTS IMS1      0                          Y    IMS3             READ
BE3PARTS IMS1      0                          Y    IMS4             EXCL
BE3PARTS IMS1      0                               IMS1   DL/I
BE3PARTS IMS2      0                               IMS2   DL/I
BE3PARTS IMS3      0                               IMS3   DL/I
DEDBJN21 IMS1      0                          Y                      UPD
DEDBJN21 IMS1      0                          Y    IMS1             UPD
DEDBJN21 IMS1      0                          Y    IMS2             UPD
DEDBJN21 IMS1      0                          Y    IMS3             UPD
DEDBJN21 IMS1      0                          Y    IMS4             UPD
DEDBJN21 IMS1      0                               IMS1   DEDB
DEDBJN21 IMS2      0                               IMS2   DEDB
DEDBJN21 IMS3      0                               IMS3   DEDB
NEWDB1   IMS1     1D3 REPOSITORY MEMBER NOT FOUND Y
NEWDB1   IMS1      0                               IMS1
NEWDB1   IMS2     10 NO RESOURCES FOUND
NEWDB1   IMS3     10 NO RESOURCES FOUND
```

(scrolled to the right screen 2)

DBName	MbrName	Repo	IMSid	TYPE	Acc	LAcc	Rsdnt	LDRsdnt	LRsdnt
BANKATMS	IMS1	Y			EXCL		N		
BANKATMS	IMS1	Y	IMS1		EXCL		N		
BANKATMS	IMS1	Y	IMS2		EXCL		N		
BANKATMS	IMS1	Y	IMS3		EXCL		N		
BANKATMS	IMS1	Y	IMS4		EXCL		N		
BANKATMS	IMS1		IMS1			EXCL		N	
BANKATMS	IMS2		IMS2			EXCL		N	
BANKATMS	IMS3		IMS3			EXCL		N	
BE3PARTS	IMS1	Y			EXCL		N		
BE3PARTS	IMS1	Y	IMS1		EXCL		N		
BE3PARTS	IMS1	Y	IMS2		UPD		N		
BE3PARTS	IMS1	Y	IMS3		READ		Y		
BE3PARTS	IMS1	Y	IMS4		EXCL		N		
BE3PARTS	IMS1		IMS1	DL/I		EXCL		N	
BE3PARTS	IMS2		IMS2	DL/I		UPD		N	
BE3PARTS	IMS3		IMS3	DL/I		READ	Y	N	
DEDBJN21	IMS1	Y			UPD		Y		
DEDBJN21	IMS1	Y	IMS1		UPD		Y		
DEDBJN21	IMS1	Y	IMS2		UPD		Y		
DEDBJN21	IMS1	Y	IMS3		UPD		Y		
DEDBJN21	IMS1	Y	IMS4		UPD		Y		
DEDBJN21	IMS1		IMS1	DEDB		UPD			Y
DEDBJN21	IMS2		IMS2	DEDB		UPD			Y
DEDBJN21	IMS3		IMS3	DEDB		UPD			Y
NEWDB1	IMS1	Y							
NEWDB1	IMS1		IMS1			UPD		N	
NEWDB1	IMS2								
NEWDB1	IMS3								

OM API input:

```
CMD(QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1)
SHOW(DEFN,IMSID,ACCTYPE,RESIDENT))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.181 22:47:45.395196</statime>
```

```
<stotime>2011.181 22:47:45.414012</stotime>
<staseq>C8005F93F4DFC05C</staseq>
<stoseq>C8005F93F977C345</stoseq>
<rqsttkn1>USRT005 10154745</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC      </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
<mbr name="IMS3    ">
<typ>IMS     </typ>
<styp>DBDC      </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
<mbr name="IMS2    ">
<typ>IMS     </typ>
<styp>DBDC      </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DB              </kwd>
<input>QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1)
 SHOW(DEFN,IMSID,ACCTYPE,RESIDENT) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RACC" llbl="Acc" scope="GBL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RRSDNT" llbl="Rsdnt" scope="GBL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="RSDNT" llbl="LDRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
```

```
<hdr slbl="LRSDNT" llbl="LRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS1    ) LRSDNT(N) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) TYP(DEDB    ) LACC(UPD)
 IMSID(IMS1    ) LRSDNT(Y) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) TYP(DL/I    ) LACC(EXCL)
 IMSID(IMS1    ) LRSDNT(N) </rsp>
<rsp>DB(NEWDB1  ) MBR(IMS1    ) CC(   0) TYP(        ) LACC(UPD)
 IMSID(IMS1    ) LRSDNT(N) </rsp>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
</rsp>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS1    )
 RACC(EXCL) RRSDNT(N) </rsp>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS2    )
 RACC(EXCL) RRSDNT(N) </rsp>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS3    )
 RACC(EXCL) RRSDNT(N) </rsp>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS4    )
 RACC(EXCL) RRSDNT(N) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) RACC(UPD ) RRSDNT(Y)
</rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS1    )
 RACC(UPD ) RRSDNT(Y) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS2    )
 RACC(UPD ) RRSDNT(Y) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS3    )
 RACC(UPD ) RRSDNT(Y) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS4    )
 RACC(UPD ) RRSDNT(Y) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
</rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS1    )
 RACC(EXCL) RRSDNT(N) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS2    )
 RACC(UPD ) RRSDNT(N) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS3    )
 RACC(READ) RRSDNT(Y) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS4    )
 RACC(EXCL) RRSDNT(N) </rsp>
<rsp>DB(NEWDB1  ) MBR(IMS1    ) CC( 1D3) CCTXT(REPOSITORY MEMBER NOT
 FOUND) REPO(Y) </rsp>
<rsp>DB(BANKATMS) MBR(IMS3    ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS3    ) LRSDNT(N) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS3    ) CC(   0) TYP(DEDB    ) LACC(UPD)
 IMSID(IMS3    ) LRSDNT(Y) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS3    ) CC(   0) TYP(DL/I    ) LACC(READ)
 IMSID(IMS3    ) RSDNT(Y) LRSDNT(N) </rsp>
<rsp>DB(NEWDB1  ) MBR(IMS3    ) CC(  10) CCTXT(NO RESOURCES FOUND)
</rsp>
<rsp>DB(BANKATMS) MBR(IMS2    ) CC(   0) TYP(        ) LACC(EXCL)
 IMSID(IMS2    ) LRSDNT(N) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS2    ) CC(   0) TYP(DEDB    ) LACC(UPD)
 IMSID(IMS2    ) LRSDNT(Y) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS2    ) CC(   0) TYP(DL/I    ) LACC(UPD)
 IMSID(IMS2    ) LRSDNT(N) </rsp>
<rsp>DB(NEWDB1  ) MBR(IMS2    ) CC(  10) CCTXT(NO RESOURCES FOUND)
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: Because SHOW(IMSID) is specified, a line is returned for each IMS that has the resource defined in the repository. Line 5 indicates that the

BANKATMS definition for IMS4 is defined in the repository for IMS4, but the definition has not yet been imported into IMS4 because IMS4 shows no local line for database BANKATMS.

Example 9 for QUERY DB command

TSO SPOC input:
```
QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1) SHOW(DEFN,GLOBAL)
```

TSO SPOC output:

(screen 1)
```
DBName   MbrName   CC CCText                        Repo IMSid   Acc  Rsdnt
BANKATMS IMS1       0                               Y            EXCL N
BE3PARTS IMS1       0                               Y            EXCL N
BE3PARTS IMS1       0                               Y    IMS2    UPD  N
BE3PARTS IMS1       0                               Y    IMS3    READ Y
DEDBJN21 IMS1       0                               Y            UPD  Y
NEWDB1   IMS1     1D3 REPOSITORY MEMBER NOT FOUND Y
```

(scrolled right to screen 2)
```
DBName   MbrName  Repo TimeCreate            TimeUpdate
BANKATMS IMS1      Y    2011.181 15:22:52.65
BE3PARTS IMS1      Y    2011.181 15:22:52.65
BE3PARTS IMS1      Y    2011.181 15:21:25.99
BE3PARTS IMS1      Y    2011.181 15:22:07.39 2011.181 15:45:37.08
DEDBJN21 IMS1      Y    2011.181 15:22:52.65
NEWDB1   IMS1      Y
```

OM API input:
```
CMD(QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1) SHOW(DEFN,GLOBAL)    )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.181 22:57:34.679151</statime>
<stotime>2011.181 22:57:34.700833</stotime>
<staseq>C80061C5F106F650</staseq>
<stoseq>C80061C5F652109E</stoseq>
<rqsttkn1>USRT005 10155734</rqsttkn1>
<rc>0200000C</rc>
<rsn>0000300C</rsn>
<rsnmsg>CSLN055I</rsnmsg>
<rsntxt>The command completed with warning(s).</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1     ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
<mbr name="IMS3     ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntxt>IMS not master, cmd ignored</rsntxt>
</mbr>
<mbr name="IMS2     ">
```

```
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntxt>IMS not master, cmd ignored</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DB             </kwd>
<input>QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1) SHOW(DEFN,GLOBAL)
 </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RACC" llbl="Acc" scope="GBL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RRSDNT" llbl="Rsdnt" scope="GBL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="RTMCR" llbl="TimeCreate" scope="GBL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="RTMUP" llbl="TimeUpdate" scope="GBL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
 RTMCR(2011.181 15:22:52.65) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) RACC(UPD ) RRSDNT(Y)
RTMCR(2011.181 15:22:52.65) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
 RTMCR(2011.181 15:22:52.65) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS2    )
 RACC(UPD ) RRSDNT(N) RTMCR(2011.181 15:21:25.99) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS3    )
 RACC(READ) RRSDNT(Y) RTMUP(2011.181 15:45:37.08) RTMCR(2011.181
 15:22:07.39) </rsp>
<rsp>DB(NEWDB1  ) MBR(IMS1    ) CC( 1D3) CCTXT(REPOSITORY MEMBER NOT
 FOUND) REPO(Y) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The stored resource definition information from the repository is returned for the specified resource names because SHOW(DEFN,GLOBAL) is specified. The access type, resident, create time, and update time from the stored resource definitions in the repository are returned. The output contains generic resource definitions and any IMS-specific sections. The generic resource definition has a blank in the IMSid column. The IMS-specific section includes an IMSID of

the IMS whose definition is different from the generic definition. For readability, the DBName and MbrName columns are repeated when the output is scrolled to the right.

Example 10 for QUERY DB command

TSO SPOC input:
```
QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1) SHOW(DEFN,LOCAL)
```

TSO SPOC output:

(screen 1)
```
DBName   MbrName   CC CCText          IMSid   TYPE   LAcc LDRsdnt LRsdnt
BANKATMS IMS1       0                 IMS1           EXCL         N
BANKATMS IMS2       0                 IMS2           EXCL         N
BANKATMS IMS3       0                 IMS3           EXCL         N
BE3PARTS IMS1       0                 IMS1    DL/I   EXCL         N
BE3PARTS IMS2       0                 IMS2    DL/I   UPD          N
BE3PARTS IMS3       0                 IMS3    DL/I   READ Y       N
DEDBJN21 IMS1       0                 IMS1    DEDB   UPD          Y
DEDBJN21 IMS2       0                 IMS2    DEDB   UPD          Y
DEDBJN21 IMS3       0                 IMS3    DEDB   UPD          Y
NEWDB1   IMS1       0                 IMS1           UPD          N
NEWDB1   IMS2      10 NO RESOURCES FOUND
NEWDB1   IMS3      10 NO RESOURCES FOUND
```

(scrolled right to screen 2)
```
DBName   MbrName  LTimeCreate          LTimeUpdate          LTimeAccess
BANKATMS IMS1     2011.181 15:22:52.65
BANKATMS IMS2     2011.181 15:21:25.99
BANKATMS IMS3     2011.181 15:22:07.39
BE3PARTS IMS1     2011.181 15:22:52.65
BE3PARTS IMS2     2011.181 15:21:25.99 2011.181 15:42:22.19
BE3PARTS IMS3     2011.181 15:22:07.39 2011.181 15:45:32.27
DEDBJN21 IMS1     2011.181 15:22:52.65
DEDBJN21 IMS2     2011.181 15:21:25.99
DEDBJN21 IMS3     2011.181 15:22:07.39
NEWDB1   IMS1     2011.181 15:42:43.01
NEWDB1   IMS2
NEWDB1   IMS3
```

OM API input:
```
CMD(QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1) SHOW(DEFN,LOCAL)     )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.181 23:06:01.535261</statime>
<stotime>2011.181 23:06:01.559762</stotime>
<staseq>C80063A95131DA82</staseq>
<stoseq>C80063A9572D2645</stoseq>
<rqsttkn1>USRT005 10160601</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
<rsnmsg>CSLN023I</rsnmsg>
<rsntxt>At least one request was successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS3    ">
<typ>IMS     </typ>
```

```
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
<mbr name="IMS2    ">
<typ>IMS    </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DB              </kwd>
<input>QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS,NEWDB1) SHOW(DEFN,LOCAL)
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RSDNT" llbl="LDRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LRSDNT" llbl="LRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(    0) TYP(        ) LACC(EXCL)
 IMSID(IMS1    ) LRSDNT(N) TMCR(2011.181 15:22:52.65) TMUP(
        ) TMIM(                     ) TMAC(                     ) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(    0) TYP(DEDB    ) LACC(UPD)
IMSID(IMS1    ) LRSDNT(Y) TMCR(2011.181 15:22:52.65) TMUP(
        ) TMIM(                     ) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(    0) TYP(DL/I    ) LACC(EXCL)
 IMSID(IMS1    ) LRSDNT(N) TMCR(2011.181 15:22:52.65) TMUP(
        ) TMIM(                     ) TMAC(                     ) </rsp>
<rsp>DB(NEWDB1  ) MBR(IMS1    ) CC(    0) TYP(        ) LACC(UPD)
 IMSID(IMS1    ) LRSDNT(N) TMCR(2011.181 15:42:43.01) TMUP(
        ) TMIM(                     ) TMAC(                     ) </rsp>
<rsp>DB(BANKATMS) MBR(IMS3    ) CC(    0) TYP(        ) LACC(EXCL)
```

```
     IMSID(IMS3     ) LRSDNT(N) TMCR(2011.181 15:22:07.39) TMUP(
         ) TMIM(                    ) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS3     ) CC(   0) TYP(DEDB     ) LACC(UPD)
 IMSID(IMS3     ) LRSDNT(Y) TMCR(2011.181 15:22:07.39) TMUP(
         ) TMIM(                    ) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS3     ) CC(   0) TYP(DL/I     ) LACC(READ)
 IMSID(IMS3     ) RSDNT(Y) LRSDNT(N) TMCR(2011.181 15:22:07.39)
 TMUP(2011.181 15:45:32.27) TMIM(                    ) TMAC(
         ) </rsp>
<rsp>DB(NEWDB1  ) MBR(IMS3     ) CC(  10) CCTXT(NO RESOURCES FOUND)
</rsp>
<rsp>DB(BANKATMS) MBR(IMS2     ) CC(   0) TYP(         ) LACC(EXCL)
 IMSID(IMS2     ) LRSDNT(N) TMCR(2011.181 15:21:25.99) TMUP(
         ) TMIM(                    ) TMAC(                    ) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS2     ) CC(   0) TYP(DEDB     ) LACC(UPD)
 IMSID(IMS2     ) LRSDNT(Y) TMCR(2011.181 15:21:25.99) TMUP(
         ) TMIM(                    ) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS2     ) CC(   0) TYP(DL/I     ) LACC(UPD)
 IMSID(IMS2     ) LRSDNT(N) TMCR(2011.181 15:21:25.99) TMUP(2011.181
 15:42:22.19) TMIM(                    ) TMAC(                    )
</rsp>
<rsp>DB(NEWDB1  ) MBR(IMS2     ) CC(  10) CCTXT(NO RESOURCES FOUND)
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The runtime resource definitions at the IMS system are returned for the specified resource names. Because SHOW(LOCAL) is specified, only the local runtime definitions from each IMS system are returned. No stored resource definitions from repository are returned.

Example 11 for QUERY DB command

TSO SPOC input:
```
QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS) SHOW(IMSID)
```

TSO SPOC output:
```
DBName    MbrName    CC Repo IMSid
BANKATMS IMS1        0 Y    IMS1
BANKATMS IMS1        0 Y    IMS2
BANKATMS IMS1        0 Y    IMS3
BANKATMS IMS1        0 Y    IMS4
BE3PARTS IMS1        0 Y    IMS1
BE3PARTS IMS1        0 Y    IMS3
BE3PARTS IMS1        0 Y    IMS2
BE3PARTS IMS1        0 Y    IMS4
DEDBJN21 IMS1        0 Y    IMS3
DEDBJN21 IMS1        0 Y    IMS1
DEDBJN21 IMS1        0 Y    IMS2
DEDBJN21 IMS1        0 Y    IMS4
```

OM API input:
```
CMD(QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS) SHOW(IMSID))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.181 23:14:08.509271</statime>
<stotime>2011.181 23:14:08.533678</stotime>
<staseq>C8006579BB557B48</staseq>
```

```
<stoseq>C8006579C14AE3CA</stoseq>
<rqsttkn1>USRT005 10161408</rqsttkn1>
<rc>02000004</rc>
<rsn>00001014</rsn>
<rsnmsg>CSLN055I</rsnmsg>
<rsntxt>At least one request completed with warning(s).</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS3    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntxt>IMS not master, cmd ignored</rsntxt>
</mbr>
<mbr name="IMS2    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntxt>IMS not master, cmd ignored</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DB             </kwd>
<input>QRY DB NAME(BANKATMS,DEDBJN21,BE3PARTS) SHOW(IMSID) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS1    ) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS1    ) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS1    ) </rsp>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS2    ) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS2    ) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS2    ) </rsp>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS3    ) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS3    ) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS3    ) </rsp>
<rsp>DB(BANKATMS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS4    ) </rsp>
<rsp>DB(BE3PARTS) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS4    ) </rsp>
<rsp>DB(DEDBJN21) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS4    ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The SHOW(IMSID) keyword without the DEFN filter is specified.
The IMSID information from the repository is returned for the specified resource

names. The IMSIDs that have the specified resource names in their IMS resource list in the repository are returned in the IMSid column. All the IMSID information is returned in the IMSid column.

Example 12 for QUERY DB command

In this example, TYPE is set to DHISNDX for a HISAM or a SHISAM secondary index database.

TSO SPOC input:
```
QUERY DB NAME(D*) SHOW(ACCTYPE)
```

TSO SPOC output:
```
DATABASE   TYPE      TOTAL UNUSED   TOTAL UNUSED ACC   CONDITIONS     IMS1
DBHDOJ01   PHDAM                                 UP                   IMS1
PDHDOJA    PART                                  UP    NOTOPEN        IMS1
PDHDOJB    PART                                  UP    NOTOPEN        IMS1
PDHDOJC    PART                                  UP    NOTOPEN        IMS1
PDHDOJD    PART                                  UP    NOTOPEN        IMS1
DEDBJN24   DEDB      SEQ DEPEND   DIRECT ADDRES  UP    NOTOPEN        IMS1
DB24A000   AREA      N/A    N/A     N/A     N/A  N/A   NOTOPEN        IMS1
DB24A001   AREA      N/A    N/A     N/A     N/A  N/A   NOTOPEN        IMS1
DB24A239   AREA      N/A    N/A     N/A     N/A  N/A   NOTOPEN        IMS1
DEHSJX24   DHISNDX                               UP    NOTOPEN        IMS1
*09043/155658*     IMS1
```

Explanation: DBHDOJ01 is a PHDAM database with four partitions: PDHDOJA, PDHDOJB, PDHDOJC, and PDHDOJD. DEDBJN24 is a HISAM or SHISAM secondary index database with three areas: DB24A000, DB24A001, and DB24A239. DEHSJX24 is a HISAM or a SHISAM secondary index database. The local access of each database is UPDATE.

Example 13 for QUERY DB command

In this example, TYPE is set to DHISNDX for a HISAM or a SHISAM secondary index database.

TSO SPOC input:
```
QUERY DB NAME(D*) SHOW(ACCTYPE)
```

TSO SPOC output:
```
DBName      AreaName PartName  MbrName  CC   TYPE      LAcc
DBHDOJ01                       IMS1     0    PHDAM     UPD
DBHDOJ01             PDHDOJA   IMS1     0    PART      UPD
DBHDOJ01             PDHDOJB   IMS1     0    PART      UPD
DBHDOJ01             PDHDOJC   IMS1     0    PART      UPD
DBHDOJ01             PDHDOJD   IMS1     0    PART      UPD
DEDBJN24                       IMS1     0    DEDB      UPD
DEDBJN24    DB24A000           IMS1     0    AREA
DEDBJN24    DB24A001           IMS1     0    AREA
DEDBJN24    DB24A239           IMS1     0    AREA
DEHSJX24                       IMS1     0    DHISNDX   UPD
```

Explanation: DBHDOJ01 is a PHDAM database with four partitions: PDHDOJA, PDHDOJB, PDHDOJC, and PDHDOJD. DEDBJN24 is a HISAM or SHISAM secondary index database with three areas: DB24A000, DB24A001, and DB24A239. DEHSJX24 is a HISAM or a SHISAM secondary index database. The local access type of each database is UPDATE.

Example 14 for QUERY DB command

TSO SPOC input:

```
QUERY DB NAME(D*) TYPE(DHISNDX,DEDB)
```

TSO SPOC output:

```
DBName      AreaName   MbrName   CC   TYPE
DEDBJN24               IMS1      0    DEDB
DEDBJN24    DB24A000   IMS1      0    AREA
DEDBJN24    DB24A001   IMS1      0    AREA
DEDBJN24    DB24A239   IMS1      0    AREA
DEHSJX24               IMS1      0    DHISNDX
```

Explanation: DEDBJN24 is a HISAM or SHISAM secondary index database with three areas: DB24A000, DB24A001, and DB24A239. DEHSJX24 is a HISAM or a SHISAM secondary index database for a DEDB primary database.

Example 15 for QUERY DB command

TSO SPOC input:

```
QUERY DB NAME(DEDBGS1A,FPSI1AH*) SHOW(SNDX)
```

TSO SPOC output:

```
Response for: QUERY DB NAME(DEDBGS1A,FPSI1AH*) SHOW(SNDX)
DBName    AreaName SndxName  MbrName    CC  CCText        TYPE
DEDBGS1A                     IMS1       0                 DEDB
DEDBGS1A           FPSI1AHA  IMS1       0                 DHISNDX
DEDBGS1A           FPSI1AH1  IMS1       0                 DHISNDX
DEDBGS1A           FPSI1AH2  IMS1       0                 DHISNDX
DEDBGS1A  GS1AAR0            IMS1       0                 AREA
DEDBGS1A  GS1AAR1            IMS1       0                 AREA
DEDBGS1A  GS1AAR2            IMS1       0                 AREA
DEDBGS1A  GS1AAR3            IMS1       0                 AREA
DEDBGS1A  GS1AAR4            IMS1       0                 AREA
DEDBGS1A  GS1AAR5            IMS1       0                 AREA
DEDBGS1A  GS1AAR6            IMS1       0                 AREA
FPSI1AHA                     IMS1       193 NOT A DEDB
FPSI1AH1                     IMS1       193 NOT A DEDB
FPSI1AH2                     IMS1       193 NOT A DEDB
```

Explanation: DEDBGS1A is a DEDB database with seven areas: GS1AAR0, GS1AAR1, GS1AAR2, GS1AAR3, GS1AAR4, GS1AAR5, and GS1AAR6. DEDBGS1A has three HISAM or SHISAM secondary index databases: FPSI1AHA, FPSI1AH1, and FPSI1AH2. Because databases FPSI1AHA, FPSI1AH1, and FPSI1AH2 are not DEDB databases, they cannot have Fast Path secondary index databases defined. These databases get a completion code of 193, which indicates that they are not DEDB databases.

Example 16 for QUERY DB command

The following example shows a HALDB master database, DBVHDJ05, and two partitions, PVHDJ5B and PVHDJ5D, that are currently being processed by the HALDB online alter function.

TSO SPOC input:

```
QRY DB NAME(DBVHDJ05) SHOW(WORK)
```

TSO SPOC output:

```
DBName      PartName MbrName   CC    TYPE       Work
DBVHDJ05             IMS1      0     PHDAM      ALTER INCOMPLETE
DBVHDJ05    PVHDJ5B  IMS1      0     PART       ALTER INCOMPLETE
DBVHDJ05    PVHDJ5D  IMS1      0     PART       ALTER INCOMPLETE
```

Example 17 for QUERY DB command

TSO SPOC input:

```
QUERY DB NAME(DB0*) SHOW(EXPORTNEEDED)
```

TSO SPOC output:

```
DBName     MbrName    CC TYPE     ExportNeeded
----------------------------------------------------------------
DB000001 IMS1         0           N
DB000002 IMS1         0           Y
```

OM API input:

```
CMD(QUERY DB SHOW(EXPORTNEEDED))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20</xmlvsn>
<statime>2014.337 22:30:51.302415 22:30:51</statime>
<stotime>2014.337 22:30:51.302792</stotime>
<staseq>CE267AE1EFA0F110</staseq>
<stoseq>CE267AE1EFB88990</stoseq>
<rqsttkn1>1234            </rqsttkn1>
<rc>00000000</rc><rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>DB            </kwd>
<input>QUERY DB NAME(DB%0000%) SHOW(EXPORTNEEDED) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PART" llbl="PartName" scope="LCL" sort="a" key="5"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYP" llbl="TYPE" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="EXPN" llbl="ExportNeeded" scope="LCL" sort="y" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(DBR00001) MBR(IMS1    ) CC(   0) TYP(        ) EXPN(Y) </rsp>
<rsp>DB(DB000002) MBR(IMS1    ) CC(   0) TYP(        ) EXPN(Y) </rsp>
</cmdrspdata>
</imsout>
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY DB command (Commands)

/DISPLAY DBD command (Commands)

/DISPLAY MODIFY command (Commands)

/DISPLAY STATUS command (Commands)

Command keywords and their synonyms (Commands)

QUERY DBDESC command

Use the QUERY DBDESC command to query information about database descriptors. A descriptor is a model that can be used to create descriptors or resources.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 102
- "Usage notes" on page 105
- "Output fields" on page 106
- "Return, reason, and completion codes" on page 110
- "Examples" on page 112

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 25. Valid environments for the QUERY DBDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY DBDESC	X	X	
NAME	X		X
DEFAULT	X		X
SHOW	X		X

Syntax

```
|                       ►──────────────────────────────────────────────────────◄
                            ┌──────────,──────────┐
                            │              ▼       │
                        ─SHOW(─┬─ALL──────┬────────┐
                               ├─ACCTYPE──┤
                               ├─DEFAULT──┤
                               ├─DEFN─────┤
                               ├─DEFNTYPE─┤
                               ├─GLOBAL───┤
                               ├─IMSID────┤
                               ├─LOCAL────┤
                               ├─MODEL────┤
                               ├─RESIDENT─┤
                               └─TIMESTAMP┘
                        └─SHOW(EXPORTNEEDED)────────┘
```
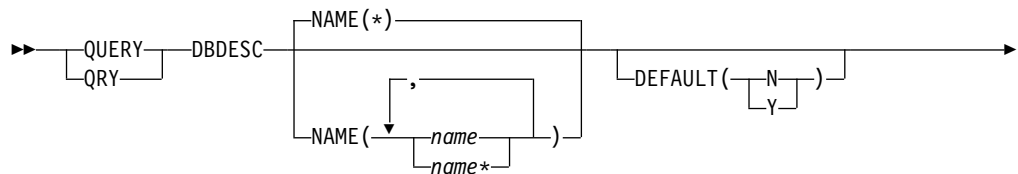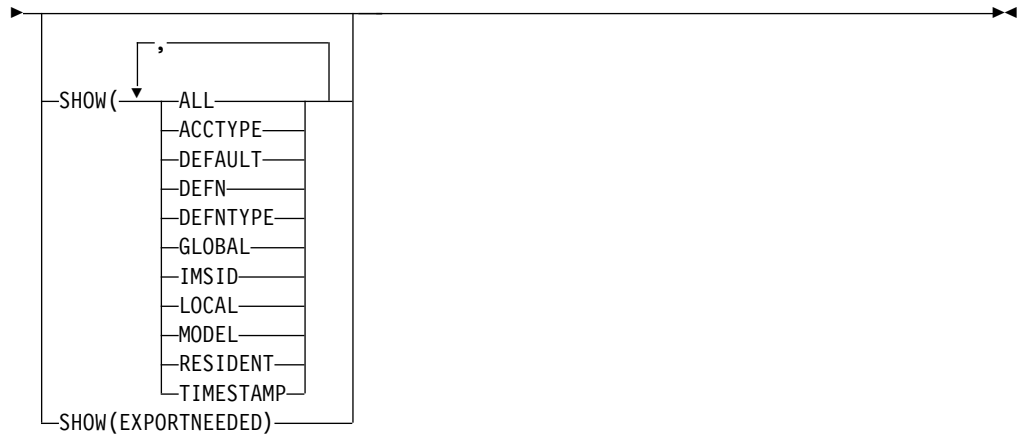
Keywords

The following keywords are valid for the QUERY DBDESC command:

DEFAULT()

Specifies which descriptor or descriptors to display.

N Displays all the database descriptors that are not the default.

Y Displays the default database descriptor.

NAME()

Specifies the 1-8 character name of the database descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all database descriptors.

SHOW()

Specifies the database descriptor output fields to be returned. The database descriptor name is always returned, along with the name of the IMS that created the output for the database descriptor and the completion code. The filters supported with the SHOW keyword are:

ALL

Returns all information about the database descriptor itself.

Restriction: The ExportNeeded column, which is returned when SHOW(EXPORTNEEDED) is specified, is not returned with SHOW(ALL).

ACCTYPE

Type of access to the database created using this database descriptor.

DEFAULT

Default descriptor option.

DEFN

Specifies that the resource definitions are to be returned.

The database descriptor attributes that can be returned are: ACCTYPE, DEFAULT, RESIDENT, the repository create and update time stamps, and the IMS runtime create, update, import, and access time stamps.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes, including those defined globally in the repository and those defined locally in the IMS system, are returned. The runtime resource definitions from the IMS system are returned by each

IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE or MODEL.
- The LModelName, LModelType, and LDefnType columns, which are returned on the QRY DBDESC SHOW(ALL) command, are not returned with SHOW(DEFN).
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).

When querying database descriptor information from the repository, the resource definitions stored in the repository are used to determine the response lines with the repository information, and the runtime resource definitions are used to determine the response lines with the IMS runtime resource information. The response lines are returned for each stored resource or runtime resource definition that matches the specified filter. If SHOW(DEFN,GLOBAL) is specified, only the stored resource definitions that match the specified filter are returned. If SHOW(DEFN,LOCAL) is specified, only the runtime resource definitions that match the specified filter are returned.

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

DEFNTYPE
Definition type. This is how the descriptor was defined.

EXPORTNEEDED
Specifies that only the database descriptors that match the NAME

parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the database descriptors that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each database descriptor name specified on the command. If the specified database descriptor does not need to be exported to the repository, an N is returned in the ExportNeeded column.

SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords and is valid only when IMS is enabled with the IMSRSC repository.

The QUERY DBDESC SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources and descriptors that have not been exported to the IMSRSC repository, you can issue the EXPORT DEFN TARGET(REPO) OPTION(CHANGESONLY) command to write the resource and descriptor definitions to the repository so that they are available for the next IMS cold start.

GLOBAL
Specifies that the stored resource definitions from the repository are to be returned.

If SHOW(GLOBAL,DEFN) is specified, the global resource definitions from the repository are returned by the command master IMS. SHOW(GLOBAL,DEFN) is valid only when the command master IMS is enabled to use the repository.

IMSID
Specifies that the IMSID of the IMS systems whose resource lists contain the resource name are to be returned.

SHOW(IMSID) is processed only by the command master IMS and is valid only when command master IMS is enabled to use the repository.

When SHOW(IMSID) is specified with the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

SHOW(IMSID) cannot be specified with any other SHOW filters other than DEFN and GLOBAL. If SHOW(IMSID,GLOBAL) is specified, GLOBAL is ignored; that is, SHOW(IMSID,GLOBAL) is treated as SHOW(IMSID). SHOW(DEFN,IMSID,LOCAL) is treated as SHOW(DEFN,LOCAL).

LOCAL
Specifies that the runtime resource definitions from the IMS system are to be returned.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

MODEL

The model name and model type used to create this descriptor. The model name and model type is blank for the IMS-defined descriptor DFSDSDB1. The CREATE command specified without the LIKE keyword creates a descriptor using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSDB1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor using a model. The descriptor is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify descriptors that were created with the same model. The model name and model type of a descriptor are exported and imported. The IMPORT command does not use the model name and model type when creating a descriptor.

RESIDENT

Specifies the resident option. The value is always RESIDENT(Y) for Fast Path DEDBs. The RESIDENT(Y) option takes effect at the next restart, unless the database descriptor was created or updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart.

TIMESTAMP

The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS) time, and import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

Usage notes

This command can be issued only through the Operations Manager API. This command applies to DB/DC and DBCTL systems.

This command is allowed on XRF alternate and RSR tracker systems.

The QUERY DBDESC command is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

If you want to display information about resource definitions, specify SHOW(DEFN). If you want to know which IMS systems have the resource defined and also know the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource defined, specify SHOW(IMSID).

Output fields

The following table shows the QUERY DBDESC output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 26. Output fields for the QUERY DBDESC command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified descriptor. The completion code is always returned. Refer to the return, reason, and completion codes for QUERY DBDESC.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the nonzero completion code.
DESC	DescName	DBDESC	N/A	Database descriptor name.
DFLT	LDflt	DBDESC	LCL	Default descriptor (Y) or not (N).
				N The descriptor is not the default.
				Y The descriptor is the default. When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a database descriptor called DFSDSDB1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor.

Table 26. Output fields for the QUERY DBDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning

DFNT	LDefnType	DEFNTYPE	N/A	Definition type, which can be one of the following:
				
				CREATE
				    Defined by a CREATE command.
				
				IMPORT
				    Defined by an IMPORT command. The DEFNTYPE is not changed if the descriptor or resource is updated with an UPDATE command.
				
				IMS
				    Defined by IMS. DFSDSDB1 is an IMS-defined database descriptor containing the default database descriptor values.

EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the database descriptor has been exported to the IMSRSC repository. The value can be Y or N.
IMSID	IMSid	IMSID	GBL	Returns the IMS IDs that have resources defined from the repository.

Table 26. Output fields for the QUERY DBDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
LACC	LAcc	ACCTYPE	LCL	Type of access to database descriptor which can be one of the following:
				BRWS The database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data since a sharing IMS subsystem could be updating the database. The database is opened for read-only processing.
				EXCL The named database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC.
				READ The database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. Access type READ differs from access type BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.
				UPD The database is available for update as well as read processing in the IMS subsystem.
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. The IMS identifier of the IMS that built the output. The IMS identifier is always returned.
MDLN	LModelName	MODEL	N/A	Model name. Name of the resource or descriptor used as a model to create this descriptor. DFSDSDB1 is the IMS descriptor name for databases.
MDLT	LModelType	MODEL	N/A	Model type, either RSC or DESC. RSC means that the descriptor was created using another resource as a model. DESC means that the descriptor was created using a descriptor as the model.

Table 26. Output fields for the QUERY DBDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
PGM	LPgmName	PGM	LCL	Program name from the local IMS.
RACC	Acc	DEFN	GBL	Access type obtained from the repository.
RDFLT	Dflt	DEFN	GBL	Default value from the repository.
REPO	Repo	DEFN	GBL	Indicates whether the line contains stored resource definitions. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
RRSDNT	Rsdnt	DEFN	GBL	Resident value obtained from the repository.
RSDNT	LDRsdnt	RESIDENT	LCL	Resident option value. For a database created from the descriptor, it indicates whether the DMB is to reside in local storage at the next IMS restart. **N** The DMB for a database created from the named database descriptor resource is not made resident in storage. The DMB is loaded at scheduling time. **Y** The DMB for a database created from the named database descriptor resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the DMB and initializes it. A resident database is accessed from local storage, which eliminates I/O to the ACBLIB.
RTMCR	TimeCreate	DEFN	GBL	Create time from the repository. This is the time the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN	GBL	Update time from the repository. This is the time the resource was last updated in the repository.

Table 26. Output fields for the QUERY DBDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the descriptor was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. This access time stamp value is obtained from the local IMS. For a database descriptor, when the CREATE command references the descriptor as the model, the last access time is updated.
TMCR	LTimeCreate	TIMESTAMP	LCL	The time the descriptor was created. This is the result of a CREATE DB command, IMPORT command that creates the database descriptor, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT. This create time stamp value is obtained from the local IMS.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the descriptor was last imported, if applicable. The import time is retained across warm start and emergency restart. This import time stamp value is obtained from the local IMS.
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated as a result of the UPDATE DBDESC command or the IMPORT command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 27. Return and reason codes for the QUERY DBDESC command

Return code	Reason code	Meaning
X'00000004'	X'00001010'	No resources were found to be returned. The descriptor names specified might be invalid, or: • There were no resources that match the filter specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'0000000C'	X'00003004'	No requests were successful.
X'00000010'	X'00004004'	No CQS address space.
X'00000010'	X'00004018'	No resource structure, or resource structure is not available.
X'00000010'	X'00004100'	Resource structure is full.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Repository access was denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than IMS data version.
X'00000010'	X'0000450A'	Repository Server is being shut down.
X'00000010'	X'0000450B'	Repository Server is not available.
X'00000010'	X'0000450C'	Repository Server is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields that are related to the IMSRSC repository.
X'00000014'	X'0000501C'	IMODULE GETMAIN error.
X'00000014'	X'00005100'	RM request error.
X'00000014'	X'00005104'	Unexpected CQS error.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.

Table 27. Return and reason codes for the QUERY DBDESC command (continued)

Return code	Reason code	Meaning
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY DBDESC command is rejected because the resource list is unusable. Contact IBM Software Support. Send in the output from the command in error, and a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the IMSRSC repository RID and RMD data sets.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 28. Completion codes for the QUERY DBDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for database or database descriptor.
10	NO RESOURCES FOUND	Database descriptor name is invalid, or the wildcard parameter specified does not match any database descriptor names.

Examples

The following are examples of the QUERY DBDESC command:

Example 1 for QUERY DBDESC command

TSO SPOC input:

```
QRY DBDESC SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
DescName MbrName    CC LAcc LDRsdnt LDflt LModelName LModelType LTimeCreate
DFSDSDB1 IMS1        0 UPD  N       Y                           2011.181 10:22:15.10
(scrolled to the right screen 2)
DescName MbrName  LTimeUpdate        LTimeAccess        LTimeImport
DFSDSDB1 IMS1
(scrolled to the right screen 3)
DescName MbrName  LDefnType
DFSDSDB1 IMS1     IMS
```

OM API input:

```
CMD(QUERY DBDESC SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.181 17:40:56.620170</statime>
```

```
<stotime>2011.181 17:40:56.621294</stotime>
<staseq>C8001AFFFB08A640</staseq>
<stoseq>C8001AFFFB4EE240</stoseq>
<rqsttkn1>USRT005 10104056</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DBDESC        </kwd>
<input>QUERY DBDESC SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RSDNT" llbl="LDRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DFLT" llbl="LDflt" scope="LCL" sort="n" key="0"
 scroll="yes"len="1" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MDLN" llbl="LModelName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MDLT" llbl="LModelType" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DFSDSDB1) MBR(IMS1    ) CC(   0) LACC(UPD) DFNT(IMS)
 RSDNT(N) TMCR(2011.181 10:22:15.10) TMUP(              )
 TMIM(                   ) TMAC(                   ) DFLT(Y) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The database descriptors are displayed. There is one database
descriptor in the system: the IMS-defined descriptor DFSDSDB1. All output fields
are returned for the descriptor. All of the database descriptor output fields do not
fit on one screen, so you must scroll to the right for additional output fields. The
database descriptor name and member name that built the line of output are
displayed on every screen. The time the DFSDSDB1 descriptor was created was at
IMS cold start time. IMS-defined descriptors display a blank LModelName and
LModelType, since no model is used to create them. The Dflt column shows a value
of Y, which indicates that DFSDSDB1 is defined as the default database descriptor.

Example 2 for QUERY DBDESC command

TSO SPOC input:

```
QUERY DBDESC NAME(*) SHOW(DEFN,ACCTYPE,RESIDENT)
```

TSO SPOC output:

```
DescName MbrName   CC Repo IMSid    Acc  LAcc Rsdnt LDRsdnt
DBDESC1  IMS1       0 Y             EXCL      N
DBDESC1  IMS1       0    IMS1            EXCL      N
DBDESC1  IMS2       0    IMS2            EXCL      N
DBDESC1  IMS3       0    IMS3            EXCL      N
DEDBDESC IMS1       0 Y             EXCL      N
DEDBDESC IMS1       0    IMS1            EXCL      N
DEDBDESC IMS2       0    IMS2            EXCL      N
DEDBDESC IMS3       0    IMS3            EXCL      N
DFSDSDB1 IMS1       0    IMS1            UPD       N
DFSDSDB1 IMS2       0    IMS2            UPD       N
DFSDSDB1 IMS3       0    IMS3            UPD       N
NEWDESC  IMS1       0    IMS1            READ      N
```

OM API input:

```
CMD(QUERY DBDESC NAME(*) SHOW(DEFN,ACCTYPE,RESIDENT))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.181 18:22:06.526302</statime>
<stotime>2011.181 18:22:06.610106</stotime>
<staseq>C80024337775E358</staseq>
<stoseq>C80024338BEBAB18</stoseq>
<rqsttkn1>USRT005 10112206</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DBDESC          </kwd>
<input>QUERY DBDESC NAME(*) SHOW(DEFN,ACCTYPE,RESIDENT) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RACC" llbl="Acc" scope="GBL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LACC" llbl="LAcc" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RRSDNT" llbl="Rsdnt" scope="GBL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="RSDNT" llbl="LDRsdnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DEDBDESC) MBR(IMS1    ) CC(   0) LACC(EXCL) IMSID(IMS1    )
```

```
    RSDNT(N) </rsp>
<rsp>DESC(DBDESC1 ) MBR(IMS1    ) CC(   0) LACC(EXCL) IMSID(IMS1    )
    RSDNT(N) </rsp>
<rsp>DESC(NEWDESC ) MBR(IMS1    ) CC(   0) LACC(READ) IMSID(IMS1    )
    RSDNT(N) </rsp>
<rsp>DESC(DFSDSDB1) MBR(IMS1    ) CC(   0) LACC(UPD) IMSID(IMS1    )
    RSDNT(N) </rsp>
<rsp>DESC(DBDESC1 ) MBR(IMS1    ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
    </rsp>
<rsp>DESC(DEDBDESC) MBR(IMS1    ) CC(   0) REPO(Y) RACC(EXCL) RRSDNT(N)
    </rsp>
<rsp>DESC(DEDBDESC) MBR(IMS3    ) CC(   0) LACC(EXCL) IMSID(IMS3    )
    RSDNT(N) </rsp>
<rsp>DESC(DBDESC1 ) MBR(IMS3    ) CC(   0) LACC(EXCL) IMSID(IMS3    )
    RSDNT(N) </rsp>
<rsp>DESC(DFSDSDB1) MBR(IMS3    ) CC(   0) LACC(UPD) IMSID(IMS3    )
    RSDNT(N) </rsp>
<rsp>DESC(DEDBDESC) MBR(IMS2    ) CC(   0) LACC(EXCL) IMSID(IMS2    )
    RSDNT(N) </rsp>
<rsp>DESC(DBDESC1 ) MBR(IMS2    ) CC(   0) LACC(EXCL) IMSID(IMS2    )
    RSDNT(N) </rsp>
<rsp>DESC(DFSDSDB1) MBR(IMS2    ) CC(   0) LACC(UPD) IMSID(IMS2    )
    RSDNT(N) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The stored resource definitions and the runtime resource definitions
for the specified resources are returned. DBDESC1 and DEDBDESC have the stored
resource definitions in the repository and also at the IMS systems. The descriptor
NEWDESC is created at IMS1 but does not exist in the repository or at any other
IMS systems.

Because SHOW(ACCTYPE,RESIDENT) is specified, only the resident and access
type information is returned. DFSDSDB1 is the default descriptor and is only at
each of the IMS systems. The default descriptor definitions are not in the
repository.

Example 3 for QUERY DBDESC command

TSO SPOC input:
```
QUERY DBDESC DEFAULT(Y)
```

TSO SPOC output:
```
DescName MbrName    CC LDflt
DFSDSDB1 IMS1        0 Y
DFSDSDB1 IMS2        0 Y
DFSDSDB1 IMS3        0 Y
```

OM API input:
```
CMD(QUERY DBDESC DEFAULT(Y))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.181 18:26:10.474718</statime>
<stotime>2011.181 18:26:10.498099</stotime>
<staseq>C800251C1D2DEC12</staseq>
<stoseq>C800251C22E33512</stoseq>
```

```
<rqsttkn1>USRT005 10112610</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>DBDESC       </kwd>
<input>QUERY DBDESC DEFAULT(Y) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DFLT" llbl="LDflt" scope="LCL" sort="n" key="0"
scroll="yes"len="1" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DFSDSDB1) MBR(IMS1    ) CC(   0) DFLT(Y) </rsp>
<rsp>DESC(DFSDSDB1) MBR(IMS3    ) CC(   0) DFLT(Y) </rsp>
<rsp>DESC(DFSDSDB1) MBR(IMS2    ) CC(   0) DFLT(Y) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Only the local default descriptors are returned.

Example 4 for QUERY DBDESC command

TSO SPOC input:

```
QUERY DBDESC SHOW(EXPORTNEEDED)
```

TSO SPOC output:

```
            Return   Reason
  MbrName   Code     Code      Reason text
  IMS1      00000004 00001010  No resources or matches found
```

OM API input:

```
CMD(QUERY DBDESC SHOW(EXPORTNEEDED))
```

Related concepts:

↪ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

↪ Command keywords and their synonyms (Commands)

QUERY IMS command

Use the QUERY IMS command, which is a type-2 command, to display status and attributes information from IMS.

The information that is returned to the user is the IMS status that can be modified with the UPDATE IMS command. The QUERY MEMBER command is used to return other IMS status.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 118
- "Output fields" on page 118
- "Return, reason, and completion codes" on page 120
- "Examples" on page 121

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY IMS command and keywords can be issued.

Table 29. Valid environments for QUERY IMS command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUERY IMS	X	X	X
SHOW	X	X	X
TYPE	X	X	X

Syntax
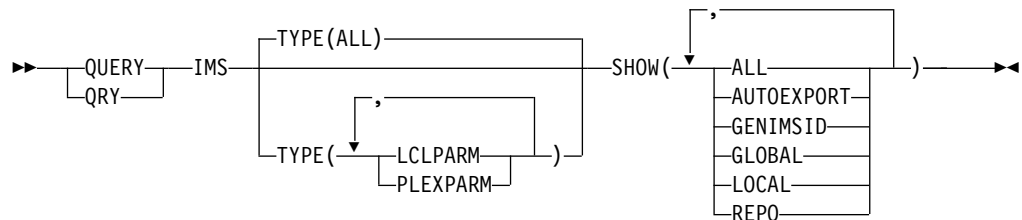
Keywords

The following keywords are valid for the QUERY IMS command:

SHOW()
   (Required) - Specifies the output fields to be returned.

   ALL
      Returns all possible output fields for the data specified in the TYPE parameter.

   AUTOEXPORT
      Returns the AUTOEXPORT status. AUTOEXPORT applies to TYPE(LCLPARM) only.

   GENIMSID
      Returns the value of the GENIMSID parameter of the IMS DFSDCxxx PROCLIB member. The GENIMSID parameter defines a shared generic IMS ID for an MSC TCP/IP generic resource group.

   GLOBAL
      Returns the global values for the specified TYPE options.

LOCAL
  Returns the local values for the specified TYPE options.

REPO
  Returns the IMSRSC repository parameters and indicates whether IMS is enabled with the repository. If IMS is not enabled with the repository, the return and reason codes that indicate "IMS is not enabled with the repository" are returned. REPO applies to TYPE(LCLPARM) only.

TYPE()
Specifies the data to be returned.

ALL
  Returns data for all possible types. ALL is the default.

LCLPARM
  Displays the local LCLPARM values at the current IMS. Information is returned based on the SHOW options specified.

  You cannot specify SHOW(GLOBAL) with TYPE(LCLPARM). SHOW(ALL) and SHOW(LOCAL) return the same local parameter information.

PLEXPARM
  Displays the global PLEXPARM values at the current IMS.

  If global values are requested, the values from the global PLEXPARM entry in RM are returned by the IMS command master. If local values are requested, the values from each IMS that processes the command are returned.

Usage notes

This command can be specified only through the Operations Manager API.

Output fields

The output provided by QUERY IMS is described in the following table. Specifying SHOW parameter options determines what is returned in the output. Specifying SHOW(ALL) returns all output fields.

The columns in the table are as follows:

Short label
  Contains the short label generated in the XML output.

Long label
  Contains the long label generated in the XML output.

Keyword
  Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
  Provides a brief description of the output field.

Table 30. QUERY IMS output fields

Short label	Long label	Keyword	Scope	Meaning
AUTEXP	AutoExport	LCLPARM, LCL AUTOEXPORT	LCL	AUTOEXPORT status value, which can be one of the following: **N** Automatic export is disabled. **Y** Automatic export is enabled to the system RDDS, IMSRSC repository, or both.
BMP	BMP	LCLPARM	LCL	Amount of time that IMS waits before lock requests for BMP regions are timed out. BMP regions include IMS BMP and JBP regions. The value, which represents time in seconds, can range from 1 to 32767.
BMPOPT	BMPOPT	LCLPARM	LCL	Indicates whether IMS ends a timed-out task abnormally (ABEND) or returns a status code to the application (STATUS).
CC	CC	n/a	n/a	Completion code, which indicates whether or not IMS was able to process the command for the specified resource. The completion code is always returned.
CCTXT	CCText	*error*	LCL	Text returned with the completion code to provide additional information. It can include a return code from a service. Completion code text is returned only if the completion code is nonzero.
FP64STAT	FPBP64STAT	LCLPARM	LCL	Fast Path 64-bit buffer usage statistics per unit of work for dependent regions. FPBP64STAT output is displayed only when Fast Path 64-bit buffer manager is enabled. Fast Path 64-bit buffer usage statistics are recorded in X'5945' log records. The values include: **N** Does not write Fast Path 64-bit buffer usage statistics per unit of work for dependent regions in X'5945' log records to OLDS. **Y** Writes Fast Path 64-bit buffer usage per unit of work for dependent regions in X'5945' log records to OLDS.
GSTSAREA	GSTSAREA	PLEXPARM	LCL or GBL	The global status for areas. The values include: **Y** Global status is maintained for areas. **N** Global status is not maintained for areas. **NULL** Global status has not been set for areas.
GSTSDB	GSTSDB	PLEXPARM	LCL or GBL	Global status for databases. The values include: **Y** Global status is maintained for databases. **N** Global status is not maintained for databases. **NULL** Global status has not been set for databases.

Table 30. QUERY IMS output fields (continued)

Short label	Long label	Keyword	Scope	Meaning
GSTSTRAN	GSTSTRAN	PLEXPARM	LCL or GBL	Global status for transactions. The values include:
				Y Global status is maintained for transactions.
				N Global status is not maintained for transactions.
				NULL Global status has not been set for transactions.
LEXPTM	LastExportTime	LCLPARM, REPO	LCL	Time stamp of the last successful export to the repository. The last export time is initialized to the import time if definitions are successfully imported from the repository during cold start or if AUTOIMPORT=REPO is specified in the dynamic resource definition section of the DFSDFxxx PROCLIB member and the repository is empty during cold start.
MBR	MbrName	n/a	n/a	The IMS identifier of the IMS that built the output. The IMS identifier is always returned.
MSG	MSG	LCLPARM	LCL	Amount of time that IMS waits before lock requests for MSG regions are timed out. MSG regions include IMS MPP, JMP, and IFP regions as well as DRA threads. The value, which represents time in seconds, can range from 1 to 32767.
MSGOPT	MSGOPT	LCLPARM	LCL	Indicates whether IMS ends a timed-out task abnormally (ABEND) or returns a status code to the application (STATUS).
REPONM	RepositoryName	LCLPARM, REPO	LCL	Repository name.
REPOTP	RepositoryType	LCLPARM, REPO	LCL	Repository type.
TGN	TcpipGenImsID	LCLPARM, GENIMSID	LCL	Indicates the generic IMS ID name that is used for MSC TCP/IP generic resources, as it is specified on the GENIMSID parameter of the DFSDCxxx PROCLIB member. Remote MSC-enabled IMS systems can use the GENIMSID value to connect to an IMSplex without specifying a specific IMS system.
VGN	VtamGenName	LCLPARM, GENIMSID	LCL	Indicates the name of the VTAM generic resource group, as it is specified on the GRSNAME parameter of the IMS or DCC startup procedure or the /START VGR command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMS command are standard for all commands that are entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 31. Return and reason codes for the QUERY IMS command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMS command completed successfully.
X'00000004'	X'00001000'	The QUERY IMS command is not processed on the IMS system because the IMS system is not the command master.
X'00000008'	X'00002040'	The QUERY IMS command is not processed because no filter or an invalid filter was specified for the SHOW keyword.
X'0000000C'	X'00003004'	No requests were successful.
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000014'	X'00005004'	The QUERY IMS command is not processed because the DFSOCMD0 GETBUF storage could not be obtained.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains completion codes can be returned on a QUERY IMSPLEX command.

Table 32. Completion codes for the QUERY IMS command

Completion code	Meaning
0	The QUERY IMS command completed successfully.
50	The QUERY IMS command is not processed because CQS is not available.
51	The QUERY IMS command is not processed because there is no resource structure.
52	The QUERY IMS command is not processed because the resource structure is full.
53	The QUERY IMS command is not processed because RM is not available.
54	The QUERY IMS command is not processed because SCI is not available.
90	The QUERY IMS command is not processed because of an IMS internal error.
94	The QUERY IMS command is not processed because of an RM error.
95	The QUERY IMS command is not processed because of an RM error.
98	The QUERY IMS command is not processed because of a CQS error.

Examples

The following are examples of the QUERY IMS command:

Example 1 for the QUERY IMS *command*

TSO SPOC input:
```
QRY IMS SHOW(ALL)
```

TSO SPOC output:

(screen 1)
```
MbrName    CC GLOBAL GSTSAREA GSTSDB GSTSTRAN FPBP64STAT   MSG  MSGOPT   BMP
IMS2        0 Y      Y        Y      Y
IMS1        0        Y        Y      Y        Y                 N/A  ABEND   N/A
IMS2        0        Y        Y      Y                          N/A  ABEND   N/A
```

(scrolled to the right screen 2)
```
MbrName   BMPOPT AutoExport RepositoryType
IMS2
IMS1      ABEND  N          IMSRSC
IMS2      ABEND  N
```

(scrolled to the right screen 3)
```
MbrName   RepositoryName                         LastExportTime
IMS2
IMS1      IMSRSC_REPOSITORY                      2011.188 16:48:30.07
IMS2
```

(scrolled to the right screen 4)
```
MbrName   TcpipGenImsID VtamGenName
IMS2
IMS1      IMS           AAA
IMS2
```

OM API input:

```
CMD(QRY IMS SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.189 00:02:23.704750</statime>
<stotime>2011.189 00:02:23.720449</stotime>
<staseq>C8093D51082AE9CC</staseq>
<stoseq>C8093D510C00161C</stoseq>
<rqsttkn1>USRT005 10170223</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>IMS            </kwd>
<input>QRY IMS SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="GLOBAL" llbl="GLOBAL" scope="GBL" sort="d" key="1"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="GSTSAREA" llbl="GSTSAREA" scope="LCL" sort="n" key="0"
scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="GSTSDB" llbl="GSTSDB" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="GSTSTRAN" llbl="GSTSTRAN" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="FP64STAT" llbl="FPBP64STAT" scope="LCL" sort="n" key="0"
```

```
  scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSG" llbl="  MSG" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="MSGOPT"  llbl=" MSGOPT" scope="LCL" sort="n" key="0"
 scroll="yes" len="6" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="BMP" llbl="  BMP" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="BMPOPT"  llbl=" BMPOPT" scope="LCL" sort="n" key="0"
 scroll="yes" len="6" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="AUTEXP" llbl="AutoExport" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPOTP" llbl="RepositoryType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPONM" llbl="RepositoryName" scope="LCL" sort="n" key="0"
 scroll="yes" len="44" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LEXPTM" llbl="LastExportTime" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TGN" llbl="TcpipGenImsID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="VGN" llbl="VtamGenName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS2    ) CC(   0) GLOBAL(Y) GSTSAREA(Y   ) GSTSDB(Y   )
 GSTSTRAN(Y   ) </rsp>
<rsp>MBR(IMS2    ) CC(   0) GSTSAREA(Y   ) GSTSDB(Y   ) GSTSTRAN(Y   )
 MSG(  N/A) MSGOPT(ABEND ) BMP(  N/A) BMPOPT(ABEND ) AUTEXP(N) </rsp>
<rsp>MBR(IMS1    ) CC(   0) GSTSAREA(Y   ) GSTSDB(Y   ) GSTSTRAN(Y   )
 MSG(  N/A) MSGOPT(ABEND ) BMP(  N/A) BMPOPT(ABEND ) FP64STAT(Y)
 AUTEXP(N) REPOTP(IMSRSC) REPONM(IMSRSC_REPOSITORY
     ) LEXPTM(2011.188 16:48:30.07) TGN(IMS     ) VGN(AAA     ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: All the local parameter information from all IMS systems, IMS1 and IMS2, is returned. IMS1 is enabled with the IMSRSC repository, Fast Path 64-bit buffers, MSC TCP/IP generic IMS ID, and VTAM generic IMS ID, while IMS2 does not have any of these parameters enabled. IMS2, which is the command master, returns the IMSplex parameter information.

Example 2 for the QUERY IMS *command*

TSO SPOC input:
```
QRY IMS TYPE(LCLPARM) SHOW(ALL)
```

TSO SPOC output:
(screen 1)
```
MbrName    CC FPBP64STAT  MSG  MSGOPT   BMP  BMPOPT AutoExport RepositoryType
IMS1        0 Y           N/A  ABEND    N/A  ABEND  N          IMSRSC
```

(scrolled to the right screen 2)
```
MbrName   RepositoryName                          LastExportTime
IMS1      IMSRSC_REPOSITORY                       2011.188 16:48:30.07
```

(scrolled to the right screen 3)
```
MbrName   TcpipGenImsID VtamGenName
IMS1      IMS           AAA
```

OM API input:
```
CMD(QRY IMS TYPE(LCLPARM) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.188 23:49:21.164614</statime>
<stotime>2011.188 23:49:21.165415</stotime>
<staseq>C8093A66BE5460C2</staseq>
<stoseq>C8093A66BE8678C2</stoseq>
<rqsttkn1>USRT005 10164921</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>IMS            </kwd>
<input>QRY IMS TYPE(LCLPARM) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="FP64STAT" llbl="FPBP64STAT" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSG" llbl="  MSG" scope="LCL" sort="n" key="0" scroll="yes"
len="4" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="MSGOPT"  llbl=" MSGOPT" scope="LCL" sort="n" key="0"
 scroll="yes" len="6" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="BMP" llbl="  BMP" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="BMPOPT"  llbl=" BMPOPT" scope="LCL" sort="n" key="0"
 scroll="yes" len="6" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="AUTEXP" llbl="AutoExport" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPOTP" llbl="RepositoryType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPONM" llbl="RepositoryName" scope="LCL" sort="n" key="0"
 scroll="yes" len="44" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LEXPTM" llbl="LastExportTime" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TGN" llbl="TcpipGenImsID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="VGN" llbl="VtamGenName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS1    ) CC(   0) MSG( N/A) MSGOPT(ABEND ) BMP(  N/A)
 BMPOPT(ABEND ) FP64STAT(Y) AUTEXP(N) REPOTP(IMSRSC) REPONM(IMSRSC_REPO
SITORY                          ) LEXPTM(2011.188 16:48:30.07) TGN(IMS
    ) VGN(AAA    ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: All the local parameter information from IMS1 is returned. IMS1 is enabled with the IMSRSC repository, Fast Path 64-bit buffers, MSC TCP/IP generic IMS ID, and VTAM generic IMS ID.

Example 3 for the QUERY IMS *command*

TSO SPOC input:

```
QUERY IMS TYPE(LCLPARM) SHOW(ALL)
```

TSO SPOC output:

(screen 1)
```
MbrName   CC   MSG  MSGOPT  BMP  BMPOPT AutoExport RepositoryType
IMS1       0   N/A  ABEND   N/A  ABEND  N          IMSRSC
IMS2       0   N/A  ABEND   N/A  ABEND  N          IMSRSC
IMS3       0   N/A  ABEND   N/A  ABEND  N          IMSRSC
```

(scrolled right to screen 2)
```
MbrName   RepositoryName                          LastExportTime
IMS1      IMSRSC_REPOSITORY                       2011.182 14:15:36.43
IMS2      IMSRSC_REPOSITORY
IMS3      IMSRSC_REPOSITORY                       2011.182 14:11:44.28
```

OM API input:

```
CMD(QRY IMS TYPE(LCLPARM) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.182 21:32:47.301031</statime>
<stotime>2011.182 21:32:47.315601</stotime>
<staseq>C80190AFB35A7900</staseq>
<stoseq>C80190AFB6E9181A</stoseq>
<rqsttkn1>USRT005 10143247</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>IMS            </kwd>
<input>QRY IMS TYPE(LCLPARM) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="FP64STAT" llbl="FPBP64STAT" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSG" llbl="  MSG" scope="LCL" sort="n" key="0" scroll="yes"
len="4" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="MSGOPT"  llbl=" MSGOPT" scope="LCL" sort="n" key="0"
 scroll="yes" len="6" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="BMP" llbl="  BMP" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="BMPOPT"  llbl=" BMPOPT" scope="LCL" sort="n" key="0"
 scroll="yes" len="6" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="AUTEXP" llbl="AutoExport" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPOTP" llbl="RepositoryType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPONM" llbl="RepositoryName" scope="LCL" sort="n" key="0"
 scroll="yes" len="44" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LEXPTM" llbl="LastExportTime" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TGN" llbl="TcpipGenImsID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="VGN" llbl="VtamGenName" scope="LCL" sort="n" key="0"
```

```
                 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS1    ) CC(   0) MSG(  N/A) MSGOPT(ABEND ) BMP(  N/A)
 BMPOPT(ABEND ) AUTEXP(N) REPOTP(IMSRSC) REPONM(IMSRSC_REPOSITORY
                    ) LEXPTM(2011.182 14:15:36.43) </rsp>
<rsp>MBR(IMS3    ) CC(   0) MSG(  N/A) MSGOPT(ABEND ) BMP(  N/A)
 BMPOPT(ABEND ) AUTEXP(N) REPOTP(IMSRSC) REPONM(IMSRSC_REPOSITORY
                    ) LEXPTM(2011.182 14:11:44.28) </rsp>
<rsp>MBR(IMS2    ) CC(   0) MSG(  N/A) MSGOPT(ABEND ) BMP(  N/A)
 BMPOPT(ABEND ) AUTEXP(N) REPOTP(IMSRSC) REPONM(IMSRSC_REPOSITORY
                    ) LEXPTM( ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: This example shows QUERY IMS command output for TYPE(LCLPARM) when IMS is enabled with IMSRSC repository but is not enabled with Fast Path 64-bit buffer manager or generic IMS ID.

Example 4 for the QUERY IMS *command*

TSO SPOC input:
```
QUERY IMS TYPE(LCLPARM) SHOW(REPO)
```

TSO SPOC output:

MbrName	CC	RepositoryType	RepositoryName	LastExportTime
IMS1	0	IMSRSC	IMSRSC_REPOSITORY	2011.213 08:57:00.39
IMS2	0	IMSRSC	IMSRSC_REPOSITORY	2011.213 08:57:08.14

OM API input:
```
CMD(QUERY IMS TYPE(LCLPARM) SHOW(REPO))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.213 15:57:56.076888</statime>
<stotime>2011.213 15:57:56.095315</stotime>
<staseq>C8283FA881558D6E</staseq>
<stoseq>C8283FA885D538EE</stoseq>
<rqsttkn1>USRT011 10085755</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>IMS            </kwd>
<input>QUERY IMS TYPE(LCLPARM) SHOW(REPO) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPOTP" llbl="RepositoryType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPONM" llbl="RepositoryName" scope="LCL" sort="n" key="0"
```

```
 scroll="yes" len="44" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LEXPTM" llbl="LastExportTime" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS2    ) CC(   0) MSG(  N/A) MSGOPT(ABEND ) BMP(  N/A)
 BMPOPT(ABEND ) REPOTP(IMSRSC) REPONM(IMSRSC_REPOSITORY
          ) LEXPTM(2011.213 08:57:08.14) </rsp>
<rsp>MBR(IMS1    ) CC(   0) MSG(  N/A) MSGOPT(ABEND ) BMP(  N/A)
 BMPOPT(ABEND ) REPOTP(IMSRSC) REPONM(IMSRSC_REPOSITORY
          ) LEXPTM(2011.213 08:57:00.39) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The local parameters at IMS for the repository settings are returned. Only the repository information is returned because the SHOW(REPO) keyword is specified.

Related concepts:

⮕ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

⮕ Command keywords and their synonyms (Commands)

QUERY IMSCON commands

Use the QUERY IMSCON commands to display the status and activity of one or more IMS Connect resources.

The TYPE keyword specifies the type of IMS Connect resource to display. The default is TYPE(CONFIG), which displays general IMS Connect information.

The QUERY IMSCON command is processed by every IMS Connect to which OM routes the command, whether or not OM has designated a particular IMS Connect as the command master.

Subsections:
Related reference:

➥ Equivalent IMS Connect WTOR, z/OS, and type-2 commands (Commands)

QUERY IMSCON TYPE(ALIAS) command

Use the QUERY IMSCON TYPE(ALIAS) command to display the status and activity of one or more IMS aliases and the associated ODBMs defined to IMS Connect.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 129
- "Usage notes" on page 129
- "Equivalent WTOR and z/OS commands" on page 130
- "Output fields" on page 130
- "Return, reason, and completion codes" on page 131
- "Examples" on page 132

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►─┬─QUERY─┬─IMSCON─TYPE(ALIAS)─┬─NAME(*)──────────────────────────┬─►
   └─QRY───┘                    │         ┌─,─────────────┐        │
                                │         ▼               │        │
                                └─NAME(──┬─alias_name──┬──┴──)─────┘
                                         └─alias_name*─┘

►─┬────────────────────────────┬─┬──────────────────────┬─►
  │      ┌─,───────────┐       │ │      ┌─,───────┐      │
  │      ▼             │       │ │      ▼         │      │
  └─ODBM(──┬─odbm_name──┬──┴──)─┘ └─SHOW(──┬─ALL────┬─┴──)─┘
           └─odbm_name*─┘                  ├─ODBM───┤
                                           └─STATUS─┘

►─┬─────────────────────────────────┬─►◄
  │          ┌─,──────────┐         │
  │          ▼            │         │
  └─STATUS(──┬─ACTIVE────┬──┴──)────┘
             └─NOTACTIVE─┘
```
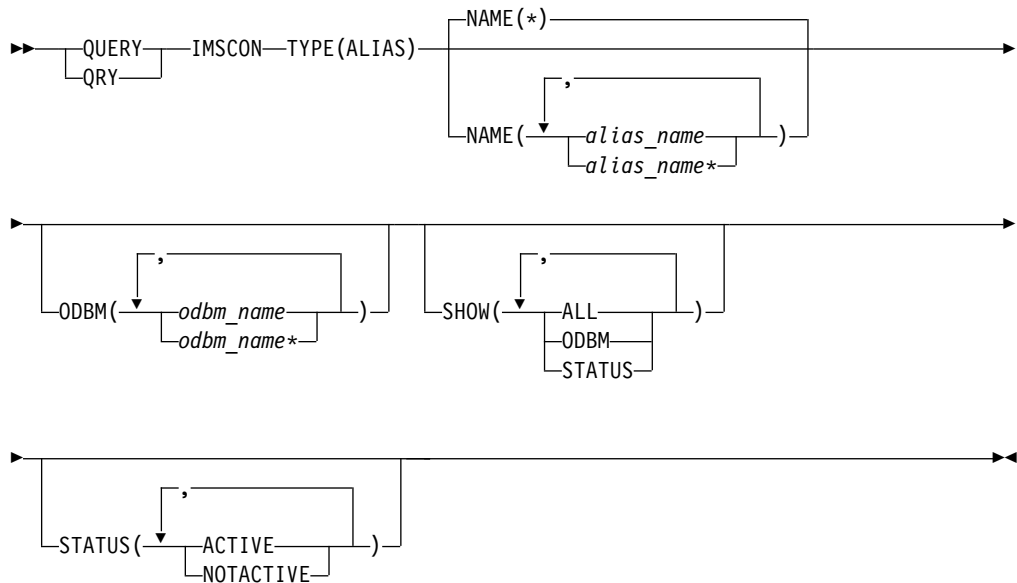
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(ALIAS) command.

NAME
Specifies the name of one or more IMS aliases to be displayed. You can specify a single alias name or a list of alias names separated by commas. Wildcards can be used in the names.

You can specify NAME(*) to display all IMS aliases. NAME(*) is the default.

ODBM
Selects aliases for display that are associated with the specified ODBM name. You can specify a single ODBM name or a list of ODBM names separated by commas. Wildcards can be used in the names.

Only the aliases that are associated with a specified ODBM name are displayed. Aliases that match the NAME parameter, but are not associated with the specified ODBM name, are not displayed.

If the ODBM keyword is specified, ODBM information is displayed even if SHOW(ODBM) is not specified.

SHOW
Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the alias name, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL
Displays all output fields.

ODBM
Displays the name of the ODBM for which the alias is defined.

STATUS
Displays the status of the alias. For a description of the possible alias status returned, see the STATUS keyword in Table 34 on page 130.

STATUS
Displays the ODBM aliases that are in at least one of the specified states. You can specify a single status, or a list of statuses separated by commas, in any order.

ACTIVE
The alias is active.

NOTACTIVE
The alias is not active, either in IMS Connect, ODBM, or both.

If the STATUS keyword is specified, status information is displayed even if SHOW(STATUS) is not specified.

Usage notes

You can issue the QUERY IMSCON TYPE(ALIAS) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Equivalent WTOR and z/OS commands

The following table lists WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(ALIAS) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 33. WTOR and IMS Connect z/OS equivalents for the QUERY IMSCON TYPE(ALIAS) command

QUERY IMSCON TYPE(ALIAS) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
QUERY IMSCON TYPE(ALIAS) NAME(*) SHOW(ALL |*show_parm*)	VIEWIA ALL	QUERY ALIAS NAME(*)
QUERY IMSCON TYPE(ALIAS) NAME(*alias_name*) SHOW(ALL |*show_parm*)	VIEWIA *alias_name*	QUERY ALIAS NAME(*aliasName*)
QUERY IMSCON TYPE(ALIAS) NAME(*alias_name*) ODBM(*odbm_name*)	VIEWIA *alias_name* *odbm_name*	QUERY ALIAS NAME(*aliasName*) ODBM(*odbmName*)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 34. Output fields for the QUERY IMSCON TYPE(ALIAS) command

Short label	Long label	Keyword	Meaning
ALIAS	AliasName	N/A	The alias name of an IMS data store defined to the instance of ODBM. The alias name is always returned.
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 36 on page 132.

Table 34. Output fields for the QUERY IMSCON TYPE(ALIAS) command (continued)

Short label	Long label	Keyword	Meaning
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
ODBM	ODBMName	ODBM	Name of the ODBM associated with the alias.

STT	Status	STATUS	Status of the alias, which is one of the following:
			
			ACTIVE
			   The alias is active.
			
			NOTACTIVE(IMSCON)
			   The alias has been deactivated in IMS Connect by using the STOPIA command (or equivalent).
			
			NOTACTIVE(ODBM)
			   The alias has been deactivated in ODBM by using the ODBM type-2 UPDATE ODBM STOP(CONNECTION) ALIAS command.
			
			NOTACTIVE(IMSCON,ODBM)
			   The alias has been deactivated both in IMS Connect (by using the STOPIA command or equivalent) and in ODBM (by using the ODBM type-2 UPDATE ODBM STOP(CONNECTION) ALIAS command).

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(ALIAS) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 35. Return and reason codes for the QUERY IMSCON TYPE(ALIAS) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(ALIAS) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 36. Completion codes for the QUERY IMSCON TYPE(ALIAS) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(ALIAS) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for QUERY IMSCON TYPE(ALIAS) command

TSO SPOC input:

```
QUERY IMSCON TYPE(ALIAS) NAME(*) SHOW(ALL)
```

TSO SPOC output:

```
AliasName MbrName CC  ODBMName Status
 IMSA     HWS1     0  ODBM1OD  NOTACTIVE(IMSCON)
 IMSA     HWS1     0  ODBM2OD  NOTACTIVE(ODBM)
 IMS1     HWS1     0  ODBM1OD  ACTIVE
 IMS1     HWS1     0  ODBM2OD  NOTACTIVE(IMSCON,ODBM)
 IMS2     HWS1     0  ODBM2OD  ACTIVE
 IMS3     HWS1     0  ODBM3OD  NOTACTIVE(IMSCON)
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(ALIAS) NAME(*) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 15:26:08.249532</statime>
<stotime>2010.298 15:26:08.251073</stotime>
<staseq>C6C82E53FF2BCD37</staseq>
<stoseq>C6C82E53FF8C1EB7</stoseq>
<rqsttkn1>USRT001 10082608</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(ALIAS) NAME(*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="ALIAS" llbl="AliasName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
```

```
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ODBM" llbl="ODBMName" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>ALIAS(IMSA    ) MBR(HWS1            ) CC(   0) ODBM(ODBM1OD )
 STT(NOTACTIVE(IMSCON)) </rsp>
<rsp>ALIAS(IMSA    ) MBR(HWS1            ) CC(   0) ODBM(ODBM2OD )
 STT(NOTACTIVE(ODBM)) </rsp>
<rsp>ALIAS(IMS1    ) MBR(HWS1            ) CC(   0) ODBM(ODBM1OD )
 STT(ACTIVE) </rsp>
<rsp>ALIAS(IMS1    ) MBR(HWS1            ) CC(   0) ODBM(ODBM2OD )
 STT(NOTACTIVE(IMSCON,ODBM)) </rsp>
<rsp>ALIAS(IMS2    ) MBR(HWS1            ) CC(   0) ODBM(ODBM2OD )
 STT(ACTIVE) </rsp>
<rsp>ALIAS(IMS3    ) MBR(HWS1            ) CC(   0) ODBM(ODBM3OD )
 STT(NOTACTIVE(IMSCON)) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: There are two ODBM resources that have aliases currently defined. IMSA and IMS1 are aliases defined to both ODBM1OD and ODBM2OD. IMS2 is an alias defined only to ODBM2OD, and IMS3 is an alias defined only to ODBM3OD. The status NOTACTIVE(IMSCON) indicates that the alias has been deactivated by the IMS Connect STOPIA (or equivalent) command. The status NOTACTIVE(ODBM) indicates that the alias has been deactivated by the UPDATE ODBM STOP(CONNECTION) ALIAS command.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

VIEWIA command (Commands)

QUERY IMSCON TYPE(CLIENT) command

Use the QUERY IMSCON TYPE(CLIENT) command to display the status and activity of one or more active client socket connections on which IMS Connect receives messages from a client.

If the connection is to another instance of IMS Connect or a CICS® subsystem, IMS Connect uses a separate send socket on a different port to send transactions and reply data. The QUERY IMSCON TYPE(CLIENT) command does not display these send client socket connections. To display information about send client socket connections, use the QUERY IMSCON TYPE(SENDCLNT) command.

Subsections:
- "Environment" on page 134
- "Syntax" on page 134
- "Keywords" on page 135
- "Usage notes" on page 138
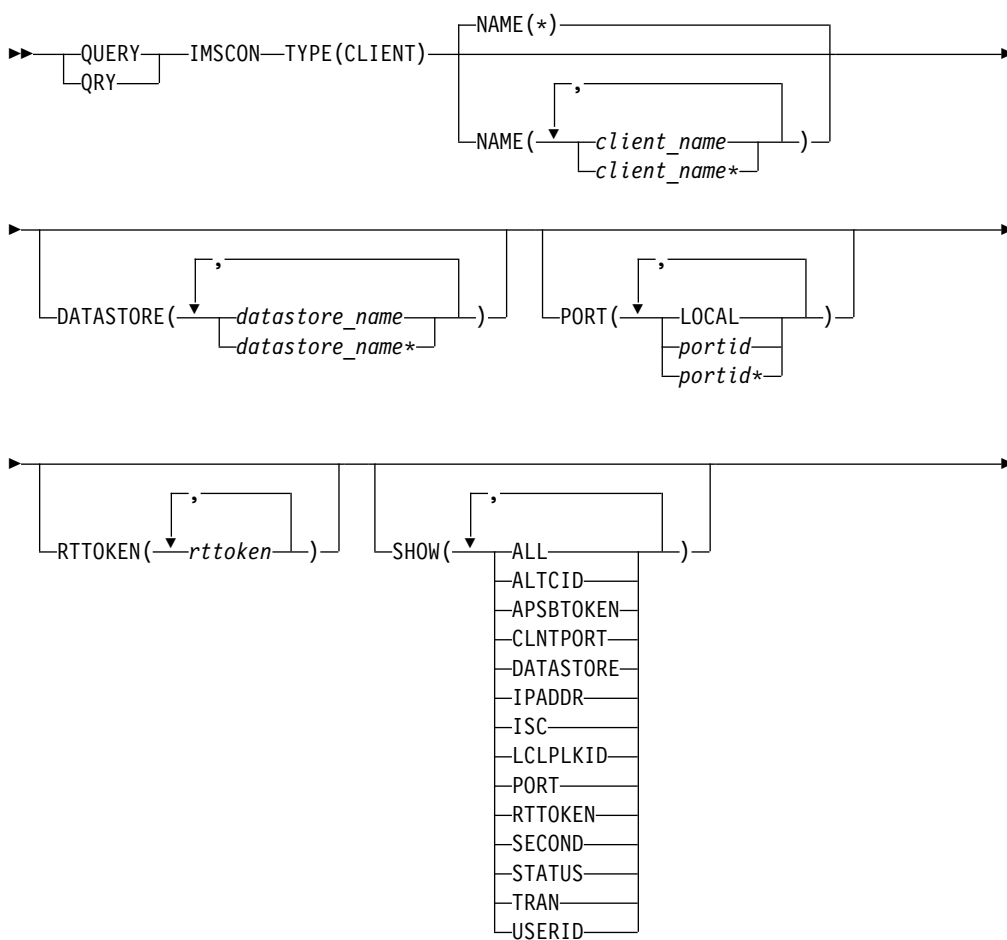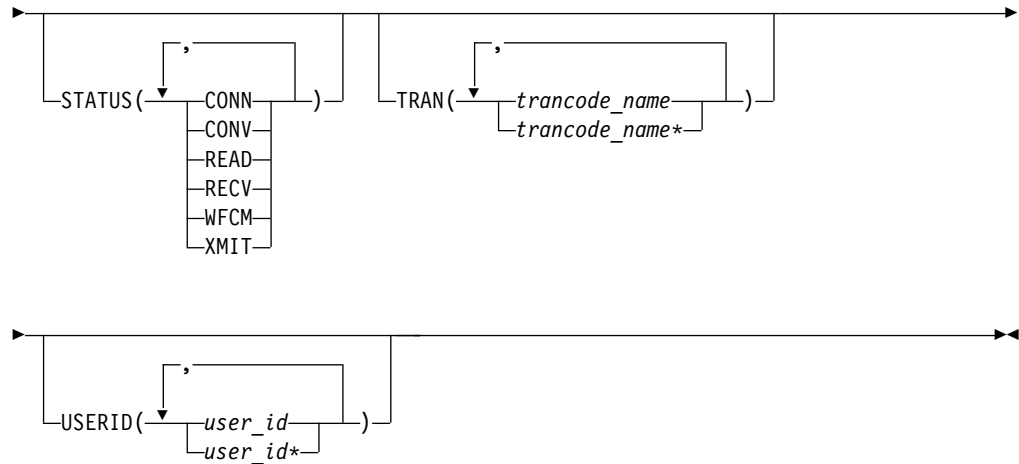- "Equivalent WTOR and z/OS commands" on page 140

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
                                  ┌─NAME(*)──────────────────┐
►►──┬─QUERY─┬──IMSCON──TYPE(CLIENT)─┤                          ├──►
    └─QRY───┘                       │         ┌─,──────────┐   │
                                    └─NAME(──▼─┬─client_name─┬─┴──)─┘
                                              └─client_name*─┘


►──┬──────────────────────────────────┬──┬──────────────────────┬──►
   │          ┌─,──────────────┐       │  │        ┌─,────────┐   │
   └─DATASTORE(──▼─┬─datastore_name─┬─┴──)─┘  └─PORT(──▼─┬─LOCAL──┬─┴──)─┘
                  └─datastore_name*─┘                   ├─portid──┤
                                                        └─portid*─┘


►──┬─────────────────────┬──┬────────────────────┬──►
   │       ┌─,───────┐    │  │       ┌─,────────┐  │
   └─RTTOKEN(──▼─rttoken─┴──)─┘  └─SHOW(──▼─┬─ALL───────┬─┴──)─┘
                                           ├─ALTCID────┤
                                           ├─APSBTOKEN─┤
                                           ├─CLNTPORT──┤
                                           ├─DATASTORE─┤
                                           ├─IPADDR────┤
                                           ├─ISC───────┤
                                           ├─LCLPLKID──┤
                                           ├─PORT──────┤
                                           ├─RTTOKEN───┤
                                           ├─SECOND────┤
                                           ├─STATUS────┤
                                           ├─TRAN──────┤
                                           └─USERID────┘
```

```
                ┌─,──────┐
►─┬──────────────────────┬─┬─────────────────────────┬─────►
  └─STATUS(─▼─┬─CONN─┬──)─┘ └─TRAN(─▼─┬─trancode_name──┬──)─┘
             ├─CONV─┤             │   └─trancode_name*─┘
             ├─READ─┤             └─,─────────────────┘
             ├─RECV─┤
             ├─WFCM─┤
             └─XMIT─┘
```

```
                ┌─,────────┐
►─┬───────────────────────────────┬────────────────────────►◄
  └─USERID(─▼─┬─user_id──┬──)─┘
            └─user_id*─┘
```

Keywords

The following keywords are valid for the QUERY IMSCON TYPE(CLIENT) command.

DATASTORE
Selects clients for display that have a transaction submitted to the specified data store. You can specify a single data store name or a list of data store names separated by commas. Wildcards can be used.

If the DATASTORE keyword is specified, data store information is displayed even if SHOW(DATASTORE) is not specified.

NAME
Specifies the name of one or more clients to be displayed. You can specify a single client name or a list of client names separated by commas. Wildcards can be used.

You can specify NAME(*) to display all clients. NAME(*) is the default.

IMS Connect uses the client name to identify the client socket connection. The name of an IMS Connect client can be provided by the client or, if the client does not provide it, IMS Connect randomly generates the name.

IMS Connect always generates the client name for connections with another IMS Connect instance or a CICS subsystem. In these cases, the first characters of the client name identify the IMS communication type. For example:

ISC ISC communication with a CICS subsystem

MSC MSC communication between IMS systems

OTM OTMA communication between IMS systems

To display the active socket connections for one of the preceding types of IMS communication, you can specify the character identifier followed by a wildcard character. For example, NAME(MSC*), returns all IMS-to-IMS send socket connections for MSC communications.

PORT
Selects clients for display that are active on the specified port. You can specify a single port number or a list of port numbers separated by commas. Wildcards can be used.

To filter on the local port used by the IMS TM Resource Adapter, specify NAME(LOCAL).

The SSL port is displayed with the character "S" appended to the end of the port number. To filter on the SSL port, specify the port number either with or without the character "S" appended to the end of the port number.

The port defined for ODBM use is displayed with the character "D" appended to the end of the port number. To filter on the ODBM port, specify the port number either with or without the character "D" appended to the end of the port number.

The port defined for ISC links to CICS is displayed with the character "C" appended to the end of the port number. To filter on a CICS port, specify the port number either with or without the character "C" appended to the end of the port number.

If the PORT keyword is specified, port information is displayed even if SHOW(PORT) is not specified.

RTTOKEN

Selects clients for display that are currently issuing a RESUME TPIPE call and have a matching RESUME TPIPE token. Each RESUME TPIPE token is up to 16 hexadecimal digits in length. You can specify a single RESUME TPIPE token or a list of RESUME TPIPE tokens separated by commas.

When the RTTOKEN filter is specified, the RESUME TPIPE token is displayed even if SHOW(RTTOKEN) is not specified.

SHOW

Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the client name, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL

Displays all output fields.

ALTCID

Display the alternate client ID specified by the client issuing a Resume TPIPE.

APSBTOKEN

Displays the ODBM APSB token in use by the client for ODBM access.

CLNTPORT

Displays the client port number, which is a random number that TCP/IP generates to represent a connection from a client.

DATASTORE

Displays the data store to which the transaction was submitted.

IPADDR

Displays the IP address being used by the connection of the client to IMS Connect.

ISC

Displays the name of the Intersystem Communication (ISC) resource that this client is associated with. This value is specified on the ID parameter of the ISC statement in the IMS Connect configuration member.

LCLPLKID

Displays the local MSC physical link ID that is using this connection, as specified on the LCLPLKID parameter of the MSC statement in the IMS Connect configuration member. This filter is valid for MSC messages only.

PORT

Displays the port number of the port on which the client is active. If the port is a local port used by the TM Resource Adapter, LOCAL is displayed. If the port is an SSL port, character "S" is appended to the end of the port number. If the port is an ODBM port, character "D" is appended to the end of the port number.

If the port is a CICS port, character "C" is appended to the end of the port number.

RTTOKEN

Displays the Resume TPIPE token of the clients that are issuing a Resume TPIPE.

SECOND

Displays the number of seconds that the client has been in its current status.

STATUS

Displays the status of the client. For a description of the possible status returned, see the STATUS keyword in Table 37 on page 138.

TRAN

Displays the transaction code submitted by the client.

USERID

Displays the user ID passed to IMS Connect.

STATUS

Displays the clients that are in at least one of the specified states. You can specify a single client status, or a list of client statuses separated by commas, in any order.

CONN

Waiting for output from IMS.

CONV

In a conversational state.

READ

Reading an input message from the client.

RECV

Waiting for input from the client (in a receive state).

WFCM

Waiting for confirmation (ACK, NAK, or DEALLOCATE) from the client.

XMIT

Sending data to the client.

If the STATUS keyword is specified, status information is displayed even if SHOW(STATUS) is not specified.

TRAN

Selects clients for display that have a specified transaction submitted to a data store. You can specify a single transaction name or a list of transaction names separated by commas. Wildcards can be used.

If the TRAN keyword is specified, transaction information is displayed even if SHOW(TRAN) is not specified.

USERID
Selects clients for display that have one of the specified user IDs. You can specify a single user ID or a list of user IDs separated by commas. Wildcards can be used.

If the USERID keyword is specified, user ID information is displayed even if SHOW(USERID) is not specified.

Usage notes

You can issue the QUERY IMSCON TYPE(CLIENT) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
Provides a brief description of the output field.

Table 37. Output fields for the QUERY IMSCON TYPE(CLIENT) command

Short label	Long label	Keyword	Meaning
ALTCID	AltCID	ALTCID	The alternate client ID that is specified by a client that is issuing a RESUME TPIPE call.
APTK	ApsbToken	APSBTOKEN	The APSB token for ODBM.
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 39 on page 140.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
CLID	ClientID	N/A	Name of the client. The client name is always returned.
CPORT	ClntPort	CLNTPORT	The client port number, which is a random number that TCP/IP generates to represent a connection from a client.
DS	DataStore	DATASTORE	Data store to which the transaction was submitted by the client.

Table 37. Output fields for the QUERY IMSCON TYPE(CLIENT) command (continued)

Short label	Long label	Keyword	Meaning
IP	IpAddress	IPADDR	The IP address being used by the connection of the client to IMS Connect.
ISC	ISCName	ISC	Displays the name of the Intersystem Communication (ISC) resource that this client is associated with. This value is specified on the ID parameter of the ISC statement in the IMS Connect configuration member.
LPLK	LclPlkID	LCLPLKID	The local MSC physical link ID that is using this connection, as specified on the LCLPLKID parameter of the MSC statement in the IMS Connect configuration member. This field is displayed for MSC messages only.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
PORT	Port	PORT	The port number on which the client is active. For a local port used by TM Resource Adapter, the port number is "LOCAL". For an SSL port, the character "S" is appended to the end of the port number. For an ODBM port, the character "D" is appended to the end of the port number. For a CICS port, the character "C" is appended to the end of the port number.
RTTK	RTToken	RTTOKEN	The Resume TPIPE Token of the client that is issuing a RESUME TPIPE call.
SEC	Second	SECOND	Number of seconds that the client has been in its current state or status.
STT	Status	STATUS	Status or state of the thread of the client, which can be one or more of the following:
			CONN Waiting for output from IMS.
			CONV In a conversational state.
			READ Reading an input message from the client.
			RECV Waiting for input from the client (in other words, in a receive state).
			WFCM Waiting for confirmation (ACK, NAK, or DEALLOCATE) from the client.
			XMIT Sending data to the client.
TRAN	Trancode	TRAN	Transaction code submitted by the client.
UID	UserID	USERID	User ID for the client that is passed to IMS Connect.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(CLIENT) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 38. Return and reason codes for the QUERY IMSCON TYPE(CLIENT) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(CLIENT) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 39. Completion codes for the QUERY IMSCON TYPE(CLIENT) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(CLIENT) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Equivalent WTOR and z/OS commands

There are no equivalent WTOR and z/OS commands because CLIENT is not supported as a resource type for WTOR and z/OS commands for IMS Connect. You can display client information by using the VIEWPORT and QUERY PORT commands, but you cannot use the client name as a search argument.

Examples

Example 1 for QUERY IMSCON TYPE(CLIENT) command

TSO SPOC input:
```
QUERY IMSCON TYPE(CLIENT) NAME(CLIENT*,MSC*) SHOW(ALL)
```

TSO SPOC output:
```
(Screen 1)
 ClientID MbrName CC  Port UserID  LclPlkID Trancode DataStore Second
 CLIENT01 HWS1     0  9999 USRT003          FESTX2   IMS1         2468
```

```
CLIENT02 HWS1    0  9999 USRT003         FESTX2  IMS1      1741
CLIENT03 HWS1    0  9999 USRT003                 IMS1         7
CLIENT09 HWS1    0  9999 USRT003         FESTX2  IMS1      1658
CLIENT12 HWS1    0  9999 USRT002         FESTX2  IMS1        15
CLIENT25 HWS1    0 12345 USRT005         APOL11  IMS1        42
MSC33333 HWS1    0  9999          MSC12                      14
MSC44444 HWS1    0  9999          MSC12                       9
(Screen 2)
ClientID MbrName ClntPort IpAddress            ApsbToken Status
CLIENT01 HWS1        2363 0:0:0:0:0:FFFF:930:6E53         RECV
CLIENT02 HWS1        2277 0:0:0:0:0:FFFF:930:6E53         RECV
CLIENT03 HWS1       53742 0:0:0:0:0:FFFF:930:6E53         RECV WFCM
CLIENT09 HWS1        2280 0:0:0:0:0:FFFF:930:6E53         RECV
CLIENT12 HWS1        2323 0:0:0:0:0:FFFF:930:6E53         RECV
CLIENT25 HWS1        2348 0:0:0:0:0:FFFF:930:6E53         RECV
MSC33333 HWS1        1739 0:0:0:0:0:FFFF:A64:C802         CONN
MSC44444 HWS1        2684 0:0:0:0:0:FFFF:A64:C802         CONN
(Screen 3)
ClientID MbrName AltCID   RTToken
CLIENT01 HWS1
CLIENT02 HWS1
CLIENT03 HWS1     CLIENT99 CBB45E89AAF02E8A
CLIENT09 HWS1
CLIENT12 HWS1
CLIENT25 HWS1
MSC33333 HWS1
MSC44444 HWS1
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(CLIENT) NAME(CLIENT*) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.297 22:46:36.442739</statime>
<stotime>2010.297 22:46:36.444137</stotime>
<staseq>C6C74EEA6CC7302A</staseq>
<stoseq>C6C74EEA6D1E9C6A</stoseq>
<rqsttkn1>USRT001 10154636</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(CLIENT) NAME(CLIENT*,MSC*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
len="8" dtype="CHAR" align="left"skipb="yes" />
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
```

```
|                    scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                   <hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
|                    scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                   <hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
|                    scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                   <hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
|                    len="10" dtype="INT" align="right" skipb="yes" />
|                   <hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
|                    scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
|                   <hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
|                    scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
|                   <hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
|                    scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
|                   <hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
|                    len="9" dtype="CHAR" align="left" skipb="yes" />
|                   <hdr slbl="ALTCID" llbl="AltCID" scope="LCL" sort="n" key="0"
|                    scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                   <hdr slbl="RTTK" llbl="RTToken" scope="LCL" sort="n" key="0"
|                    scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
|                 </cmdrsphdr>
|                 <cmdrspdata>
|                 <rsp>CLID(CLIENT25) MBR(HWS1            ) CC(   0) PORT(12345   )
|                  UID(USRT005 ) TRAN(APOL11 ) DS(IMS1    ) SEC(42) CPORT(2348)
|                  IP(0:0:0:0:0:FFFF:930:6E53) APTK(                ) STT(RECV      )
|                 </rsp>
|                 <rsp>CLID(CLIENT12) MBR(HWS1            ) CC(   0) PORT(9999    )
|                  UID(USRT002 ) TRAN(FESTX2 ) DS(IMS1    ) SEC(15) CPORT(2323)
|                  IP(0:0:0:0:0:FFFF:930:6E53) APTK(                ) STT(RECV      )
|                 </rsp>
|                 <rsp>CLID(CLIENT09) MBR(HWS1            ) CC(   0) PORT(9999    )
|                  UID(USRT003 ) TRAN(FESTX2 ) DS(IMS1    ) SEC(1658) CPORT(2280)
|                  IP(0:0:0:0:0:FFFF:930:6E53) APTK(                ) STT(RECV      )
|                 </rsp>
|                 <rsp>CLID(CLIENT03) MBR(HWS1            ) CC(   0) PORT(9999    )
|                  UID(USRT003 ) TRAN(       ) DS(IMS1    ) SEC(7) CPORT(53742)
|                  IP(0:0:0:0:0:FFFF:930:6E53) STT(RECV WFCM) ALTCID(CLIENT99)
|                  RTTK(CBB45E89AAF02E8A) </rsp>
|                 <rsp>CLID(CLIENT02) MBR(HWS1            ) CC(   0) PORT(9999    )
|                  UID(USRT003 ) TRAN(FESTX2 ) DS(IMS1    ) SEC(1741) CPORT(2277)
|                  IP(0:0:0:0:0:FFFF:930:6E53) APTK(                ) STT(RECV      )
|                 </rsp>
|                 <rsp>CLID(CLIENT01) MBR(HWS1            ) CC(   0) PORT(9999    )
|                  UID(USRT003 ) TRAN(FESTX2 ) DS(IMS1    ) SEC(2468) CPORT(2363)
|                  IP(0:0:0:0:0:FFFF:930:6E53) APTK(                ) STT(RECV      )
|                 </rsp>
|                 <rsp>CLID(MSC33333) MBR(HWS1            ) CC(   0) PORT(9999    )
|                  LPLK(MSC12  ) SEC(14) CPORT(1739) IP(0:0:0:0:0:FFFF:A64:C802)
|                  APTK(                ) STT(CONN      ) </rsp>
|                 <rsp>CLID(MSC44444) MBR(HWS1            ) CC(   0) PORT(9999    )
|                  LPLK(MSC12  ) SEC(9) CPORT(2684) IP(0:0:0:0:0:FFFF:A64:C802)
|                  APTK(                ) STT(CONN      ) </rsp>
|                 </cmdrspdata>
|                 </imsout>
```

| **Explanation:** In the preceding example, there are seven clients, two of which are
| associated with an MSC physical link on port 9999, one client, which is also on
| port 9999, that is issuing a RESUME TPIPE call with an alternate client ID
| specified, and one client on port 12345. Each line of output displays information
| and status specific to each of the clients that are active on the port.

| CLIENT03 is issuing the RESUME TPIPE call and has specified an alternate client
| ID of CLIENT99. To match CLIENT03 with its specific RESUME TPIPE call in
| OTMA, issue the IMS command /DISPLAY TMEMBER, as in the following
| example:

```
/DIS TMEM HWS1 TPIPE ALL OUTPUT
DFS000I    MEMBER/TPIPE        ENQCT      DEQCT       QCT      INPCT STATUS
DFS000I    HWS1
DFS000I    -CLIENT99              0          0          0          0 PMRY
DFS000I    -CLIENT99              0          0          0          0 HLDQ
DFS000I    -CLIENT99          RT CBB45E89AAF02E8A   OPT A     MODE S
```

Then issue the QUERY IMSCON TYPE(CLIENT) command with the RTTOKEN keyword specified with SHOW(ALTCID), as in the following example:

```
QRY IMSCON TYPE(CLIENT) RTTOKEN(CBB45E89AAF02E8A) SHOW(ALTCID RTTOKEN)
```

The client ID that corresponds to the RTTOKEN and the ALTCID is displayed with the RTTOKEN and ALTCID values:

```
ClientID MbrName          CC ALTCID   RTToken
CLIENT03 IMSPLEX1          0 CLIENT99 CBB45E89AAF02E8A
```

Example 2 for QUERY IMSCON TYPE(CLIENT) command

TSO SPOC input:

```
QUERY IMSCON TYPE(CLIENT) USERID(USRT002)
```

TSO SPOC output:

```
 ClientID MbrName CC UserID
 CLIENT12 HWS1     0 USRT002
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(CLIENT) USERID(USRT002) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.297 23:08:15.607128</statime>
<stotime>2010.297 23:08:15.608263</stotime>
<staseq>C6C753C167958165</staseq>
<stoseq>C6C753C167DC71E5</stoseq>
<rqsttkn1>USRT001 10160815</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON         </kwd>
<input>QUERY IMSCON TYPE(CLIENT) USERID(USRT002) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
len="8" dtype="CHAR" align="left"skipb="yes" />
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
```

```
<hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>CLID(CLIENT12) MBR(HWS1            ) CC(   0) UID(USRT002 ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The command requests any client with a user ID of USRT002. CLIENT12 is the only client with this user ID. The SHOW keyword was not specified, but SHOW(USERID) is assumed because the USERID keyword was specified.

Example 3 for QUERY IMSCON TYPE(CLIENT) command

In the following example, four clients active on the system. One client is an OTMA client on port 9999 that has sent an IMS transaction to the IMS1 datastore. Two clients are ISC clients on CICS port 9876 that are associated with the ISC datastore named IMS1CIC4. One client is an MSC client on port 9999 that is associated with the MSC datastore named MSC12.

TSO SPOC input:
```
QUERY IMSCON TYPE(CLIENT) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
ClientID MbrName CC Port     UserID   LclPlkID ISCName
CLIENT01 HWS1    0  9999     USRT003
ISC17C1B HWS1    0  9876C                       IMS1CIC4
ISC5A81F HWS1    0  9876C                       IMS1CIC4
MSC78E94 HWS1    0  9999              MSC12

(screen 2)
ClientID MbrName  Trancode DataStore     Second ClntPort
CLIENT01 HWS1     FESTX2   IMS1              26 4758
ISC17C1B HWS1                               80 1661
ISC5A81F HWS1                               77 1663
MSC78E94 HWS1                               85 1659

(screen 3)
ClientID MbrName  IpAddress              Status
CLIENT01 HWS1     0:0:0:0:0:FFFF:91E:1688 RECV
ISC17C1B HWS1     0:0:0:0:0:FFFF:91E:72F3 RECV
ISC5A81F HWS1     0:0:0:0:0:FFFF:91E:72F3 RECV
MSC78E94 HWS1     0:0:0:0:0:FFFF:91E:72F3 RECV
```

OM API input:
```
CMD ( QUERY IMSCON TYPE(CLIENT) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2012.129 22:12:55.400673</statime>
<stotime>2012.129 22:12:55.402695</stotime>
<staseq>C989DF900E4E141C</staseq>
<stoseq>C989DF900ECC789C</stoseq>
<rqsttkn1>USRT001 10151255</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1     </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(CLIENT) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>CLID(CLIENT01) MBR(HWS1            ) CC(   0) PORT(9999    )
 UID(USRT003 ) TRAN(FESTX2  ) DS(IMS1    ) SEC(26) CPORT(4758)
 IP(0:0:0:0:0:FFFF:91E:1688) STT(RECV     ) </rsp>
<rsp>CLID(MSC78E94) MBR(HWS1            ) CC(   0) PORT(9999    )
 LPLK(MSC12   ) SEC(85) CPORT(1659) IP(0:0:0:0:0:FFFF:91E:72F3)
 STT(RECV     ) </rsp>
<rsp>CLID(ISC5A81F) MBR(HWS1            ) CC(   0) PORT(9876C   )
 ISC(IMS1CIC4) SEC(77) CPORT(1663) IP(0:0:0:0:0:FFFF:91E:72F3) STT(RECV
      ) </rsp>
<rsp>CLID(ISC17C1B) MBR(HWS1            ) CC(   0) PORT(9876C   )
```

```
   ISC(IMS1CIC4) SEC(80) CPORT(1661) IP(0:0:0:0:0:FFFF:91E:72F3) STT(RECV
      ) </rsp>
</cmdrspdata>
</imsout>
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

QUERY IMSCON TYPE(CONFIG) command

Use the QUERY IMSCON TYPE(CONFIG) command to display the status and activity of IMS Connect. Unlike the similar WTOR command VIEWHWS or the z/OS command QUERY MEMBER TYPE(IMSCON), individual resources such as PORT and DATASTORE are not displayed with this command.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 147
- "Usage notes" on page 150
- "Equivalent WTOR and z/OS commands" on page 150
- "Output fields" on page 150
- "Return and reason codes" on page 154
- "Completion codes" on page 155
- "Examples" on page 155

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
          ►►─┬─QUERY─┬─IMSCON─TYPE(CONFIG)──────────────────────────────────────────►◄
            └─QRY───┘                      │       ┌─,──────┐              │
                                           └─SHOW(─▼─┬─ALL───────┬─)─┘
                                                     ├─ADAPTER───┤
                                                     ├─CASCADE───┤
                                                     ├─CM0ATOQ───┤
                                                     ├─ICONID────┤
                                                     ├─IDLETO────┤
                                                     ├─IPADDR────┤
                                                     ├─MAXLSSSZ──┤
                                                     ├─MAXSOC────┤
                                                     ├─MULTIRTP──┤
                                                     ├─NUMSOC────┤
                                                     ├─ODBMAC────┤
                                                     ├─ODBMIPMEM─┤
                                                     ├─ODBMTMEM──┤
                                                     ├─ODBMTO────┤
                                                     ├─ODRACFST──┤
                                                     ├─PSWDMC────┤
                                                     ├─RACF──────┤
                                                     ├─RECORDER──┤
                                                     ├─RRS───────┤
                                                     ├─SMEM──────┤
                                                     ├─TCPIPQ────┤
                                                     ├─TIMEOUT───┤
                                                     ├─TMRACFST──┤
                                                     ├─UIDAGE────┤
                                                     ├─UIDCACHE──┤
                                                     ├─VERSION───┤
                                                     ├─WARNINC───┤
                                                     └─WARNSOC───┘
```
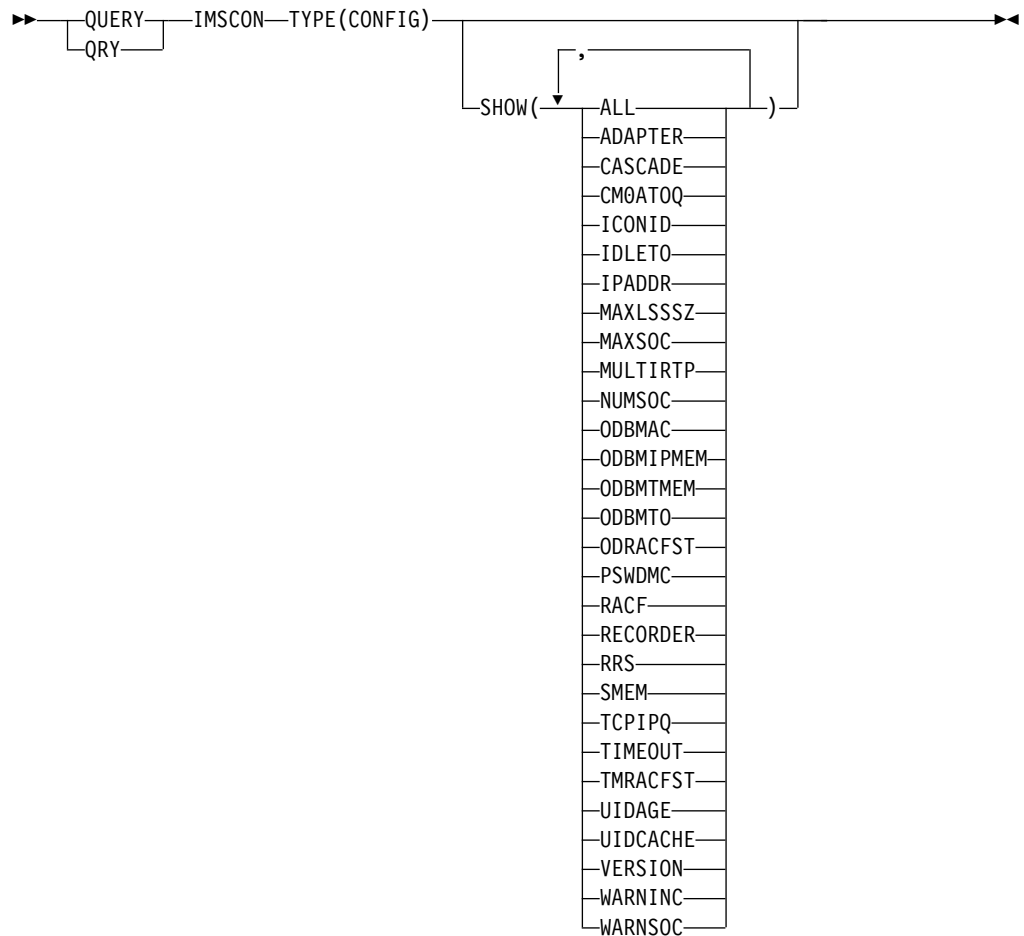
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(CONFIG) command.

SHOW

Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL

Displays all output fields.

ADAPTER

Displays whether XML adapter support is enabled, the maximum number of XML converters that this instance of IMS Connect can load, and the current number of XML converters that this instance of IMS Connect has loaded.

CM0ATOQ

Display the name of the OTMA CM0 ACK timeout queue, as specified by the CM0ATOQ= keyword in the IMS Connect configuration file.

CASCADE

Displays the IMS Connect default setting for whether data store

connections from this IMS Connect support the cascading of global z/OS Resource Recovery Services transactions to IMS TM data stores that are located on different z/OS images (LPARs) than IMS Connect. The IMS Connect default for cascaded transaction support is set either by the CASCADE= parameter on the HWS= statement in the HWSCFG*xx* member of the IMS PROCLIB data set or by using the IMS type-2 command UPDATE IMSCON TYPE(CONFIG) SET(CASCADE()).

ICONID
Display the name of the IMS Connect, as defined in the ID substatement of the HWS configuration statement in the HWSCFGxx configuration member.

IDLETO
Displays the global setting for the amount of time open socket connections can remain idle in RECV state before they are terminated due to inactivity. The global settings serves as the default IDLETO value for ports that are defined by the PORT or PORTID parameters in the TCPIP configuration statement for communication with IMS TM.

The value displayed is the value of the IDLETO= parameter in the TCPIP statement in the IMS Connect configuration file.

The value shown is hundredths of seconds. Timeout values can be from 0 to 2 147 483 647 (X'7FFFFFFF'). A value of 0 means this timeout function is disabled, which prevents inactive connections from timing out.

This global-level idle timeout value can be overridden by the port-level idle timeout level.

IPADDR
Displays the IP address for this instance of IMS Connect. Client application programs use this IP address to connect to IMS Connect.

MAXLSSSZ
Displays the maximum language structure segment size. This value is passed to the XML converter when it is called.

MAXSOC
Displays the maximum total number of sockets that this instance of IMS Connect can open. The maximum number of physical connections that can be made is the MAXSOC= value minus the number of ports, because IMS Connect uses one socket on each port for listening.

MULTIRTP
Displays the current specification of the MULTIRTP keyword for this IMS Connect instance. The value displayed is the default value for all DATASTORE connections that do not specify a MULTIRTP value themselves. The MULTIRTP value determines whether the OTMA tpipes support multiple active resume tpipe requests.

NUMSOC
Displays the current number of sockets.

ODBMAC
Displays the ODBM auto connection value, which indicates whether the instance of IMS Connect identified in the ID field is configured to automatically connect to the instances of ODBM in the IMSplex in which IMS Connect is a member.

ODBMIPMEM
Displays the ODBM IMSplex member name, which is the name of the IMS Connect that SCI uses to manage communications between IMS Connect and ODBM.

ODBMTMEM
Displays the ODBM IMSplex TMEMBER name, which is the name of the SCI that manages the communication between IMS Connect and ODBM.

ODBMTO
Displays the ODBM timeout value, which is the time interval that IMS Connect waits before disconnecting a client application program that uses the Open Database architecture.

ODRACFST
Displays the RACF statistics option value that is used to enable or disable RACF statistics when connections to IMS DB through ODBM are authenticated.

PSWDMC
Indicates whether mixed-case password support is currently enabled for this instance of IMS Connect.

RACF
Indicates whether RACF support is currently enabled for this instance of IMS Connect.

RECORDER
Indicates whether the line trace data set is open or closed.

RRS
Displays the following z/OS Resource Recovery Services (RRS) information for this instance of IMS Connect:
- Whether RRS is enabled in the HWS configuration file
- The current status of RRS

SMEM
Displays the OTMA super member name.

TCPIPQ
Displays the maximum number of queued connection requests without assigned sockets that this IMS Connect can maintain.

TIMEOUT
Displays the interval of time IMS Connect waits before disconnecting a client application program after either of the following situations:
- Waiting for a response for the client from IMS
- Waiting for data from the client after the client opens a socket connection

TMRACFST
Displays the RACF statistics option value that is used to enable or disable RACF statistics when connections to IMS TM are authenticated.

UIDAGE
Displays the RACF user ID aging value in seconds. When the RACF user ID aging value is reached, IMS Connect refreshes the user ID before it processes the next input message for that user ID. Valid values are from 0 to 2,147,483,647 seconds. UIDAGE is only effective when RACF user ID caching is enabled.

UIDCACHE
Displays whether RACF user ID caching is used when RACF authentication is enabled.

VERSION
Displays the version of this instance of IMS Connect.

WARNINC
Displays the warning incremental percentage, which is a decimal value between 1 - 50. After the warning level (WARNSOC value in the TCPIP configuration statement) has been reached, IMS Connect will reissue an HWSS0772W message each time the number of sockets increases by the warning incremental percentage.

WARNSOC
Displays the warning level as a percentage of the maximum sockets limit (MAXSOC= value in the TCPIP configuration statement), which is a decimal value between 50 - 99. When the number of sockets increases to this warning level, IMS Connect issues an HWSS0772W message.

Usage notes

You can issue the QUERY IMSCON TYPE(CONFIG) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Equivalent WTOR and z/OS commands

The following table lists WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(CONFIG) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 40. WTOR and IMS Connect z/OS equivalents for the QUERY IMSCON TYPE(CONFIG) command

QUERY IMSCON TYPE(CONFIG) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
QUERY IMSCON TYPE(CONFIG) SHOW(ALL │ *show_parm*)	VIEWHWS	QUERY MEMBER TYPE(IMSCON) SHOW(ALL)

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be

generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
Provides a brief description of the output field.

Table 41. Output fields for the QUERY IMSCON TYPE(CONFIG) command

Short label	Long label	Keyword	Meaning
ADAP	Adapter	ADAPTER	Whether XML adapter support is enabled or disabled: **Y** XML adapter support is enabled. **N** XML adapter support is not enabled.
CASCD	Cascade	CASCADE	Shows the IMS Connect default specification for support for cascading global RRS transactions to IMS TM data stores that are located on different LPARs than IMS Connect. This specification was set either by the CASCADE= parameter of the HWS= statement in the HWSCFG*xx*member of the IMS PROCLIB data set or by the type-2 UPDATE IMSCON TYPE(CONFIG) SET(CASCADE()) command. The value shown here can be overridden in the definition of an individual data store connection. One of the following values is displayed: **Y** Indicates that when IMS Connect and the IMS data store are on different z/OS images (LPARs), IMS Connect enables cascaded transaction support for all data store connections that do not have a different CASCADE value in their definitions. **N** Indicates that by default, cascaded transaction support is disabled for data store connections that do not have a different CASCADE value in their definitions.
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 43 on page 155.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
CM0ATOQ	Cm0Atoq	CM0ATOQ	The name of the OTMA CM0 ACK timeout queue, as specified by the CM0ATOQ= keyword in the IMS Connect configuration file.
ICID	IconID	ICONID	The name of the IMS Connect, as defined in the ID substatement of the HWS configuration statement in the HWSCFGxx configuration member.
IP	IpAddress	IPADDR	The IP address for this instance of IMS Connect. Client application programs use this IP address to connect to IMS Connect.
ITO	IdleTO	IDLETO	Timeout value for an idle socket connection in RECV state.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.

Table 41. Output fields for the QUERY IMSCON TYPE(CONFIG) command (continued)

Short label	Long label	Keyword	Meaning
MCNV	MaxCvrt	ADAPTER	The maximum number of XML converters that this IMS Connect instance can load concurrently.
MLSS	MaxLSSSz	MAXLSSSZ	The maximum language structure segment size, which is passed to the XML converters.
MRTP	MultiRTP	MULTIRTP	The current specification of MULTIRTP for this IMS Connect instance. The value displayed here can be overridden by the MULTIRTP specification on the DATASTORE connection definition. Possible values include:
			⌂ If the value is blank, the MULTIRTP support is determined either by the DATASTORE connection definition or by the OTMA client descriptor.
			Y By default, all client bid requests from this IMS Connect instance request an OTMA tpipe that supports multiple active resume tpipe requests.
			N By default, all client bid requests from this IMS Connect instance request an OTMA tpipe that supports only a single active resume tpipe request.
MSOC	MaxSoc	MAXSOC	The maximum total number of sockets that this instance of IMS Connect can open. The maximum number of physical connections that can be made is the MAXSOC= value minus the number of ports, because IMS Connect uses one socket on each port for listening.
NCNV	NumCvrt	ADAPTER	The number of XML converters currently loaded in this instance of IMS Connect.
NSOC	NumSoc	NUMSOC	The current number of sockets.
OAC	ODBMAC	ODBMAC	The ODBM auto connection value, which indicates whether the instance of IMS Connect identified in the ID field is configured to automatically connect to the instances of ODBM in the IMSplex in which IMS Connect is a member. Y ODBM auto connection is enabled. N ODBM auto connection is not enabled.
OIMEM	ODBMIpMem	ODBMIPMEM	The ODBM IMSplex member name, which is the name of the IMS Connect that SCI uses to manage communications between IMS Connect and ODBM.
ORSTAT	ODBMRacfStat	ODRACFST	Whether RACF statistics are recorded and updated when IMS Connect issues the RACF command RACROUTE REQUEST=VERIFY to authenticate ODBM client connections to IMS DB. One of the following values is displayed:
			Y RACF statistics and messages are enabled and updated for the execution of the RACF RACROUTE REQUEST=VERIFY call.
			N RACF statistics and messages are not enabled and not updated for the execution of the RACF RACROUTE REQUEST=VERIFY call.
OTIMO	ODBMTO	ODBMTO	The ODBM timeout value, which is the time interval that IMS Connect waits before disconnecting a client application program that uses the Open Database architecture.

Table 41. Output fields for the QUERY IMSCON TYPE(CONFIG) command (continued)

Short label	Long label	Keyword	Meaning
OTMEM	ODBMTMem	ODBMTMEM	The ODBM IMSplex TMEMBER name, which is the name of the SCI that manages the communication between IMS Connect and ODBM.
PMC	PswdMc	PSWDMC	Whether mixed-case password support is currently enabled for this instance of IMS Connect. One of the following values is displayed: **Y** Mixed-case password support is enabled. **N** Mixed-case password support is not enabled. **R** Mixed-case password support depends on the mixed-case password specification in RACF.
RACF	Racf	RACF	Whether RACF support is currently enabled for this instance of IMS Connect. One of the following values is displayed: **Y** RACF support is enabled. **N** RACF support is not enabled.
RCDR	Recorder	RECORDER	Whether the line trace data set is open or closed. One of the following values is displayed: **Y** The line trace data set is open. **N** The line trace data set is closed.
RRS	RRS	RRS	Whether RRS is enabled in the HWS configuration file: **Y** RRS support is enabled. **N** RRS support is not enabled.
RSTT	RRSStat	RRS	The current status of RRS: **ACTIVE** IMS Connect restart with RRS has completed. **NOTACTIVE** IMS Connect has not registered with RRS. **REGISTERED** IMS Connect has registered with RRS.
SMEM	SMem	SMEMBER	The OTMA super member name.
TCPQ	TCPIPQ	TCPIPQ	The maximum number of queued connection requests without assigned sockets.
TIMO	TimeOut	TIMEOUT	The interval of time that IMS Connect waits before disconnecting a client application program after either of the following situations: • Waiting for a response for the client from IMS • Waiting for data from the client after the client opens a socket connection
TRSTAT	TMRacfStat	TMRACFST	Whether RACF statistics are recorded and updated when IMS Connect issues the RACF command RACROUTE REQUEST=VERIFY to authenticate OTMA client connections to IMS TM. One of the following values is displayed: `Y` RACF statistics and messages are enabled and updated for the execution of the RACF RACROUTE REQUEST=VERIFY call. `N` RACF statistics and messages are not enabled and not updated for the execution of the RACF RACROUTE REQUEST=VERIFY call.

Short label	Long label	Keyword	Meaning
UIDA	UidAge	UIDAGE	The RACF user ID aging value in seconds. When the RACF user ID aging value is reached, IMS Connect refreshes the user ID before it processes the next input message for that user ID. Valid values are from 0 to 2,147,483,647 seconds. UIDAGE is effective only when RACF user ID caching is enabled.
UIDC	UidCache	UIDCACHE	Whether RACF user ID caching is used when RACF authentication is enabled: **Y** IMS Connect will cache the RACF user IDs when RACF authentication is enabled. **N** IMS Connect will use the old session level caching.
VER	Version	VERSION	The version of this instance of IMS Connect.
WINC	WarnInc	WARNINC	The warning incremental percentage, which is a decimal value between 1 - 50. After the warning level (WARNSOC value in the TCPIP configuration statement) has been reached, IMS Connect reissues an HWSS0772W message each time the number of sockets increases by the warning incremental percentage.
WSOC	WarnSoc	WARNSOC	The warning level as a percentage of the maximum sockets limit (MAXSOC= value in the TCPIP configuration statement), which is a decimal value between 50 - 99. When the number of sockets increases to this warning level, IMS Connect issues an HWSS0772W message.

Return and reason codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(CONFIG) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 42. Return and reason codes for the QUERY IMSCON TYPE(CONFIG) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(CONFIG) command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See the "Completion codes" on page 155 table for details.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Completion codes

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 43. Completion codes for the QUERY IMSCON TYPE(CONFIG) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(CONFIG) command completed successfully for the resources.

Examples

Example 1 for QUERY IMSCON TYPE(CONFIG) command

TSO SPOC input:
```
QUERY IMSCON TYPE(CONFIG) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
MbrName            CC Version IconID    IpAddress        MaxSoc      TimeOut NumSoc
HWS1                0 V14     HWS1      127.0.0.1            50 2147483647      4

(screen 2)
MbrName            WarnSoc WarnInc     TCPIPQ UidCache      UidAge RACF PswdMc RRS
HWS1                   80        5 2147483647 Y        2147483647 N    N      N

(screen 3)
MbrName            RRSStat    Recorder SMem    Cm0Atoq  Adapter  MaxCvrt NumCvrt
HWS1               REGISTERED N        SM01             Y            400      15

(screen 4)
MbrName            ODBMAC      ODBMTO ODBMIpMem        ODBMTMem        MaxLSSSz
HWS1               Y            18000 HWS1            PLEX1              32767

(screen 5)
MbrName            MultiRTP Cascade ODBMRacfStat    IdleTO TMRacfStat
HWS1               N        N       N                 5000 N
```

OM API input:
```
CMD ( QUERY IMSCON TYPE(CONFIG) SHOW(ALL) )
```

OM API output:
```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime> 2014.118 22:49:29.653223</statime>
<stotime> 2014.118 22:49:29.663453</stotime>
<staseq>C9F23082045267A5</staseq>
<stoseq>C9F23082046F18A5</stoseq>
<rqsttkn1>USRT002 10142849</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
```

```
<cmd>
<master>HWS1     </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(CONFIG) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="VER" llbl="Version" scope="LCL" sort="n" key="0"
 scroll="yes" len="3" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ICID" llbl="IconID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="15" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSOC" llbl="MaxSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TIMO" llbl="TimeOut" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="int" align="right" skipb="yes" />
<hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="WSOC" llbl="WarnSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="2" dtype="INT" align="right" skipb="yes" />
<hdr slbl="WINC" llbl="WarnInc" scope="LCL" sort="n" key="0"
 scroll="yes" len="2" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TCPQ" llbl="TCPIPQ" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="UIDC" llbl="UidCache" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="UIDA" llbl="UidAge" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="RACF" llbl="RACF" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PMC" llbl="PswdMc" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RRS" llbl="RRS" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RSTT" llbl="RRSStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RCDR" llbl="Recorder" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SMEM" llbl="SMem" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CM0ATOQ" llbl="Cm0Atoq" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ADAP" llbl="Adapter" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MCNV" llbl="MaxCvrt" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="NCNV" llbl="NumCvrt" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="OAC" llbl="ODBMAC" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="OTIMO" llbl="ODBMTO" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="OIMEM" llbl="ODBMIpMem" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="OTMEM" llbl="ODBMTMem" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MLSS" llbl="MaxLSSSz" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="MRTP" llbl="MultiRTP" scope="LCL" sort="n" key="0"
```

```
          scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
         <hdr slbl="CASCD" llbl="Cascade" scope="LCL" sort="n" key="0"
          scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
         <hdr slbl="ORSTAT" llbl="ODBMRacfStat" scope="LCL" sort="n" key="0"
          scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
         <hdr slbl="ITO" llbl="IdleTO" scope="LCL" sort="n" key="0"
          scroll="yes"len="10" dtype="INT" align="right" skipb="yes" />
         <hdr slbl="TRSTAT" llbl="TMRacfStat" scope="LCL" sort="n" key="0"
          scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
         </cmdrsphdr>
         <cmdrspdata>
         <rsp>MBR(HWS1            ) CC(   0) VER(V14) ICID(HWS1    )
          IP(127.0.0.1) MSOC(   50) TIMO(2147483647) NSOC(    4) WSOC(80)
          WINC( 5) TCPQ(2147483647) UIDC(Y) UIDA(2147483647) RACF(N) PMC(N) RRS(N)
          RSTT(REGISTERED) RCDR(N) SMEM(SM01) CM0ATOQ(         ) ADAP(Y)
          MCNV(400) NCNV(15) OAC(Y) OTIMO(    18000) OIMEM(HWS1           )
           OTMEM(PLEX1          ) MLSS(32767) MRTP(N) CASCD(N) ORSTAT(N)
           ITO(5000) TRSTAT(N)
         </rsp>
         </cmdrspdata>
         </imsout>
```

Explanation: This command shows general status related to IMS Connect.

Example 2 for QUERY IMSCON TYPE(CONFIG) command

In the following example, the ODRACFST keyword is used to display the current value of the ODRACFST= option, which is defined in the ODACCESS statement of the HWSCFGxx member.

TSO SPOC input:

```
QRY IMSCON TYPE(CONFIG) SHOW(ODRACFST)
```

TSO SPOC output:

```
MbrName           CC ODBMRacfStat
HWS1               0 Y
```

OM API input:

```
CMD(QRY IMSCON TYPE(CONFIG) SHOW(ALL))
```

OM API output:

```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2017.111 00:29:35.253108</statime>
<stotime>2017.111 00:29:35.253390</stotime>
<staseq>D26B2A2D6B674CF1</staseq>
<stoseq>D26B2A2D6B78E8F1</stoseq>
<rqsttkn1>USRT011 10172935</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QRY IMSCON TYPE(CONFIG) SHOW(ODRACFST) </input>
```

```
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ORSTAT" llbl="ODBMRacfStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(HWS1            ) CC(   0) ORSTAT(Y) </rsp>
</cmdrspdata>
</imsout>
```

Example 3 for QUERY IMSCON TYPE(CONFIG) command

In the following example, the IMS Connect instance has a global-level IDLETO
value of 0. If any port does not define its own IDLETO value, it will use this
global-level IDLETO value, and its port connection idle timeout function is
disabled.

TSO SPOC input:
```
QUERY IMSCON TYPE(CONFIG) SHOW(IDLETO)
```

TSO SPOC output:
```
Response for: QRY IMSCON TYPE(CONFIG) SHOW(IDLETO)
MbrName         CC      IdleTO
HWS1             0           0
```

OM API input:
```
CMD(QRY IMSCON TYPE(CONFIG) SHOW(IDLETO))
```

OM API output:
```
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ITO" llbl="IdleTO" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(HWS1            ) CC(   0) ITO(0) </rsp>
</cmdrspdata>
```
Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

➡ VIEWHWS command (Commands)

➡ IMS Connect QUERY MEMBER command (Commands)

QUERY IMSCON TYPE(CONVERTER) command

Use the QUERY IMSCON TYPE(CONVERTER) command to display the status of IMS Connect XML converters.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 160
- "Equivalent WTOR and z/OS commands" on page 160
- "Output fields" on page 160
- "Return, reason, and completion codes" on page 161
- "Examples" on page 162

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►─┬─QUERY─┬─IMSCON─TYPE(CONVERTER)────────────────────────────────►

   └─QRY───┘


►──┬──────────────────────────────────┬──┬─────────────────────────┬──►◄
   │          ┌─,────────────────┐     │  │         ┌─,──────────┐   │
   │          │     ┌─*─────────┐ │     │  │         │  ┌─ALL──────┐ │
   └─NAME─(───▼─────┼─converter_name──┼─┴─)─┘  └─SHOW(─▼──┼─TIMESTAMP─┼─┴─)─┘
                    └─converter_name*─┘                  └─USECOUNT──┘
```
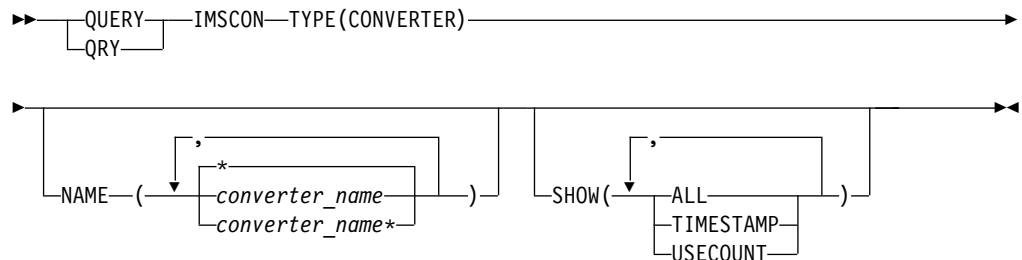
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(CONVERTER) command.

NAME
    Specifies one or more XML converter names to display. You can specify one or more full and wildcard-prefix names in a comma-separated list. If you do not specify this keyword and a value, all XML converters are displayed.

SHOW
    Specifies which optional output fields to display in addition to the standard output fields. You can specify the following additional output fields:

    ALL
        Display all available output fields.

    TIMESTAMP
        Display the timestamp of the last successful load (TimeCreate) and

execution (TimeAccess) for each converter. The timestamp is local time. Timestamps are displayed in the following format:

`YYYY.JJJ HH:MM:SS.TH`

YYYY Year

JJJ   Julian day

HH   Hour

MM   Minute

SS   Second

TH   Tenths and hundredths of a second

USECOUNT
>The number of times the converter was executed since it was loaded.

Usage notes

You can issue the QUERY IMSCON TYPE(CONVERTER) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

The QUERY IMSCON TYPE(CONVERTER) command is only processed by IMS Connect.

Equivalent WTOR and z/OS commands

The QUERY IMSCON TYPE(CONVERTER) type-2 command does not have an equivalent WTOR or z/OS command.

Output fields

Short label
>Contains the short label that is generated in the XML output.

Long label
>Contains the column heading displayed on the TSO SPOC screen.

Keyword
>Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) is shown for output fields that are always returned. *error* is shown for output fields that are returned only in the case of an error.

Meaning
>Provides a brief description of the output field.

Table 44. Output fields for the QUERY IMSCON TYPE(CONFIG) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned.

Table 44. Output fields for the QUERY IMSCON TYPE(CONFIG) command (continued)

Short label	Long label	Keyword	Meaning
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
CVRTR	Converter	N/A	Name of the XML converter.
MBR	MbrName	N/A	Name of the IMS Connect member that built the command output.
TMAC	TimeAccess	TIMESTAMP	The date and time that this XML converter was last executed. This field is returned only if SHOW(TIMESTAMP) is specified in the command. The timestamp is local time. The timestamp is in the following format: `YYYY.JJJ HH:MM:SS.TH` **YYYY** Year **JJJ** Julian day **HH** Hour **MM** Minute **SS** Second **TH** Tenths and hundredths of a second
TMCR	TimeCreate	TIMESTAMP	The date and time that this XML converter was loaded. This field is only returned if SHOW(TIMESTAMP) is specified in the command. The field format is the same as for the TMAC output field and the timestamp is local time.
UCNT	UseCount	USECOUNT	The number of times that this XML converter was executed since the last time it was loaded. This field is returned only if SHOW(USECOUNT) is specified in the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(CONVERTER) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 45. Return and reason codes for the QUERY IMSCON TYPE(CONVERTER) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(CONVERTER) command completed successfully. The output contains a completion code for each resource.
X'0C000004'	X'00001010′	No matches were found for the specified NAME value or values.
X'0C000008'	X'00002004'	The command was not processed because one of the specified keywords was invalid or mutually exclusive keywords were specified.

Table 45. Return and reason codes for the QUERY IMSCON TYPE(CONVERTER) command (continued)

Return code	Reason code	Meaning
X'0C000008'	X'0000200C'	No resources were found.
X'0C00000C'	X'00002014'	An invalid character was used in one of the specified NAME values.
X'0C00000C'	X'00003000'	The command failed for some of the specified resources. Check the completion codes for each resource in the output.
X'0C00000C'	X'00003004'	The command failed for all of the specified resources. Check the completion codes for each resource in the output.
X'0C000014'	X'00005004'	IMS Connect could not obtain storage for the command response data.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 46. Completion codes for the QUERY IMSCON TYPE(CONVERTER) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(CONVERTER) command completed successfully for the resources.
10	NO RESOURCES FOUND	The OM command processing client did not find resources that match the requested names. The resource name might be spelled incorrectly or the resource might be inactive.

Examples

Example 1 for QUERY IMSCON TYPE(CONVERTER) command

The following example shows the TSO SPOC output for an IMS Connect system with two active XML converters.

TSO SPOC input:
```
QUERY IMSCON TYPE(CONVERTER) SHOW(ALL)
```

Output:
```
Converter MbrName CC UseCount TimeAccess          TimeCreate
IMSCVRTD  HWS1     0       30 2012.250 13:15:55.88 2012.250 13:10:22.01
IMSPHBKD  HWS1     0        5 2012.251 22:30:42.28 2012.251 22:18:47.33
```

XML output from the OM API interface:
```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2012.251 22:38:47.382725</statime>
<stotime>2012.251 22:38:47.383079</stotime>
<staseq>CA2348E2602C532A</staseq>
```

```
<stoseq>CA2348E260427D2A</stoseq>
<rqsttkn1>USER002 10153847</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USER002 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(CONVERTER) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="CVRTR" llbl="Converter" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="UCNT" llbl="UseCount" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TMAC" llbl="TimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>CVRTR(IMSCVRTD) MBR(HWS1            ) CC(   0) UCNT(30)
 TMAC(2012.250 13:15:55.88) TMCR(2012.250 13:10:22.01) </rsp>
<rsp>CVRTR(IMSPHBKD) MBR(HWS1            ) CC(   0) UCNT(5)
 TMAC(2012.251 22:30:42.28) TMCR(2012.251 22:18:47.33) </rsp>
</cmdrspdata>
</imsout>
```

QUERY IMSCON TYPE(DATASTORE) command

Use the QUERY IMSCON TYPE(DATASTORE) command to display the status and activity of one or more data stores defined to IMS Connect.

Subsections:
- "Environment"
- "Syntax" on page 164
- "Keywords" on page 164
- "Usage notes" on page 166
- "Equivalent WTOR and z/OS commands" on page 167
- "Output fields" on page 167
- "Return, reason, and completion codes" on page 170
- "Examples" on page 171

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
                                              ┌─NAME(*)────────────────────────┐
►►──┬─QUERY─┬──IMSCON──TYPE(DATASTORE)──┼────────────────────────────────┼──►
    └─QRY───┘                            │        ┌─,────────────────┐    │
                                         └─NAME(──▼─┬─datastore_name──┬─)──┘
                                                    └─datastore_name*─┘


         ┌─,──────────┐                    ┌─,───────────────────┐
►──┬──────────────────────┬──┬───────────────────────────────────┬──►
   │       ┌▼─ALL──────┐  │  │           ┌▼─supermember_name───┐  │
   └─SHOW(─┼──ACKTO────┼─)┘  └─SMEM(─────┼──supermember_name*──┼─)┘
           ├──APPL─────┤                 └─────────────────────┘
           ├──CASCADE──┤
           ├──CM0ATOQ──┤
           ├──DRU──────┤
           ├──GROUP────┤
           ├──LPAR─────┤
           ├──MAXI─────┤
           ├──MEMBER───┤
           ├──MULTIRTP─┤
           ├──OAAV─────┤
           ├──RRNAME───┤
           ├──SMEM─────┤
           ├──STATE────┤
           ├──STATUS───┤
           └──TMEM─────┘


       ┌─,─────────┐              ┌─,─────────────┐
►──┬─────────────────────┬──┬───────────────────────────┬──►◄
   │        ┌▼─AVAIL──┐  │  │          ┌▼─ACTIVE─────┐  │
   └─STATE(─┼──SEVERE─┼─)┘  └─STATUS(──┼──DISCONNECT─┼─)┘
            └──WARN───┘               └──NOTACTIVE──┘
```
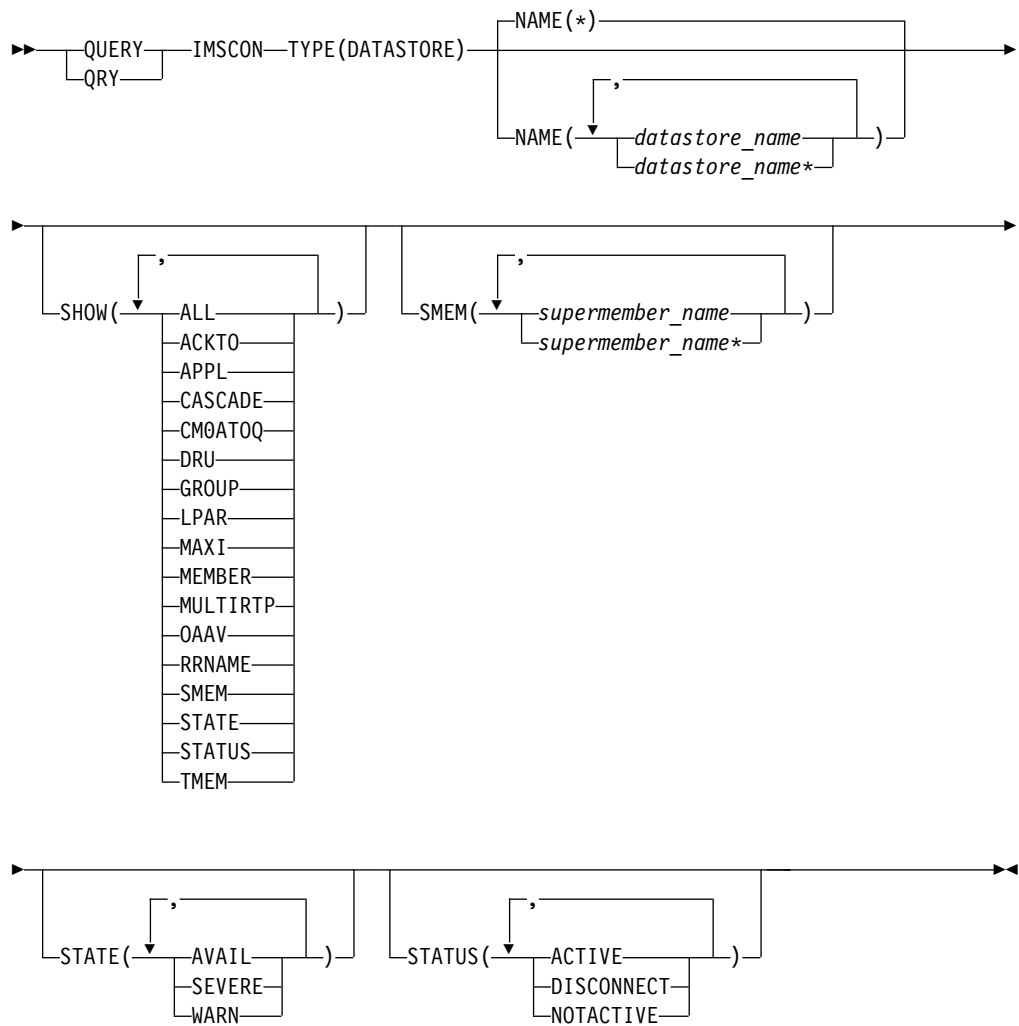
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(DATASTORE) command.

NAME

Specifies one or more data store names to be displayed. You can specify a single data store name or a list of data store names separated by commas. Wildcards can be used in the names. You can specify NAME(*) to display all data stores. NAME(*) is the default.

SHOW

Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the data store name, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL
Displays all output fields.

ACKTO
Displays the OTMA CM1 ACK timeout value, as specified by the ACKTO=
keyword in the IMS Connect configuration file.

APPL
Displays the RACF APPL name for PassTicket and data store access
control, as specified by the APPL= keyword in the IMS Connect
configuration file.

CASCADE
Displays the CASCADE value that is specified in the data store definition
and whether the support is currently enabled. The CASCADE keyword
applies to support for cascading global z/OS Resource Recovery Services
transactions to an IMS TM data store that is located on a different z/OS
image (LPAR) than IMS Connect.

CM0ATOQ
Displays the name of the OTMA CM0 ACK timeout queue, as specified by
the CM0ATOQ= keyword in the IMS Connect configuration file.

DRU
Displays the name of the OTMA destination resolution user exit that is to
be passed to OTMA, as specified by the DRU= keyword in the IMS
Connect configuration file.

GROUP
Displays the name of the z/OS cross-system coupling facility (XCF) group,
as specified by the GROUP= keyword in the IMS Connect configuration
file. This is the XCF group to which IMS Connect and IMS OTMA belong.

LPAR
Displays the name of the z/OS image (LPAR) that the data store is running
on and whether the LPAR is the same or different than the LPAR IMS
Connect is running on.

MAXI
Displays the OTMA input message flood control value, as specified by the
MAXI= keyword in the IMS Connect configuration file.

MEMBER
Displays the name of the IMS Connect member in the XCF group, as
specified by the MEMBER= keyword in the IMS Connect configuration file.

MULTIRTP
Displays the current MULTIRTP setting in the data store connection
definition. For a description of the possible values returned, see the MRTP
row in Table 48 on page 167.

OAAV
Displays the current OTMA accessor environment element (ACEE) aging
value, as specified by the OAAV= keyword in the IMS Connect
configuration file. This value determines how frequently OTMA refreshes
the security definitions that are cached in an ACEE for IMS Connect.

RRNAME
Displays the name of an alternate destination specified in a client reroute
request, as specified by the RRNAME= keyword in the IMS Connect
configuration file.

SMEM

Displays the OTMA super member name, as specified by the SMEMBER= keyword in the IMS Connect configuration file.

STATE

Displays the state of the OTMA server. For a description of the possible state returned, see the STATE keyword in Table 48 on page 167.

STATUS

Displays the status of the data store. For a description of the possible state returned, see the STATUS keyword in Table 48 on page 167.

TMEM

Displays the name of the IMS OTMA member in the XCF group, as specified by the TMEMBER= keyword in the IMS Connect configuration file.

SMEM

Selects data stores for display that have one of the specified super member names specified. You can specify a single super member name or a list of super member names separated by commas. Wildcards can be used in the names.

STATE

Selects data stores for display that are in at least one of the specified states. The filters supported with the STATE keyword, which can be specified in any order, are:

AVAIL

Selects data stores that have a state of AVAIL, which means that the OTMA server is available.

WARN

Selects data stores that have a state of WARN, which means that the OTMA server has one or more resources in warning state.

SEVERE

Selects data stores that have a state of SEVERE, which means that the OTMA server is experiencing some severe resource issues.

STATUS

Selects data stores for display that possess at least one of the specified statuses. The filters supported with the STATUS keyword, which can be specified in any order, are:

ACTIVE

Selects data stores that have a status of ACTIVE, which means that the data store is connected and active.

DISCONNECT

Selects data stores that have a status of DISCONNECT, which means that the data store resource in IMS Connect is properly started, but actual communications with IMS are not currently available.

NOTACTIVE

Selects data stores that have a status of NOTACTIVE, which means that the data store resource has been stopped in IMS Connect.

Usage notes

You can issue the QUERY IMSCON TYPE(DATASTORE) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Equivalent WTOR and z/OS commands

The following table lists WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(DATASTORE) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 47. WTOR and IMS Connect z/OS equivalents for the QUERY IMSCON TYPE(DATASTORE) command

QUERY IMSCON TYPE(DATASTORE) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
QUERY IMSCON TYPE(DATASTORE) NAME(*) SHOW(ALL | *show_parm*)	VIEWDS ALL	QUERY DATASTORE NAME(*) SHOW(ALL)
QUERY IMSCON TYPE(DATASTORE) NAME(*datastore_name*) SHOW(ALL | *show_parm*)	VIEWDS *datastore_name*	QUERY DATASTORE NAME(*datastore_name*)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 48. Output fields for the QUERY IMSCON TYPE(DATASTORE) command

Short label	Long label	Keyword	Meaning
ACKTO	AckTO	ACKTO	OTMA CM1 ACK timeout value, as specified by the ACKTO= keyword in the IMS Connect configuration file.
APPL	Appl	APPL	RACF APPL name for PassTicket and data store access control, as specified by the APPL= keyword in the IMS Connect configuration file.

Table 48. Output fields for the QUERY IMSCON TYPE(DATASTORE) command (continued)

Short label	Long label	Keyword	Meaning
CASCD	Cascade	CASCADE	Shows the specification in the data store definition for support for cascading global RRS transactions to IMS TM data stores that are located on different LPARs than IMS Connect. This specification was set either by the CASCADE= parameter of the DATASTORE= statement in the HWSCFG*xx*member of the IMS PROCLIB data set or by the SET(CASCADE()) keyword on type-2 command CREATE IMSCON TYPE(DATASTORE) or UPDATE IMSCON TYPE(DATASTORE). The value shown here overrides the IMS Connect default value for CASCADE. One of the following values is displayed: *blank* A blank value indicates that cascaded transaction support was not specified for this data store. Support for this data store is determined by the IMS Connect default specification, which can be displayed with the QUERY IMSCON TYPE(CONFIG) SHOW(CASCADE) command. **Y** Indicates that when IMS Connect and the IMS data store are on different z/OS images (LPARs), this data store connection supports cascading global IMS TM transactions. **N** Indicates that by default, this data store connection does not support cascading global IMS TM transactions.

CASCST	CascadeStatus	CASCADE	Indicates whether the data store connection is capable of cascading global RRS transactions to IMS TM data stores that are located on different z/OS images (LPARs) than IMS Connect. Possible values are:
			blank Indicates that either the data store is unavailable or the data store connection is stopped in IMS Connect.
			ENABLED
			  Indicates that support for cascading global RRS transactions to the data store is in effect because IMS Connect and the data store are on different LPARs and CASCADE=Y is specified in either the definition of the data store connection or the IMS Connect system configuration.
			DISABLED
			  Indicates that support for cascading global RRS transactions to the data store is not enabled because either:
			  • Cascading is not required because IMS Connect and the data store are on the same LPAR.
			  • CASCADE=N is specified in the definition of the data store connection.
			  • If the Cascade column is blank, CASCADE=N is specified in the IMS Connect system configuration.

CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 50 on page 171.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.

Table 48. Output fields for the QUERY IMSCON TYPE(DATASTORE) command (continued)

Short label	Long label	Keyword	Meaning
CM0ATOQ	Cm0Atoq	CM0ATOQ	Name of the OTMA CM0 ACK timeout queue, as specified by the CM0ATOQ= keyword in the IMS Connect configuration file.
DRU	DRU	DRU	Name of the OTMA destination resolution user exit that is to be passed to OTMA, as specified by the DRU= keyword in the IMS Connect configuration file.
DS	DataStore	N/A	The data store name. The data store name is always returned.
LPAR	LPAR	LPAR	Indicates whether IMS Connect and the data store are on the same z/OS image or on different z/OS images. Possible values include: *blank* Indicates that either the data store is unavailable or the data store connection is stopped in IMS Connect. **DIFFERENT** Indicates that IMS Connect and the IMS data store are running on different LPARs. **SAME** Indicates that IMS Connect and the IMS data store are running on the same LPAR.
LPARNM	LPARName	LPAR	The name of the z/OS image on which the data store is running.
MAXI	MaxI	MAXI	OTMA input message flood control value, as specified by the MAXI= keyword in the IMS Connect configuration file.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
MEMBER	Member	MEMBER	Name of the IMS Connect member in the XCF group, as specified by the MEMBER= keyword in the IMS Connect configuration file.

MRTP	MultiRTP	MULTIRTP	Shows the current specification for multiple active RESUME TPIPE request support in the data store definition. The following values are possible for MRTP:
			
			`MRTP( )`
			   The definition of this data store connection makes no specification regarding support for multiple active RESUME TPIPE requests (MULTIRTP). MULTIRTP support is determined by either the IMS Connect default specification or the specification in the OTMA client descriptor.
			
			`MRTP(#)`
			   The client bids for this data store connection do not contain a MULTIRTP specification, regardless of the IMS Connect default specification. The OTMA client descriptor determines whether multiple active resume TPIPE requests are supported.
			
			`MRTP(N)`
			   The client bids for this data store connection request a tpipe that supports only a single active RESUME TPIPE request.
			
			`MRTP(Y)`
			   The client bids for this data store connection request a tpipe that supports multiple active RESUME TPIPE requests.

OAAV	OAAV	OAAV	The current OTMA ACEE aging value, as specified by the OAAV= keyword in the IMS Connect configuration file. This value determines how frequently OTMA refreshes the security definitions that are cached in an ACEE for IMS Connect.

Short label	Long label	Keyword	Meaning
RRNAME	RRName	RRNAME	Name of an alternate destination specified in a client reroute request, as specified by the RRNAME= keyword in the IMS Connect configuration file.
SMEM	SMem	SMEM	OTMA super member name, as specified by the SMEMBER= keyword in the IMS Connect configuration file.
STATE	State	STATE	State of the OTMA server. The state can be one of the following: **AVAIL** Indicates that the OTMA server is available. **WARN** Indicates that the OTMA server has one or more resources in warning state. **SEVERE** Indicates that the OTMA server is experiencing some severe resource problems. **N/A** Indicates that OTMA has not reported status for this data store.
STT	Status	STATUS	Status of the data store. The status can be one of the following: **ACTIVE** The data store is connected and active. **NOTACTIVE** The data store resource has been stopped in IMS Connect with a command: STOPDS or UPDATE IMSCON TYPE(DATASTORE) STOP(COMM). To start communications, the data store resource in IMS Connect must be started with a command: STARTDS or UPDATE IMSCON TYPE(DATASTORE) START(COMM). **DISCONNECT** The data store resource in IMS Connect is properly started, but actual communications with IMS are not currently available, either because IMS is down, or OTMA is not started (/STOP OTMA), or the XCF group is down. Communications cannot be restored from IMS Connect commands. Instead, IMS, OTMA, or XCF must be started. If the data store goes down, IMS Connect is notified (by IMS OTMA through XCF) of the status of the data store. When the data store is brought back up and restarted, IMS Connect is notified and automatically reconnects to the data store.
TMEM	TMember	TMEM	Name of the IMS OTMA member in the XCF group, as specified by the TMEMBER= keyword in the IMS Connect configuration file.
XCFG	XCFGroup	GROUP	Name of the XCF group, as specified by the GROUP= keyword in the IMS Connect configuration file. This is the XCF group to which IMS Connect and IMS OTMA belong.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(DATASTORE) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 49. Return and reason codes for the QUERY IMSCON TYPE(DATASTORE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(DATASTORE) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 50. Completion codes for the QUERY IMSCON TYPE(DATASTORE) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(DATASTORE) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1: QUERY IMSCON TYPE(DATASTORE) SHOW(ALL)

In the following example, IMS1 and IMSA represent two data stores defined in the IMS Connect configuration file. NAME is omitted, so IMS Connect displays all data stores (the default is NAME(*)).

TSO SPOC input:
```
QUERY IMSCON TYPE(DATASTORE) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
Datastore MbrName CC XCFGroup Member   TMember State   Status
IMS1      HWS1     0 XCFGRP1  HWS1     IMS1    AVAIL   ACTIVE
IMSA      HWS1     0 XCFGRP1  HWSA     IMSA    N/A     DISCONNECT

(screen 2)
Datastore MbrName RRName   Appl    OAAV       AckTO MaxI
IMS1      HWS1    HWS$DEF  APPLID1 2147483647   120 5000
IMSA      HWS1    HWS$DEF  APPLID2 2147483647   120 5000
```

```
(screen 3)
Datastore Mbrname SMem      Cm0Atoq DRU       MultiRTP
IMS1      HWS1                       HWSYDRU0 Y
IMSA      HWS1                       HWSYDRU0 N

(screen 4)
Datastore Mbrname Cascade CascadeStatus LPAR        LPARName
IMS1      HWS1    Y       ENABLED       DIFFERENT   IMSLPAR1
IMSA      HWS1    N
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(DATASTORE) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2014.119 15:04:07.393523</statime>
<stotime>2014.119 15:04:07.409958</stotime>
<staseq>C6BAB3AD2B13AAB8</staseq>
<stoseq>C6BAB3AD2B500C78</stoseq>
<rqsttkn1>USRT001 10150810</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON           </kwd>
<input>QUERY IMSCON TYPE(DATASTORE) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="XCFG" llbl="XCFGroup" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MEMBER" llbl="Member" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TMEM" llbl="TMember" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STATE" llbl="State" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RRNAME" llbl="RRName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="APPL" llbl="Appl" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="OAAV" llbl="OAAV" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="ACKTO" llbl="AckTO" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="MAXI" llbl="MaxI" scope="LCL" sort="n" key="0" scroll="yes"
 len="6" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SMEM" llbl="SMem" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="left" skipb="yes" />
```

```
<hdr slbl="CM0ATOQ" llbl="Cm0Atoq" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DRU" llbl="DRU" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MRTP" llbl="MultiRTP" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CASCD" llbl="Cascade" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CASCST" llbl="CascadeStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPAR" llbl="LPAR" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPARNM" llbl="LPARName" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DS(IMS1    ) MBR(HWS1            ) CC(   0) XCFG(XCFGRP1 )
 MEMBER(HWS1           ) TMEM(IMS1           ) STATE(N/A       )
 STT(DISCONNECT      ) RRNAME(HWS$DEF ) APPL(APPLID1 ) OAAV(2147483647)
 ACKTO(120) MAXI(5000) SMEM(    ) CM0ATOQ(         ) DRU(HWSYDRU0)
 MRTP(Y) CASCD(Y) CASCST(ENABLED ) LPAR(DIFFERENT ) LPARNM(IMSLPAR1) </rsp>
<rsp>DS(IMSA    ) MBR(HWS1            ) CC(   0) XCFG(XCFGRP1 )
 MEMBER(HWSA           ) TMEM(IMSA           ) STATE(N/A       )
 STT(DISCONNECT      ) RRNAME(HWS$DEF ) APPL(APPLID2 ) OAAV(2147483647)
 ACKTO(120) MAXI(5000) SMEM(    ) CM0ATOQ(         ) DRU(HWSYDRU0)
 MRTP(N) CASCD(N) CASCST(        ) LPAR(          ) LPARNM(        ) </rsp>
</cmdrspdata>
</imsout>
```

Example 2: QUERY IMSCON TYPE(DATASTORE) SHOW(CASCADE,LPAR)

In the following example, the QUERY IMSCON TYPE(DATASTORE) SHOW(CASCADE,LPAR) is issued to show which IMS data store connections support the cascading of global RRS (synchlevel=syncpoint) transactions.

The specification of LPAR on the SHOW() keyword displays the z/OS image (LPAR) that each IMS data store is running on. If the IMS data store is on the same z/OS image as IMS Connect, cascaded transaction support is not needed and is therefore disabled.

For data store IMS1, the "Y" in the Cascade column indicates that cascaded transaction support is specified in the definition of the data store connection. The support is enabled, as indicated by ENABLED in the CascadeStatus column, because the IMS data store and IMS Connect are running on different LPARs.

For data store IMS2, cascaded transaction support is not explicitly enabled or disabled in the definition of the data store connection. Support is determined by the default setting that is specified for IMS Connect, which can be determined with the QUERY IMSCON TYPE(CONFIG) SHOW(CASCADE) command. However, because the IMS data store and IMS Connect are running on the same LPAR, support is disabled.

For data store IMS3, cascaded transaction support is explicitly turned off (CASCADE=N) in the definition of the data store connection. Therefore, even though the IMS data store and IMS Connect are running on different LPARs, support for cascaded transactions is disabled.

TSO SPOC input:

```
QUERY IMSCON TYPE(DATASTORE) SHOW(CASCADE,LPAR)
```

TSO SPOC output:

```
DataStore MbrName CC Cascade CascadeStatus LPAR      LPARName
IMS1      HWS1     0 Y       ENABLED       DIFFERENT IMSLPAR1
IMS2      HWS1     0         DISABLED      SAME      IMSLPAR2
IMS3      HWS1     0 N       DISABLED      DIFFERENT IMSLPAR3
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(DATASTORE) SHOW(CASCADE,LPAR) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2014.119 15:26:53.969253</statime>
<stotime>2014.119 15:26:53.978307</stotime>
<staseq>C6BAB3AD2B13AAB8</staseq>
<stoseq>C6BAB3AD2B500C78</stoseq>
<rqsttkn1>USRT001 10150810</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(DATASTORE) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CASCD" llbl="Cascade" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CASCST" llbl="CascadeStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPAR" llbl="LPAR" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPARNM" llbl="LPARName" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DS(IMS1  ) MBR(HWS1            ) CC(   0)
 CASCD(Y) CASCST(ENABLED ) LPAR(DIFFERENT ) LPARNM(IMSLPAR1) </rsp>
<rsp>DS(IMS2  ) MBR(HWS1            ) CC(   0)
 CASCD( ) CASCST(DISABLED) LPAR(SAME      ) LPARNM(IMSLPAR2) </rsp>
<rsp>DS(IMS3  ) MBR(HWS1            ) CC(   0)
 CASCD(N) CASCST(DISABLED) LPAR(DIFFERENT ) LPARNM(IMSLPAR3) </rsp>
</cmdrspdata>
</imsout>
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

VIEWDS command (Commands)

QUERY IMSCON TYPE(IMSPLEX) command

Use the QUERY IMSCON TYPE(IMSPLEX) command to display the status and activity of the IMSplex.

An IMSplex is primarily defined with the IMSPLEX configuration statement in the IMS Connect configuration file. An IMSplex can also be defined using the IMSPLEX keyword in either the ODACCESS or MSC configuration statement in the IMS Connect configuration file.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 176
- "Equivalent WTOR and z/OS commands" on page 177
- "Output fields" on page 177
- "Return, reason, and completion codes" on page 178
- "Examples" on page 179

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
                                          ┌─NAME(*)─────────────────┐
►►─┬─QUERY─┬─IMSCON─TYPE(IMSPLEX)─┼─────────────────────────┼──────►
   └─QRY───┘                       │          ┌─,─────────┐ │
                                   └─NAME(─▼─┬─implex_name─┬─┴─)─┘
                                            └─imsplex_name*─┘

►─┬──────────────────────┬─┬──────────────────────────────┬─►◄
  │      ┌─,──────┐       │ │       ┌─,──────────┐         │
  └─SHOW(─▼─┬─ALL────┬─┴─)─┘ └─STATUS(─▼─┬─ACTIVE─────┬─┴─)─┘
           ├─MEMBER─┤              ├─DISCONNECT─┤
           └─STATUS─┘              └─NOTACTIVE──┘
```
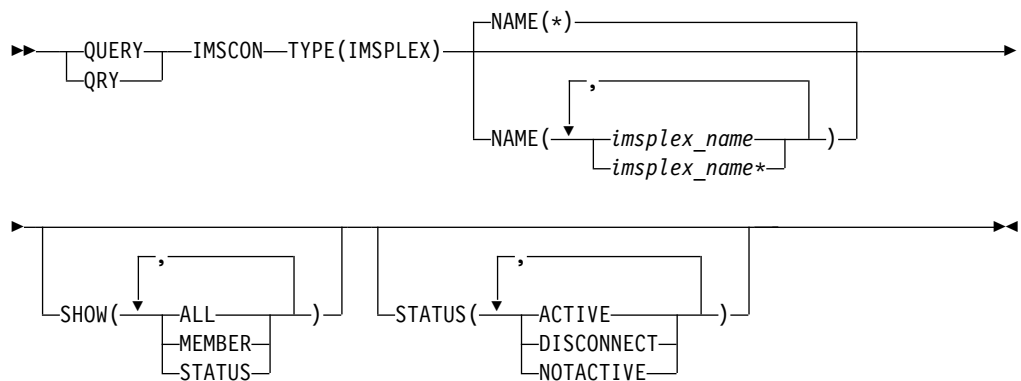
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(IMSPLEX) command.

NAME

Specifies one or more IMSplex names to be displayed. The IMSplex name is defined in the TMEMBER parameter of the IMSplex configuration statement in the IMS Connect configuration file.

You can specify a single IMSplex name or a list of IMSplex names separated by commas. Wildcards can be used in the names. You can specify NAME(*) to display all IMSplex resources. NAME(*) is the default.

SHOW

Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the IMSplex name, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL

Displays all output fields.

MEMBER

Displays the name of the IMS Connect member in the IMSplex, as specified by the MEMBER= keyword in the IMSPLEX statement in the IMS Connect configuration file.

STATUS

Displays the status of the IMSplex. For a description of the possible state returned, see the STATUS keyword in Table 52 on page 177.

If the IMSplex goes down, IMS Connect is notified (through SCI) of the status of the IMSplex. When the IMSplex is brought back up and restarted, IMS Connect is notified and automatically reconnects to the IMSplex.

STATUS

Selects IMSplex resources for display that possess at least one of the specified statuses. The filters supported with the STATUS keyword, which can be specified in any order, are:

ACTIVE

Selects IMSplex resources that have a status of ACTIVE, which means that the IMSplex is connected and active.

DISCONNECT

Selects IMSplex resources that have a status of DISCONNECT, which means that SCI is not active so that communication between IMS Connect and the IMSplex is currently not available.

NOTACTIVE

Selects IMSplex resources that have a status of NOTACTIVE, which means that communication between IMS Connect and the IMSplex is stopped.

Usage notes

You can issue the QUERY IMSCON TYPE(IMSPLEX) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Equivalent WTOR and z/OS commands

The following table lists WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(IMSPLEX) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 51. WTOR and IMS Connect z/OS equivalents for the QUERY IMSCON TYPE(IMSPLEX) command

QUERY IMSCON TYPE(IMSPLEX) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
QUERY IMSCON TYPE(IMSPLEX) NAME(*) SHOW(ALL |*show_parm*)	VIEWIP ALL	QUERY IMSPLEX NAME(*) SHOW(ALL)
QUERY IMSCON TYPE(IMSPLEX) NAME(*IMSplex_name*) SHOW(ALL | *show_parm*)	VIEWIP *IMSplex_name*	QUERY IMSPLEX NAME(*imsplexName*) SHOW(ALL)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 52. Output fields for the QUERY IMSCON TYPE(IMSPLEX) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 54 on page 179.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
IMSPLX	IMSplex	N/A	The IMSplex name. The IMSplex name is always returned.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.

Table 52. Output fields for the QUERY IMSCON TYPE(IMSPLEX) command (continued)

Short label	Long label	Keyword	Meaning
MEMBER	Member	MEMBER	Name of the IMS Connect member in the IMSplex, as specified by the MEMBER= keyword in the IMSPLEX statement in the IMS Connect configuration file.
STT	Status	STATUS	Status of the IMSplex. The status can be one of the following: **ACTIVE** The IMSplex is connected and active. **DISCONNECT** SCI is not active, so communication between IMS Connect and the IMSplex is currently not available. **NOTACTIVE** Communication between IMS Connect and the IMSplex is stopped. If SCI goes down, IMS Connect is notified (through SCI) of the status of the IMSplex. When SCI is brought back up and restarted, IMS Connect is notified and automatically reconnects to the IMSplex.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(IMSPLEX) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 53. Return and reason codes for the QUERY IMSCON TYPE(IMSPLEX) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(IMSPLEX) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 54. Completion codes for the QUERY IMSCON TYPE(IMSPLEX) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(IMSPLEX) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for QUERY IMSCON TYPE(IMSPLEX) command

TSO SPOC input:

```
QUERY IMSCON TYPE(IMSPLEX) SHOW(ALL)
```

TSO SPOC output:

```
IMSplex    MbrName CC Member    Status
PLEX1      HWS1     0 ICON1     ACTIVE
PLEX2      HWS1     0 ICON2     NOTACTIVE
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(IMSPLEX) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.297 23:24:10.869666</statime>
<stotime>2010.297 23:24:10.870584</stotime>
<staseq>C6C7575069FA2038</staseq>
<stoseq>C6C757506A338578</stoseq>
<rqsttkn1>USRT001 10162410</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>ICON1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(IMSPLEX) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MEMBER" llbl="Member" scope="LCL" sort="n" key="0"
```

```
scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>IMSPLX(PLEX1   ) MBR(HWS1              ) CC(   0) MEMBER(ICON1
     ) STT(ACTIVE          ) </rsp>
<rsp>IMSPLX(PLEX2   ) MBR(HWS1              ) CC(   0) MEMBER(ICON2
     ) STT(NOTACTIVE       ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: PLEX1 and PLEX2 represent two IMSplex resources defined in the IMS Connect configuration file. NAME is omitted, so IMS Connect displays all IMSplex resources (the default is NAME(*)).

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

VIEWIP command (Commands)

QUERY IMSCON TYPE(ISC) command

Use the QUERY IMSCON TYPE(ISC) command to display the status and attributes of one or more ISC links that are defined to IMS Connect.

An ISC link represents the connection between an ISC node in IMS and a remote CICS subsystem. An ISC link in IMS Connect and an ISC node in IMS have a one-to-one relationship.

An ISC link can support multiple ISC parallel sessions (ISCUSERS). ISCUSERs are displayed only when they are active.

Subsections:
- "Environment"
- "Syntax" on page 181
- "Keywords" on page 181
- "Usage notes" on page 184
- "Equivalent WTOR and z/OS commands" on page 185
- "Output fields" on page 185
- "Return, reason, and completion codes" on page 187
- "Examples" on page 188

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►─┬─QUERY─┬─IMSCON─TYPE(ISC)─┬─NAME(*)────────────────────┬─────────►
   └─QRY───┘                  │         ┌─,─────────┐      │
                              └─NAME(─▼─┬─isc_id──┬──┴─)───┘
                                         └─isc_id*─┘

►──┬─────────────────────────────────┬──┬─────────────────┬──────────►
   │        ┌─,──────────────┐        │  └─IUSTATUS(ACTIVE)─┘
   └─ISCUSER(─▼─┬─iscuser_name──┬──┴─)─┘
                └─iscuser_name*─┘

►──┬──────────────────────┬──┬───────────────────────┬──────────────►◄
   │      ┌─,────────┐     │  │      ┌─,──────────┐   │
   └─SHOW(─▼─┬─ALL──────┬─┴─)─┘  └─STATUS(─▼─┬─ACTIVE─────┬─┴─)─┘
             ├─CICSAPPL─┤                    ├─DISCONNECT─┤
             ├─CICSNETID┤                    └─NOTACTIVE──┘
             ├─CICSPORT─┤
             ├─HOSTNAME─┤
             ├─IMSPLEX──┤
             ├─IPADDR───┤
             ├─IPMEMBER─┤
             ├─ISCUSER──┤
             ├─LCLIMS───┤
             ├─NODE─────┤
             ├─PORT─────┤
             ├─RMTCICS──┤
             ├─STATUS───┤
             └─SUMMARY──┘
```
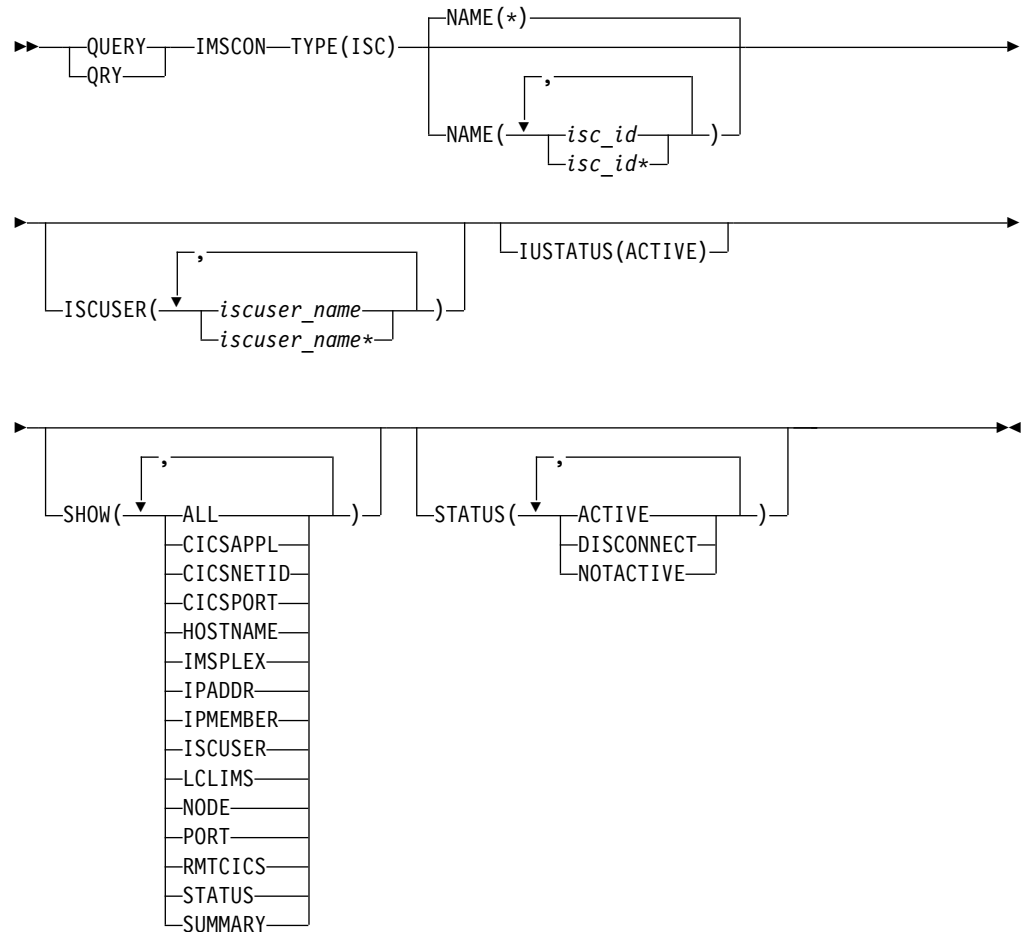
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(ISC) command.

ISCUSER

> This filter selects ISC link resources for display that have one of the specified ISC parallel sessions (ISCUSERs) defined. You can specify a single ISCUSER name, a wildcard name, or a list of ISCUSER names separated by commas.
>
> Only those ISC link resources that have a specified ISCUSER are displayed. ISC link resources that match the NAME() parameter, but that do not have the specified ISCUSER, are not displayed.
>
> When the ISCUSER filter is specified, ISCUSER information is displayed even if SHOW(ISCUSER) is not specified.
>
> Only those ISCUSERs that match the names specified by the ISCUSER filter are displayed. ISCUSERs that are associated with the ISC link that do not match the filter are not displayed.

IUSTATUS

> This filter selects ISC link resources for display that have at least one ISC parallel session (ISCUSER) that is in the specified state.

The parameter supported with the IUSTATUS keyword is:

ACTIVE
> This ISCUSER is active.

Only those ISC links that have an ISCUSER that is in the specified state are displayed. ISC links that match the NAME() parameter, but that do not have such an ISCUSER, are not displayed.

When the IUSTATUS filter is specified, ISCUSER information is displayed even if SHOW(ISCUSER) is not specified.

Only those ISCUSERs that are in the specified state are displayed. ISCUSERs that are not in the specified states are not displayed.

NAME
> Specifies one or more ISC links to be displayed. You can specify a single ISC link name or a list of ISC link names separated by commas. Wildcards can be used in the name.
>
> You can specify NAME(*) to display all ISC link definitions. NAME(*) is the default.
>
> If the NAME keyword is omitted, then all ISC link resources are displayed.

SHOW
> Specifies the optional output fields to be displayed.
>
> Certain output fields that are always displayed, even when they are not specified in the SHOW keyword. Output fields that are always displayed include the ISC link name, the name of the IMS Connect that processes the command, and the completion code.
> The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL
> Displays all output fields.

CICSAPPL
> Displays the APPLID of the remote CICS subsystem, as defined to IMS Connect on the CICSAPPL parameter of the ISC configuration statement. This value matches the APPLID value defined to CICS on the APPLID parameter of the DFHSIT*xx* member.

CICSNETID
> Displays the network ID of the remote CICS subsystem.

CICSPORT
> Displays the port on which an ISC link receives transactions and data replies from CICS. This receive port is defined in IMS Connect on the CICSPORT keyword in both the TCPIP and ISC configuration statements. This port number must be specified to the CICS subsystem on the PORT keyword of the IPCONN resource definition.

HOSTNAME
> Displays the host name of the remote CICS subsystem. The host name of a remote CICS is defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.

IMSPLEX
> Displays the name of the IMSplex, as defined in the TMEMBER subparameter of the IMSPLEX parameter of the ISC statement in the IMS Connect configuration member.

IPADDR
Displays the IP address of the remote CICS subsystem.

IPMEMBER
Displays the IMSplex member name, as defined in the MEMBER subparameter of the IMSPLEX parameter of the ISC statement in the IMS Connect configuration member. This is the IMS Connect member name registered to the IMSplex.

ISCUSER

Display information and status for the ISC parallel sessions (ISCUSERs) that are active on the specified ISC link. If the ISC link has one or more active ISCUSERs, the command output includes information about each ISCUSER. A separate output line is displayed for each ISCUSER. This is in addition to any output lines that display general ISC link information. If there are no active ISCUSERs, then no additional output line is displayed.

Information displayed for the ISCUSER includes:
- The name of the ISCUSER.

  For ISC terminals that are statically defined in IMS, the ISCUSER name is derived from the NAME keyword of the IMS SUBPOOL stage-1 system definition macro. For ISC terminals that are dynamically defined in IMS gen, the ISCUSER name is derived from the USER keyword on the /OPNDST NODE command.
- The receive client name, which is the client name that IMS Connect assigns to the receive socket for this ISCUSER.
- The send client name, which is the client name that IMS Connect assigns to the send socket for this ISCUSER.
- The status or state of the ISCUSER. The possible states of an ISCUSER can be:

  ACTIVE
  The ISCUSER is active.

LCLIMS
Displays the name of the local IMS as defined in the LCLIMS parameter of the ISC statement in the IMS Connect configuration member.

NODE
Displays the ISC node names as they are defined to IMS Connect and the local IMS system. Node names are defined to IMS Connect on the NODE parameter of the ISC statements that define the ISC links.

PORT
Displays the port on which the remote CICS subsystem receives transactions and data replies from IMS Connect on this ISC link. This port is defined in CICS on the PORT keyword of the TCPIPSERVICE resource definition. This port is specified in IMS Connect on the PORT keyword of the RMTCICS statement that defines the connection to the remote CICS subsystem.

RMTCICS
Displays the name of the remote CICS connection as defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.

STATUS
Displays the status of the ISC link. The status can be one of the following:

ACTIVE
The link is active.

DISCONNECT
The link is active, but communication with IMS is not currently available because IMS or SCI has terminated, or the IMSplex datastore associated with the ISC link has been stopped.

NOTACTIVE
The link is not active.

For a description of the possible state returned, see the STATUS keyword in Table 55 on page 185.

SUMMARY
Displays summary information related to the ISC parallel sessions (ISCUSERs) that are active on the ISC link. To view specific ISCUSER information, specify the ISCUSER parameter in addition to the SUMMARY parameter.

Information displayed includes:
- Total number of ISCUSERs active on the ISC link.
- Total number of ISCUSERs in ACTIVE state.
- Total number of ISCUSERs not in any of these states.

STATUS
Selects ISC links for display that possess at least one of the specified statuses.

When the STATUS filter is specified, status information is displayed even if SHOW(STATUS) is not specified.

The filters supported with the STATUS keyword, which can be specified in any order, are:

ACTIVE
Selects ISC links that have a status of ACTIVE.

DISCONNECT
Selects ISC links that have a status of DISCONNECT.

NOTACTIVE
Selects ISC links that have a status of NOTACTIVE.

Usage notes

You can issue the QUERY IMSCON TYPE(ISC) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Typically, this command results in one output display line for each ISC link definition being displayed. However, if SHOW(ISCUSER) or ISCUSER(*iscuser_name*) is specified, an additional line is displayed for each ISC parallel sessions (ISCUSERs) associated with the specified ISC link definition that matches the ISCUSER() filter.

Equivalent WTOR and z/OS commands

The QUERY IMSCON TYPE(ISC) command does not have any equivalent WTOR or z/OS MODIFY commands.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
Provides a brief description of the output field.

Table 55. Output fields for the QUERY IMSCON TYPE(ISC) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 57 on page 188.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
CICAP	CICSApplid	CICSAPPL	The APPLID of the remote CICS.
CICNT	CICSNetid	CICSNETID	The network ID of the remote CICS.
CICPT	CICSPort	CICSPORT	The CICSPORT port ID that is defined in the TCPIP statement in the IMS Connect configuration member.
HOST	HostName	HOSTNAME	The host name of the remote CICS subsystem. Remote CICS subsystems are defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.
IMEM	IpMember	IPMEMBER	The IMSplex member name, as defined in the MEMBER subparameter of the IMSPLEX parameter of the ISC statement in the IMS Connect configuration member. This name is the IMS Connect member name registered to the IMSPlex.
IMSPLX	IMSplex	IMSPLEX	The name of the IMSplex, as defined in the TMEMBER subparameter of the IMSPLEX parameter of the ISC statement in the IMS Connect configuration member.
IP	IpAddress	IPADDR	The IP address of the remote CICS. The remote CICS is defined in the RMTCICS of the ISC statement in the IMS Connect configuration member.

Table 55. Output fields for the QUERY IMSCON TYPE(ISC) command (continued)

Short label	Long label	Keyword	Meaning
ISC	ISCName	N/A	The name of the ISC link, as it is defined on the ID keyword of the ISC statement in the IMS Connect configuration member. The ISC name is always returned.
IUSER	ISCUser	ISCUSER	The name of an ISC parallel session (ISCUSER) that is active on the ISC link. For ISC terminals that are statically defined in IMS, the ISCUSER name is derived from the NAME keyword of the IMS SUBPOOL stage-1 system definition macro. For ISC terminals that are dynamically defined in IMS gen, the ISCUSER name is derived from the USER keyword on the /OPNDST NODE command.
IUSTT	IUStatus	ISCUSER	For an ISC parallel session (ISCUSER) on an ISC link, this is the status or state of the ISCUSER. An ISCUSER can be in the following states: • ACTIVE. The ISCUSER is active.
LIMS	LclIMS	LCLIMS	The name of the local IMS as defined in the LCLIMS parameter of the ISC statement in the IMS Connect configuration member.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
NODE	Node	NODE	The local ISC terminal name that is using this connection, as specified on the NODE parameter of the ISC statement in the IMS Connect configuration member.
PORT	Port	PORT	The port of the associated remote CICS subsystem. The remote CICS is defined on the RMTCICS parameter of the ISC statement.
RCICS	RmtCICS	RMTCICS	The name of the remote CICS connection as defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.
RCL	RecvClnt	ISCUSER	The client name that IMS Connect assigns to the socket on the CICSPORT that is used to receive the transactions and data replies from a remote CICS subsystem (RMTCICS) for the ISC parallel session that is identified in the corresponding IUSER output field.
SCL	SendClnt	ISCUSER	The client name that IMS Connect assigns to the socket that is used to send the transactions and data replies to a remote CICS subsystem (RMTCICS) for the ISC parallel session that is identified in the corresponding IUSER output field.

Table 55. Output fields for the QUERY IMSCON TYPE(ISC) command (continued)

Short label	Long label	Keyword	Meaning

STT	Status	STATUS	Status of the ISC link. The status can be one of the following:
			
			ACTIVE
			    The ISC link is active.
			
			DISCONNECT
			    The link is active, but communication with IMS is not currently available because IMS or SCI has terminated, or the IMSplex datastore associated with the ISC link has been stopped.
			
			NOTACTIVE
			    The ISC link is not active.

TACT	TotActiv	SUMMARY	Total number of ISC parallel sessions (ISCUSERs) that have a status of ACTIVE on the ISC link.
TOTH	TotOther	SUMMARY	Total number of ISC parallel sessions (ISCUSERs) that have a status that is other than ACTIVE on the ISC link.
TUSR	TotUsers	SUMMARY	Total number of ISC parallel sessions (ISCUSERs) that are active on the ISC link.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(ISC) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 56. Return and reason codes for the QUERY IMSCON TYPE(ISC) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(ISC) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C000008'	X'00002004'	An invalid keyword or keyword parameter was specified.
X'0C000008'	X'00002014'	An invalid character was specified in the NAME() parameter.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 57. Completion codes for the QUERY IMSCON TYPE(ISC) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(ISC) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

In the following example, IMS1CIC4 is the ID of the ISC statement that defines the ISC link to IMS Connect. The link is between a local IMS system that is name IMS1 and a remote CICS system that is identified by its APPLID: CICS1.

The TCP/IP connection to the CICS subsystem is defined by a RMTCICS statement. The ID of the RMTCICS statement is CICS4.

In IMS, the connection is identified by its ISC node name, CIC4.

Two parallel sessions are active on the ISC link: CIC4U07 and CIC4U08.

TSO SPOC input:
```
QUERY IMSCON TYPE(ISC) NAME(IMS1CIC4) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
ISCName  MbrName          CC Node     LclIMS   IpMember
IMS1CIC4 HWS1              0 CIC4      IMS1     HWS1
IMS1CIC4 HWS1              0
IMS1CIC4 HWS1              0
(screen 2)
ISCName  MbrName          IMSplex        RmtCics  IpAddress
IMS1CIC4 HWS1             PLEX1          CICS4    127.0.0.1
IMS1CIC4 HWS1
IMS1CIC4 HWS1
(screen 3)
ISCName  MbrName          HostName                    Port CICSApplid CICSPort
IMS1CIC4 HWS1             WWW.EXAMPLE.COM             7777 CICS1      9876
IMS1CIC4 HWS1
IMS1CIC4 HWS1
(screen 4)
ISCName  MbrName          CICSNetid Status     TotUsers TotActiv TotOther
IMS1CIC4 HWS1             IMSNET    ACTIVE            2        2        0
IMS1CIC4 HWS1
IMS1CIC4 HWS1
(screen 5)
ISCName  MbrName          ISCUser   SendClnt RecvClnt IUStatus
IMS1CIC4 HWS1
IMS1CIC4 HWS1             CIC4U07   ISCC2714 ISC41218 ACTIVE
IMS1CIC4 HWS1             CIC4U08   ISCB5D1A ISCC279A ACTIVE
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(ISC) NAME(IMS1CIC4) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2012.221 18:20:04.944914</statime>
<stotime>2012.221 18:20:04.946923</stotime>
<staseq>C9FD571B32812594</staseq>
<stoseq>C9FD571B32FEB114</stoseq>
<rqsttkn1>USRT001 10112004</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(ISC) NAME(IMS1CIC4) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="NODE" llbl="Node" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LIMS" llbl="LclIMS" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IMEM" llbl="IpMember" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RIC" llbl="RmtImsCon" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RCICS" llbl="RmtCics" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="HOST" llbl="HostName" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CICAP" llbl="CICSApplid" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CICPT" llbl="CICSPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CICNT" llbl="CICSNetid" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TUSR" llbl="TotUsers" scope="LCL" sort="d" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TACT" llbl="TotActiv" scope="LCL" sort="d" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="d" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="IUSER" llbl="ISCUser" scope="LCL" sort="a" key="3"
```

```
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RCL" llbl="RecvClnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IUSTT" llbl="IUStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="9" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>ISC(IMS1CIC4) MBR(HWS1            ) CC(   0) IUSER(CIC4U07 )
 SCL(ISCC2714) RCL(ISC41218) IUSTT(ACTIVE   ) </rsp>
<rsp>ISC(IMS1CIC4) MBR(HWS1            ) CC(   0) IUSER(CIC4U08 )
 SCL(ISCB5D1A) RCL(ISCC279A) IUSTT(ACTIVE   ) </rsp>
<rsp>ISC(IMS1CIC4) MBR(HWS1            ) CC(   0) NODE(CIC4    )
 LIMS(IMS1  ) IMEM(HWS1            ) IMSPLX(PLEX1          )
 RCICS(CICS4  ) IP(127.0.0.1) HOST(WWW.EXAMPLE.COM)
 PORT(7777) CICAP(CICS1  ) CICPT(9876    ) CICNT(IMSNET ) STT(ACTIVE
  ) TUSR(2) TACT(2) TOTH(0) </rsp>
</cmdrspdata>
</imsout>
```

Related reference:

ISC statement (System Definition)

"UPDATE IMSCON TYPE(ISC) command" on page 1133

"UPDATE IMSCON TYPE(RMTCICS) command" on page 1174

QUERY IMSCON TYPE(ISCUSER) command

Use the QUERY IMSCON TYPE(ISCUSER) command to display the current status
of one or more ISC parallel sessions that are active in IMS Connect.

You specify the ISC parallel sessions that you need to display by the user name of
the parallel session. The ISC parallel session is associated with an ISC link. ISC
links are defined to IMS Connect by ISC statements in the IMS Connect
configuration member.

Subsections:
- "Environment"
- "Syntax" on page 191
- "Keywords" on page 191
- "Usage notes" on page 194
- "Equivalent WTOR and z/OS commands" on page 194
- "Output fields" on page 194
- "Return, reason, and completion codes" on page 196
- "Examples" on page 197

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this
command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common
  Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and
  Operations Manager (OM) must be active.

Syntax

```
>>─┬─QUERY─┬─IMSCON─TYPE(ISCUSER)─┬─NAME(*)────────────────────────┬─────────>
   └─QRY───┘                      │         ┌──,──────────┐        │
                                  │         │ ▼           │        │
                                  └─NAME(───┬──iscuser_id──┬──)────┘
                                            └──iscuser_id*─┘

    ┌──────────────────────────────────┐
    │         ┌──,──────┐               │         ┌─STATUS(ACTIVE)─┐
>───┤         │ ▼       │               ├─────────┴────────────────┴───><
    └─SHOW(───┬──ALL─────────┬──)───────┘
              ├─CICSAPPL─────┤
              ├─CICSNETID────┤
              ├─CICSPORT─────┤
              ├─HOSTNAME─────┤
              ├─IMSPLEX──────┤
              ├─IPADDR───────┤
              ├─IPMEMBER─────┤
              ├─ISC──────────┤
              ├─ISTATUS──────┤
              ├─LCLIMS───────┤
              ├─NODE─────────┤
              ├─PORT─────────┤
              ├─RECVCLNT─────┤
              ├─RMTCICS──────┤
              ├─SENDCLNT─────┤
              └─STATUS───────┘
```
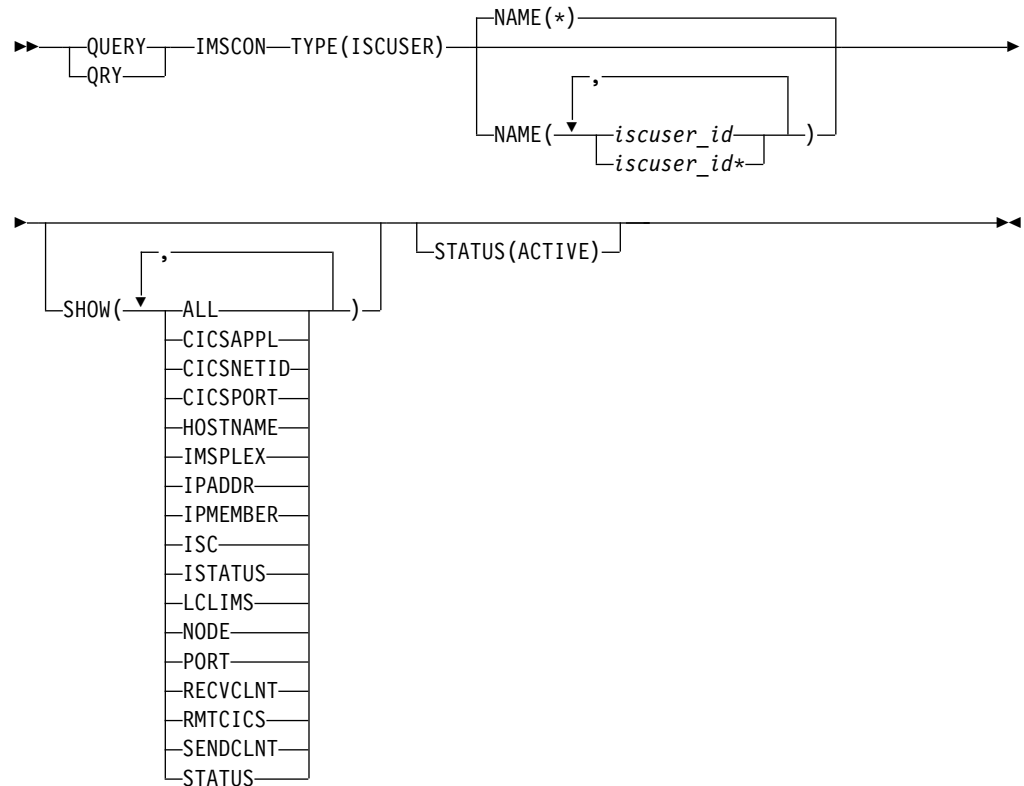
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(ISCUSER) command.

NAME

Specify the ISC user names (ISCUSER) of the parallel sessions that you need to display. You can specify one or more ISC user names separated by commas. Wildcards can be used in the name.

The ISC user names that you specify must match the user names that were used when the parallel session was started.

If the ISC terminals were statically defined in IMS, the user names are defined on the NAME keyword of the SUBPOOL macro.

If the ISC terminals are dynamically defined and the parallel session was started from IMS, the user name is defined on the USER keyword of the /OPNDST NODE command that started the session. If the parallel session of a dynamically defined terminal is started from CICS, the user name is defined on the NAME and APPLID parameters of the IPCONN resource definition.

You can specify NAME(*) to display all ISCUSER resources. NAME(*) is the default.

If the NAME keyword is omitted, all ISCUSER resources are displayed.

SHOW

Specifies the optional output fields to be displayed.

Certain output fields that are always displayed, even when they are not specified in the SHOW keyword. Output fields that are always displayed include the ISCUSER name, the name of the IMS Connect that processes the command, and the completion code.

The SHOW keyword supports the following parameters, which can be specified in any order:

ALL
Displays all output fields.

CICSAPPL
Displays the APPLID of the remote CICS subsystem, as defined to IMS Connect on the CICSAPPL parameter of the ISC configuration statement. This value matches the APPLID value defined to CICS on the APPLID parameter of the DFHSIT*xx* member.

CICSNETID
Displays the network ID of the remote CICS subsystem.

CICSPORT
Displays the port on which an ISC link receives transactions and data replies from CICS. This receive port is defined in IMS Connect on the CICSPORT keyword in both the TCPIP and ISC configuration statements. This port number must be specified to the CICS subsystem on the PORT keyword of the IPCONN resource definition.

HOSTNAME
Displays the host name of the remote CICS subsystem. The host name of a remote CICS is defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.

IMSPLEX
Displays the name of the IMSplex, as defined in the TMEMBER subparameter of the IMSPLEX parameter of the ISC statement in the IMS Connect configuration member.

IPADDR
Displays the IP address of the remote CICS subsystem.

IPMEMBER
Displays the IMSplex member name, as defined in the MEMBER subparameter of the IMSPLEX parameter of the ISC statement in the IMS Connect configuration member. This is the IMS Connect member name registered to the IMSplex.

ISC

Displays the name of the ISC link that is associated with the ISCUSER, as defined by the ID parameter of the ISC statement in the IMS Connect configuration member.

ISTATUS
Displays the status of the ISC terminal. The status can be one of the following:

   ACTIVE
      The terminal is active.

   NOTACTIVE
      The terminal is not active.

LCLIMS
Displays the name of the local IMS as defined in the LCLIMS parameter of the ISC statement in the IMS Connect configuration member.

NODE
Displays the ISC node names as they are defined to IMS Connect and the local IMS system. Node names are defined to IMS Connect on the NODE parameter of the ISC statements that define the ISC links.

PORT
Displays the port on which the remote CICS subsystem receives transactions and data replies from IMS Connect on this ISC link. This port is defined in CICS on the PORT keyword of the TCPIPSERVICE resource definition. This port is specified in IMS Connect on the PORT keyword of the RMTCICS statement that defines the connection to the remote CICS subsystem.

RECVCLNT
Displays the receive client name that identifies the socket on which IMS receives transactions and reply data from the remote CICS subsystem for this ISC parallel session (ISCUSER). The receive client name is randomly generated by IMS Connect and persists only while the ISC parallel session is active. Receive client names for ISC parallel sessions start with the characters "ISC".

RMTCICS
Displays the name of the remote CICS connection as defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.

SENDCLNT
Displays the send client name that identifies the socket that IMS uses to send transactions and reply data to the remote CICS subsystem for this ISC parallel session (ISCUSER). The send client name is randomly generated by IMS Connect and persists only while the ISC parallel session is active. Send client names for ISC parallel sessions start with the characters "ISC".

STATUS
Displays the status of the ISCUSER. The status can be one of the following:

> **ACTIVE**
> The ISCUSER is active.

For a description of the possible state returned, see the STATUS keyword in Table 58 on page 194.

STATUS
This filter selects ISCUSER resources for display that possess at least one of the specified statuses.

When the STATUS filter is specified, status information is displayed even if SHOW(STATUS) is not specified.

The filters supported with the STATUS keyword, which can be specified in any order, are:

ACTIVE
Selects ISC links that have a status of ACTIVE.

Usage notes

You can issue the QUERY IMSCON TYPE(ISCUSER) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

This command results in one output display line for each ISCUSER that is displayed.

Equivalent WTOR and z/OS commands

The QUERY IMSCON TYPE(ISCUSER) command does not have any equivalent WTOR or z/OS MODIFY commands.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 58. Output fields for the QUERY IMSCON TYPE(ISCUSER) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 60 on page 196.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
CICAP	CICSApplid	CICSAPPL	The APPLID of the remote CICS.
CICNT	CICSNetid	CICSNETID	The network ID of the remote CICS.
CICPT	CICSPort	CICSPORT	The CICSPORT port ID that is defined in the TCPIP statement in the IMS Connect configuration member.
HOST	HostName	HOSTNAME	The host name of the remote CICS subsystem. Remote CICS subsystems are defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.

Table 58. Output fields for the QUERY IMSCON TYPE(ISCUSER) command (continued)

Short label	Long label	Keyword	Meaning
IMEM	IpMember	IPMEMBER	The IMSplex member name, as defined in the MEMBER subparameter of the IMSPLEX parameter of the ISC statement in the IMS Connect configuration member. This name is the IMS Connect member name registered to the IMSPlex.
IMSPLX	IMSplex	IMSPLEX	The name of the IMSplex, as defined in the TMEMBER subparameter of the IMSPLEX parameter of the ISC statement in the IMS Connect configuration member.
IP	IpAddress	IPADDR	The IP address of the remote CICS. The remote CICS is defined in the RMTCICS of the ISC statement in the IMS Connect configuration member.
ISC	ISCName	ISC	The name of the ISC link that is associated with the ISCUSER, as defined by the ISC configuration statement in the IMS Connect configuration member.

ISTT	ISCStatus	ISTATUS	The status of the ISC link that associated with the parallel session (ISCUSER) output field. The status can be one of the following:
			
			ACTIVE
			    The link is active.
			
			NOTACTIVE
			    The link is not active.

IUSER	ISCUser	N/A	Name of the ISCUSER that represents an ISC parallel session. The name of the ISCUSER is always returned.
LIMS	LclIMS	LCLIMS	The name of the local IMS as defined in the LCLIMS parameter of the ISC statement in the IMS Connect configuration member.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
NODE	Node	NODE	The local ISC terminal name that is using this connection, as specified on the NODE parameter of the ISC statement in the IMS Connect configuration member.
PORT	Port	PORT	The port of the associated remote CICS subsystem. The remote CICS is defined on the RMTCICS parameter of the ISC statement.
RCICS	RmtCICS	RMTCICS	The name of the remote CICS connection as defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.
RCL	RecvClnt	RECVCLNT	The client name that IMS Connect assigns to the socket on the CICSPORT that is used to receive the transactions and data replies from a remote CICS subsystem (RMTCICS) for the ISC parallel session that is identified in the corresponding IUSER output field.

Table 58. Output fields for the QUERY IMSCON TYPE(ISCUSER) command (continued)

Short label	Long label	Keyword	Meaning
SCL	SendClnt	SENDCLNT	The client name that IMS Connect assigns to the socket that is used to send the transactions and data replies to a remote CICS subsystem (RMTCICS) for the ISC parallel session that is identified in the corresponding IUSER output field.
STT	Status	STATUS	Status of the ISCUSER. The status can be one of the following: **ACTIVE** The ISCUSER is active.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(ISCUSER) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 59. Return and reason codes for the QUERY IMSCON TYPE(ISCUSER) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(ISCUSER) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C000008'	X'00002004'	An invalid keyword or keyword parameter was specified.
X'0C000008'	X'00002014'	An invalid character was specified in the NAME() parameter.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 60. Completion codes for the QUERY IMSCON TYPE(ISCUSER) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(ISCUSER) command completed successfully for the resources.

Table 60. Completion codes for the QUERY IMSCON TYPE(ISCUSER) command (continued)

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for QUERY IMSCON TYPE(ISCUSER) command

In the following example, there are two parallel sessions active on the ISC link CICS that have active communications between CICS system CICSA1 and IMS system IMS1. The CICS link is defined to IMS as terminal node name CICSTRM1.

TSO SPOC input:
```
QUERY IMSCON TYPE(ISCUSER) NAME(IMSUS*) SHOW(ALL)
```

TSO SPOC output:
```
(Screen 1)
 ISCUser   MbrName CC  SendClnt   RecvClnt Status  ISCName  ISCStatus
 IMSUS01   HWS1     0  ISC11111   ISC33333 ACTIVE  CICS     ACTIVE
 IMSUS02   HWS1     0  ISC22222   ISC44444 ACTIVE  CICS     ACTIVE

 (Screen 2)

 ISCUser   MbrName Node      LclIMS IpMember IMSplex RmtCICS
 IMSUS01   HWS1    CICSTRM1  IMS1   ICON1    PLEX1   ICONNT1
 IMSUS02   HWS1    CICSTRM1  IMS1   ICON1    PLEX1   ICONNT1

 (Screen 3)
 ISCUser   MbrName IpAddress      HostName           Port
 IMSUS01   HWS1    127.0.0.1      WWW.EXAMPLE.COM     9998
 IMSUS02   HWS1    127.0.0.1      WWW.EXAMPLE.COM     9998

 (Screen 4)
 ISCUser   MbrName CICSApplid  CICSPort CICSNetid
 IMSUS01   HWS1    CICSA1      7777     CICSNET
 IMSUS02   HWS1    CICSA1      7777     CICSNET
```

OM API input:
```
CMD ( QUERY IMSCON TYPE(ISCUSER) NAME(IMSUS*) SHOW(ALL) )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2012.137 23:27:03.321818</statime>
<stotime>2012.137 23:27:03.323791</stotime>
<staseq>C993FF0D9C8DA01C</staseq>
<stoseq>C993FF0D9D08F69C</stoseq>
<rqsttkn1>USRT001 10162703</rqsttkn1>
<rc>00000000</rc>
```

```
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QRY IMSCON TYPE(ISCUSER) NAME(IMSUS*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="IUSER" llbl="ISCUser" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RCL" llbl="RecvClnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="a" key="3"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="ISTT" llbl="ISCStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="NODE" llbl="Node" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LIMS" llbl="LclIMS" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IMEM" llbl="IpMember" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RCICS" llbl="RmtCics" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="HOST" llbl="HostName" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CICAP" llbl="CICSApplid" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CICPT" llbl="CICSPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CICNT" llbl="CICSNetid" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>IUSER(IMSUS01 ) MBR(HWS1           ) CC(   0) SCL(ISC11111)
 RCL(ISC33333) STT(ACTIVE   ) ISC(CICS     ) ISTT(ACTIVE   )
NODE(CICSTRM1) LIMS(IMS1    ) IMEM(ICON1            )
IMSPLX(PLEX1          ) RCICS(ICONNT1 ) IP(127.0.0.1)
 HOST(WWW.EXAMPLE.COM) PORT(9998) CICAP(CICSA1  ) CICPT(7777    )
CICNT(CICSNET) </rsp>
<rsp>IUSER(IMSUS02 ) MBR(HWS1           ) CC(   0) SCL(ISC22222)
 RCL(ISC44444) STT(ACTIVE   ) ISC(CICS     ) ISTT(ACTIVE   )
NODE(CICSTRM1) LIMS(IMS1    ) IMEM(ICON1            )
IMSPLX(PLEX1          ) RCICS(ICONNT1 ) IP(127.0.0.1)
 HOST(WWW.EXAMPLE.COM) PORT(9998) CICAP(CICSA1  ) CICPT(7777    )
CICNT(CICSNET) </rsp>
</cmdrspdata>
</imsout>
```

Related reference:

ISC statement (System Definition)

RMTCICS statement (System Definition)

QUERY IMSCON TYPE(LINK) command

Use the QUERY IMSCON TYPE(LINK) command to display the status of one or more MSC logical links defined to IMS Connect.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 200
- "Usage notes" on page 202
- "Equivalent WTOR and z/OS commands" on page 202
- "Output fields" on page 202
- "Return, reason, and completion codes" on page 204
- "Examples" on page 205

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
      ┌────────────────────┐                              ┌─STATUS(ACTIVE)─┐
                           ,                                              
                     ┌─────▼──────┐
      └─SHOW(──▼──┬──ALL──────┬──)─┘
                  ├──GENIMSID──┤
                  ├──HOSTNAME──┤
                  ├──IMSPLEX───┤
                  ├──IPADDR────┤
                  ├──IPMEMBER──┤
                  ├──LCLIMS────┤
                  ├──MSC───────┤
                  ├──MSTATUS───┤
                  ├──PARTNER───┤
                  ├──PORT──────┤
                  ├──RECVCLNT──┤
                  ├──RMTIMS────┤
                  ├──RMTIMSCON─┤
                  ├──RMTPLKID──┤
                  ├──SENDCLNT──┤
                  └──STATUS────┘
```
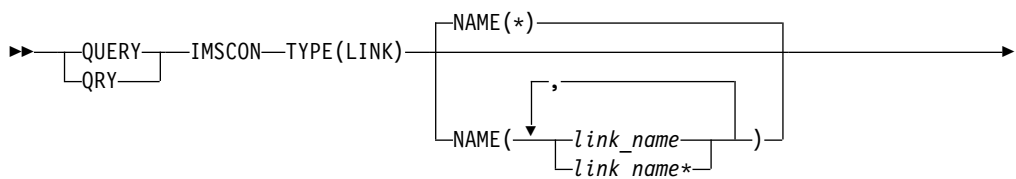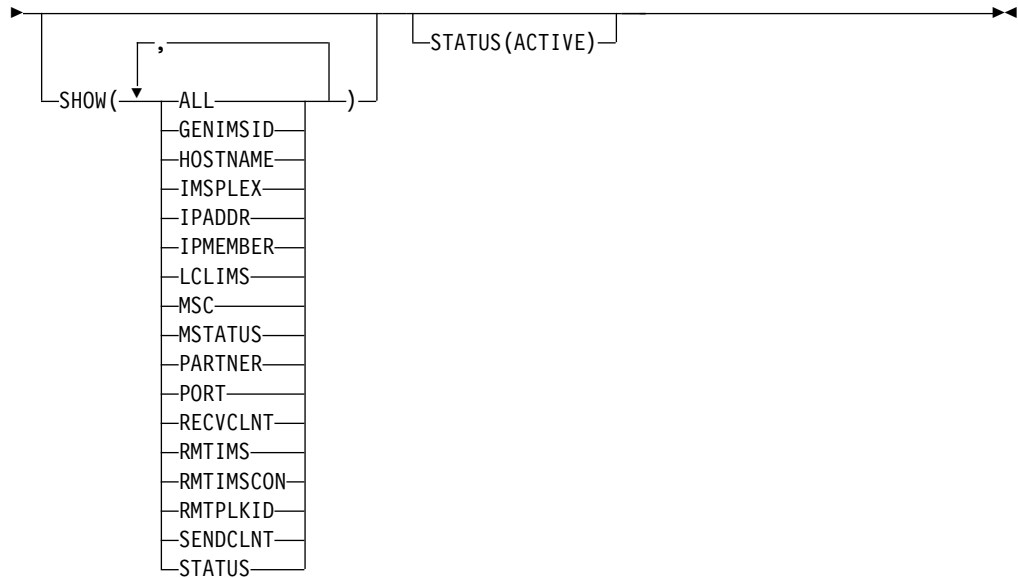
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(LINK)
command.

NAME

Specifies one or more MSC logical links to be displayed. You can specify a
single MSC logical link name or a list of MSC logical link names separated by
commas. Wildcards can be used in the names.

You can specify NAME(*) to display all MSC logical link definitions. NAME(*)
is the default.

SHOW

Specifies the optional output fields to be displayed. Output fields that are
always displayed, regardless of whether SHOW is specified, include the MSC
logical link name, the name of the IMS Connect that processes the command,
and the completion code.

The filters that are supported with the SHOW keyword, which can be specified
in any order, are:

ALL

Displays all output fields.

GENIMSID

Displays the name of the generic IMS as defined in the GENIMSID
parameter of the MSC statement in the IMS Connect configuration
member. If there is any active MSC logical link, specifying this keyword
also displays the IMS ID that the MSC logical link has affinity with.

HOSTNAME

Displays the host name of the remote IMS Connect. The remote IMS
Connect is defined in the RMTIMSCON parameter of the MSC statement
in the IMS Connect configuration member.

IMSPLEX

Displays the name of the IMSplex, as defined in the TMEMBER subparameter of the IMSPLEX parameter of the MSC statement in the IMS Connect configuration member.

IPADDR

Displays the IP address of the remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.

IPMEMBER

Displays the IMSplex member name, as defined in the MEMBER subparameter of the IMSPLEX parameter of the MSC statement in the IMS Connect configuration member. This name is the IMS Connect member name registered to the IMSPlex.

LCLIMS

Displays the name of the local IMS as defined in the LCLIMS parameter of the MSC statement in the IMS Connect configuration member.

MSC

Displays the name of the MSC physical link associated with this logical link, as defined by the MSC statement in the IMS Connect configuration member.

MSTATUS

Displays the status of the MSC physical link associated with the logical link. The status can be either ACTIVE or NOTACTIVE.

PARTNER

Displays the name of the partner ID associated with the logical link.

PORT

Displays the port of the associated remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.

RECVCLNT

Displays the receive client name, which is the name of the client ID of the remote IMS Connect that this IMS Connect receives messages from for this logical link.

RMTIMS

Displays the name of the remote IMS as defined in the RMTIMS parameter of the MSC statement in the IMS Connect configuration member.

RMTIMSCON

Displays the name of the remote IMS Connect connection as defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.

RMTPLKID

Displays the name of the remote MSC physical link ID as defined in the RMTPLKID parameter of the MSC statement in the IMS Connect configuration member.

SENDCLNT

Displays the send client name, which is the name of the client ID that IMS Connect uses to send messages to the remote IMS Connect for this logical link.

STATUS
> Displays the status of the logical link. The status can be ACTIVE, which indicates that the logical link is active.

STATUS
> Selects logical links for display that possess the following status. When the STATUS filter is specified, status information is displayed even if SHOW(STATUS) is not specified. The filter supported with the STATUS keyword is:

> **ACTIVE**
>> Selects logical links that have a status of ACTIVE.

Usage notes

You can issue the QUERY IMSCON TYPE(LINK) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Equivalent WTOR and z/OS commands

There are no equivalent WTOR and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(LINK) command.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 61. Output fields for the QUERY IMSCON TYPE(LINK) command

Short label	Long label	Keyword	Meaning
AFFIN	Affinity	GENIMSID	The name of the IMS with which the MSC logical link has affinity.
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 63 on page 204.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
GIMS	GenIMSID	GENIMSID	The name of the generic IMS as defined in the GENIMSID parameter of the MSC statement in the IMS Connect configuration member.

Table 61. Output fields for the QUERY IMSCON TYPE(LINK) command (continued)

Short label	Long label	Keyword	Meaning
HOST	HostName	HOSTNAME	The host name of the remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.
IMEM	IpMember	IPMEMBER	The IMSplex member name, as defined in the MEMBER subparameter of the IMSPLEX parameter of the MSC statement in the IMS Connect configuration member. This name is the IMS Connect member name registered to the IMSPlex.
IP	IpAddress	IPADDR	The IP address of the remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.
IMSPLX	IMSplex	IMSPLEX	The name of the IMSplex, as defined in the TMEMBER subparameter of the IMSPLEX parameter of the MSC statement in the IMS Connect configuration member.
LIMS	LclIMS	LCLIMS	The name of the local IMS as defined in the LCLIMS parameter of the MSC statement in the IMS Connect configuration member.
LINK	Link	N/A	The name of the MSC logical link. The name of the logical link is always returned.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
MSC	MscName	MSC	Name of the MSC physical link as defined in the LCLPLKID parameter of the MSC configuration statement in the IMS Connect configuration member.

MSTT	MscStatus	MSTATUS	For the MSC physical link associated with the logical link, this field indicates the status or state of the physical link. The value can be one of the following:
			
			ACTIVE
			    The physical link is active.
			
			NOTACTIVE
			    The physical link is not active.

PID	Partner	PARTNER	The name of the partner ID for the logical link.
PORT	Port	PORT	The port of the associated remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.
RCL	RecvClnt	RECVCLNT	The receive client name, which is the name of the client ID of the remote IMS Connect that this IMS Connect receives messages from for this logical link.
RIC	RmtImsCon	RMTIMSCON	The name of the remote IMS Connect connection as defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.

Table 61. Output fields for the QUERY IMSCON TYPE(LINK) command (continued)

Short label	Long label	Keyword	Meaning
RIMS	RmtIMS	RMTIMS	The name of the remote IMS as defined in the RMTIMS parameter of the MSC statement in the IMS Connect configuration member.
RPLK	RmtPlkID	RMTPLKID	The name of the remote MSC physical link ID as defined in the RMTPLKID parameter of the MSC statement in the IMS Connect configuration member.
SCL	SendClnt	SENDCLNT	The send client name, which is the name of the client ID that IMS Connect uses to send messages to the remote IMS Connect for this logical link.
STT	Status	STATUS	Status of the logical link. The status can be: **ACTIVE** The logical link is active.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(LINK) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 62. Return and reason codes for the QUERY IMSCON TYPE(LINK) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(LINK) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 63. Completion codes for the QUERY IMSCON TYPE(LINK) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(LINK) command completed successfully for the resources.

Table 63. Completion codes for the QUERY IMSCON TYPE(LINK) command (continued)

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for QUERY IMSCON TYPE(LINK) command

TSO SPOC input:

```
QUERY IMSCON TYPE(LINK) NAME(LINK12*) SHOW(ALL)
```

TSO SPOC output:

```
(Screen 1)
Link    MbrName Partner  SendClnt RecvClnt Status
LINK12A HWS1 AA MSC11111 MSC33333 ACTIVE
LINK12B HWS1 BB MSC22222 MSC44444 ACTIVE
LINK12C HWS1 CC MSC55555 MSC77777 ACTIVE

(Screen 2)
Link    MbrName MscName MscStatus RmtPlkId LclIMS LclIMS2 RmtIMS
LINK12A HWS1    MSC12   ACTIVE    MSC21    IMS1   IMS3    IMS2
LINK12B HWS1    MSC12   ACTIVE    MSC21    IMS1   IMS3    IMS2
LINK12C HWS1    MSC12   ACTIVE    MSC21    IMS1   IMS3    IMS2

(Screen 3)
Link    MbrName GenIMSID Affinity IpMember IMSplex RmtImsCon
LINK12A HWS1    IMS      IMS1     ICON1    PLEX1   CONNECT2
LINK12B HWS1    IMS      IMS1     ICON1    PLEX1   CONNECT2
LINK12C HWS1    IMS      IMS1     ICON1    PLEX1   CONNECT2

(Screen 4)
Link    MbrName IpAddress       HostName     Port
LINK12A HWS1    010.100.200.002 ICON.IBM.COM 5555
LINK12B HWS1    010.100.200.002 ICON.IBM.COM 5555
LINK12C HWS1    010.100.200.002 ICON.IBM.COM 5555
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(LINK) NAME(LINK12*) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.297 23:33:43.374620</statime>
<stotime>2010.297 23:33:43.375723</stotime>
<staseq>C6C7597265B1C3F7</staseq>
<stoseq>C6C7597265F6BFB7</stoseq>
<rqsttkn1>USRT001 10163343</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
```

```
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON         </kwd>
<input>QRY IMSCON TYPE(LINK) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="LINK" llbl="Link" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PID" llbl="Partner" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RCL" llbl="RecvClnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MSC" llbl="MscName" scope="LCL" sort="a" key="3"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MSTT" llbl="MscStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="9" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RPLK" llbl="RmtPlkID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LIMS" llbl="LclIMS" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RIMS" llbl="RmtIMS" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="GIMS" llbl="GenIMSID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AFFIN" llbl="Affin" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="IMEM" llbl="IpMember" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RIC" llbl="RmtImsCon" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="HOST" llbl="HostName" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="INT" align="right" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>LINK(LINK12A ) MBR(HWS1            ) CC(   0) PID(AA )
 SCL(MSC11111) RCL(MSC33333) STT(ACTIVE   ) MSC(MSC12  ) MSTT(ACTIVE
 ) RPLK(MSC21  ) LIMS(IMS1   ) RIMS(IMS2    ) GIMS(IMS     ) AFFIN(IMS1    )
 IMEM(ICON1           ) IMSPLX(PLEX1          ) RIC(CONNECT2)
 IP(010.100.200.002) HOST(ICON.IBM.COM) PORT(5555) </rsp>
<rsp>LINK(LINK12B ) MBR(HWS1            ) CC(   0) PID(BB )
 SCL(MSC22222) RCL(MSC44444) STT(ACTIVE   ) MSC(MSC12  ) MSTT(ACTIVE
 ) RPLK(MSC21  ) LIMS(IMS1   ) RIMS(IMS2    ) GIMS(IMS     ) AFFIN(IMS1    )
 IMEM(ICON1           ) IMSPLX(PLEX1          ) RIC(CONNECT2)
 IP(010.100.200.002) HOST(ICON.IBM.COM) PORT(5555) </rsp>
<rsp>LINK(LINK12C ) MBR(HWS1            ) CC(   0) PID(CC )
 SCL(MSC55555) RCL(MSC77777) STT(ACTIVE   ) MSC(MSC12  ) MSTT(ACTIVE
 ) RPLK(MSC21  ) LIMS(IMS1   ) RIMS(IMS2    ) GIMS(IMS     ) AFFIN(IMS1    )
```

```
    IMEM(ICON1          ) IMSPLX(PLEX1           ) RIC(CONNECT2)
 IP(010.100.200.002) HOST(ICON.IBM.COM) PORT(5555) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: MSC12 is a valid MSC physical link definition. It is currently active, with three associated logical links. Two of the logical links are currently active.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

MSC statement (System Definition)

QUERY IMSCON TYPE(MSC) command

Use the QUERY IMSCON TYPE(MSC) command to display the status of one or more MSC physical links defined to IMS Connect.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 208
- "Usage notes" on page 210
- "Equivalent WTOR and z/OS commands" on page 210
- "Output fields" on page 211
- "Return, reason, and completion codes" on page 213
- "Examples" on page 214

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►─┬──────────────────────────────────┬──┬───────────────────────────────┬─►◄
   │              ┌─,───────────┐      │  │            ┌─,────────────┐    │
   └─SHOW(─▼─┬─ALL────────┬─)───┘      └─STATUS(─▼─┬─ACTIVE───┬─)────┘
            ├─GENIMSID────┤                        └─NOTACTIVE─┘
            ├─HOSTNAME────┤
            ├─IMSPLEX─────┤
            ├─IPADDR──────┤
            ├─IPMEMBER────┤
            ├─LCLIMS──────┤
            ├─LINK────────┤
            ├─PORT────────┤
            ├─RMTIMS──────┤
            ├─RMTIMSCON───┤
            ├─RMTPLKID────┤
            └─STATUS──────┘
```
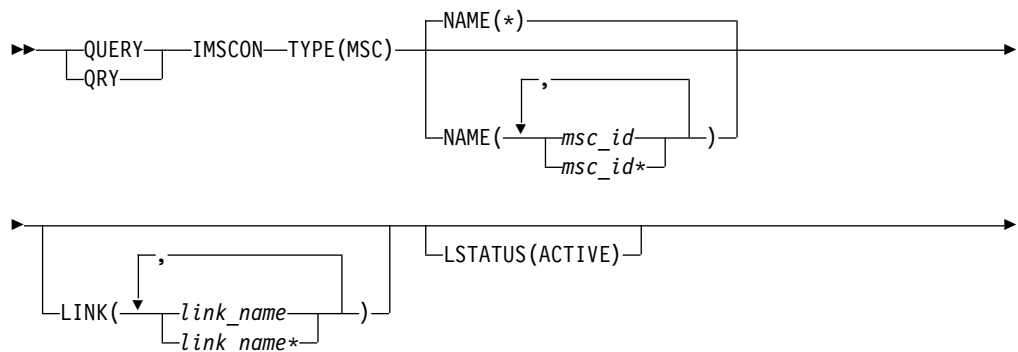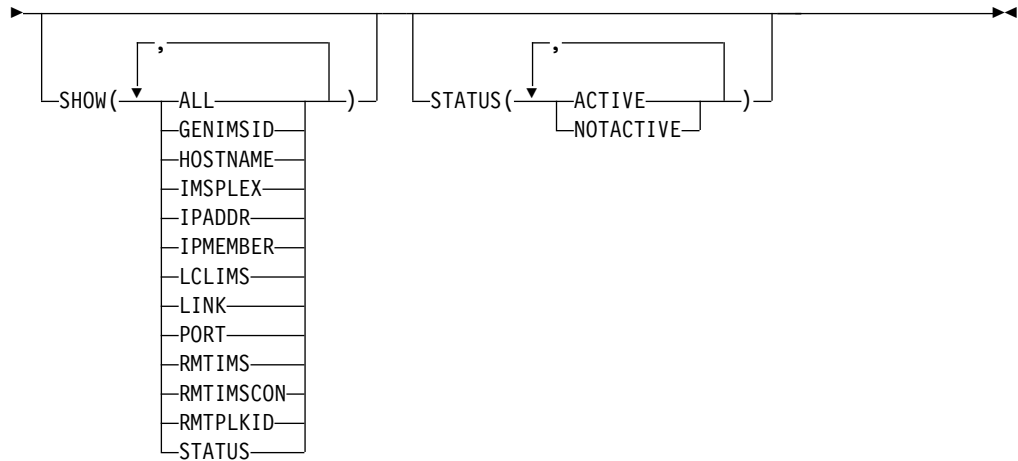
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(MSC) command.

LINK

Selects MSC physical link resources for display that have one of the specified logical links defined. You can specify a single logical link name or a list of logical link names separated by commas. Wildcards can be used in the names.

Only those MSC physical link resources that have a specified logical link are displayed. MSC physical link resources that match the NAME() parameter, but do not have the specified logical link, are not displayed.

When the LINK filter is specified, logical link information is displayed even if SHOW(LINK) is not specified. Logical links that match the names specified by the LINK filter are displayed. Logical links that are associated with the MSC physical link, but which do not match the filter, are not displayed.

LSTATUS

Selects MSC physical link resources for display that have at least one logical link that is in the following state:

ACTIVE

This logical link is active.

MSC physical links that match the NAME() parameter, but do not have such a logical link, are not displayed.

When the LSTATUS filter is specified, logical link information is displayed even if SHOW(LINK) is not specified. Logical links that are in this state are displayed.

NAME

Specifies one or more MSC physical links to be displayed. You can specify a single MSC physical link name or a list of MSC physical link names separated by commas. Wildcards can be used in the name.

You can specify NAME(*) to display all MSC physical link definitions. NAME(*) is the default.

SHOW

Specifies the optional output fields to be displayed. Output fields that are

always displayed, regardless of whether SHOW is specified, include the MSC physical link name, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL
Displays all output fields.

GENIMSID
Displays the name of the generic IMS as defined in the GENIMSID parameter of the MSC statement in the IMS Connect configuration member. If there is any active MSC logical link, specifying this keyword also displays the IMS ID that the MSC logical link has affinity with.

HOSTNAME
Displays the host name of the remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.

IMSPLEX
Displays the name of the IMSplex, as defined in the TMEMBER subparameter of the IMSPLEX parameter of the MSC statement in the IMS Connect configuration member.

IPADDR
Displays the IP address of the remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.

IPMEMBER
Displays the IMSplex member name, as defined in the MEMBER subparameter of the IMSPLEX parameter of the MSC statement in the IMS Connect configuration member. This is the IMS Connect member name registered to the IMSPlex.

LCLIMS
Displays the name of the local IMS as defined in the LCLIMS parameter of the MSC statement in the IMS Connect configuration member.

LINK
Displays information and status for logical links associated with the specified MSC physical link. If the MSC physical link has one or more associated logical links, the command output includes information about each logical link. A separate output line is displayed for each logical link. This output line is in addition to any output lines that display general MSC physical link information. If there are no logical links, no additional output line is displayed.

Information displayed for the logical link includes:
- The name of the logical link.
- The name of the partner ID.
- The receive client name, which is the name of the client ID of the remote IMS Connect that this IMS Connect receives messages from for this logical link.
- The send client name, which is the name of the client ID that IMS Connect uses to send messages to the remote IMS Connect for this logical link.
- Status or state of the logical link, which is the following:

> **ACTIVE**
>> This logical link connection is active.
>
> **PORT**
>> Displays the port of the associated remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.
>
> **RMTIMS**
>> Displays the name of the remote IMS as defined in the RMTIMS parameter of the MSC statement in the IMS Connect configuration member.
>
> **RMTIMSCON**
>> Displays the name of the remote IMS Connect connection as defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.
>
> **RMTPLKID**
>> Displays the name of the remote MSC physical link ID as defined in the RMTPLKID parameter of the MSC statement in the IMS Connect configuration member.
>
> **STATUS**
>> Displays the status of the MSC physical link. For a description of the possible state returned, see the STATUS keyword in Table 65 on page 211.

STATUS
> Selects MSC physical links for display that possess at least one of the specified statuses. When the STATUS filter is specified, status information is displayed even if SHOW(STATUS) is not specified. The filters supported with the STATUS keyword, which can be specified in any order, are:
>
> **ACTIVE**
>> Selects MSC physical links that have a status of ACTIVE.
>
> **NOTACTIVE**
>> Selects MSC physical links that have a status of NOTACTIVE.

Usage notes

You can issue the QUERY IMSCON TYPE(MSC) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Typically, this command results in one output display line for each MSC physical link definition being displayed. However, if the SHOW(LINK) keyword is specified, an additional line is displayed for each logical link associated with the specified MSC definition.

Equivalent WTOR and z/OS commands

The following table lists WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(MSC) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.

- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 64. WTOR and IMS Connect z/OS equivalents for the QUERY IMSCON TYPE(MSC) command

QUERY IMSCON TYPE(MSC) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
QUERY IMSCON TYPE(MSC) NAME(*) SHOW(ALL | *show_parm*)	VIEWMSC ALL	QUERY MSC NAME(*)
QUERY IMSCON TYPE(MSC) NAME(*msc_id*) SHOW(ALL | *show_parm*)	VIEWMSC *msc_id*	QUERY MSC NAME(*msc_id*)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 65. Output fields for the QUERY IMSCON TYPE(MSC) command

Short label	Long label	Keyword	Meaning
AFFIN	Affinity	GENIMSID	The name of the IMS that the MSC logical link has affinity with.
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 67 on page 213.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
GIMS	GenIMSID	GENIMSID	The name of the generic IMS as defined in the GENIMSID parameter of the MSC statement in the IMS Connect configuration member.
HOST	HostName	HOSTNAME	The host name of the remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.
IMEM	IpMember	IPMEMBER	The IMSplex member name, as defined in the MEMBER subparameter of the IMSPLEX parameter of the MSC statement in the IMS Connect configuration member. This name is the IMS Connect member name registered to the IMSPlex.

Table 65. Output fields for the QUERY IMSCON TYPE(MSC) command (continued)

Short label	Long label	Keyword	Meaning
IP	IpAddress	IPADDR	The IP address of the remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.
IMSPLX	IMSplex	IMSPLEX	The name of the IMSplex, as defined in the TMEMBER subparameter of the IMSPLEX parameter of the MSC statement in the IMS Connect configuration member.
LIMS	LclIMS	LCLIMS	The name of the local IMS as defined in the LCLIMS parameter of the MSC statement in the IMS Connect configuration member.
LIMS2	LclIMS2	LCLIMS	The name of the second local IMS in an XRF pair, as defined in the LCLIMS parameter of the MSC statement in the IMS Connect configuration member.
LINK	Link	LINK	The name of the logical link associated with the MSC physical link.
LSTT	LinkStatus	LINK	For a logical link associated with the physical link, this is the status or state of the logical link, which is the following: **ACTIVE** The logical link is active.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
MSC	MscName	N/A	Name of the MSC physical link as defined in the LCLPLKID parameter of the MSC configuration statement in the IMS Connect configuration member. The name of the MSC physical link is always returned.
PID	Partner	LINK	The name of the partner ID for a logical link associated with the MSC physical link.
PORT	Port	PORT	The port of the associated remote IMS Connect. The remote IMS Connect is defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.
RCL	RecvClnt	LINK	For a logical link associated with the physical link, this field shows the receive client name, which is the name of the client ID of the remote IMS Connect that this IMS Connect receives messages from for this logical link.
RIC	RmtImsCon	RMTIMSCON	The name of the remote IMS Connect connection as defined in the RMTIMSCON parameter of the MSC statement in the IMS Connect configuration member.
RIMS	RmtIMS	RMTIMS	The name of the remote IMS as defined in the RMTIMS parameter of the MSC statement in the IMS Connect configuration member.
RPLK	RmtPlkID	RMTPLKID	The name of the remote MSC physical link ID as defined in the RMTPLKID parameter of the MSC statement in the IMS Connect configuration member.

Table 65. Output fields for the QUERY IMSCON TYPE(MSC) command (continued)

Short label	Long label	Keyword	Meaning
SCL	SendClnt	LINK	For a logical link associated with the physical link, this field shows the send client name, which is the name of the client ID that IMS Connect uses to send messages to the remote IMS Connect for this logical link.
STT	Status	STATUS	Status of the MSC physical link. The status can be one of the following: **ACTIVE** The physical link is active. **NOTACTIVE** The physical link is not active.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(MSC) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 66. Return and reason codes for the QUERY IMSCON TYPE(MSC) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(MSC) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 67. Completion codes for the QUERY IMSCON TYPE(MSC) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(MSC) command completed successfully for the resources.

Table 67. Completion codes for the QUERY IMSCON TYPE(MSC) command (continued)

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for QUERY IMSCON TYPE(MSC) command

TSO SPOC input:

```
QUERY IMSCON TYPE(MSC) NAME(MSC12) SHOW(ALL)
```

TSO SPOC output:

```
(Screen 1)
MscName MbrName CC RmtPlkId LclIMS LclIMS2 RmtIMS GenIMSID
MSC12   HWS1    0  MSC21    IMS1   IMS3    IMS2   IMS
MSC12   HWS1    0
MSC12   HWS1    0
MSC12   HWS1    0

(Screen 2)
MscName MbrName Affinity IpMember IMSplex RmtImsCon
MSC12   HWS1    IMS1     ICON1    PLEX1   CONNECT2
MSC12   HWS1
MSC12   HWS1
MSC12   HWS1

(Screen 3)
MscName MbrName IpAddress       HostName      Port Status
MSC12   HWS1    010.100.200.002 ICON2.IBM.COM 5555 ACTIVE
MSC12   HWS1
MSC12   HWS1
MSC12   HWS1

(Screen 4)
MscName MbrName Link    Partner SendInt  RecvClnt LinkStatus
MSC12   HWS1
MSC12   HWS1    LINK12A AA      MSC11111 MSC33333 ACTIVE
MSC12   HWS1    LINK12B BB      MSC22222 MSC44444 ACTIVE
MSC12   HWS1    LINK12C CC      MSC55555 MSC77777 ACTIVE
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(MSC) NAME(MSC12) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.297 23:48:48.614927</statime>
<stotime>2010.297 23:48:48.616017</stotime>
<staseq>C6C75CD1B3A0FC8A</staseq>
<stoseq>C6C75CD1B3E51F4A</stoseq>
```

```
<rqsttkn1>USRT001 10164848</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON         </kwd>
<input>QRY IMSCON TYPE(MSC) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSC" llbl="MscName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RPLK" llbl="RmtPlkID" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LIMS" llbl="LclIMS" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RIMS" llbl="RmtIMS" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="GIMS" llbl="GenIMSID" scope="GEN" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AFFIN" llbl="Affin" scope="GEN" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="IMEM" llbl="IpMember" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RIC" llbl="RmtImsCon" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="HOST" llbl="HostName" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="INT" align="right" skipb="no" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LINK" llbl="Link" scope="LCL" sort="a" key="3" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="PID" llbl="Partner" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RCL" llbl="RecvClnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LSTT" llbl="LinkStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="9" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSC(MSC12   ) MBR(HWS1            ) CC(   0) LINK(LINK12A ) PID(AA )
 SCL(MSC11111) RCL(MSC33333) LSTT(ACTIVE   ) </rsp>
<rsp>MSC(MSC12   ) MBR(HWS1            ) CC(   0) LINK(LINK12B ) PID(BB )
 SCL(MSC22222) RCL(MSC44444) LSTT(ACTIVE   ) </rsp>
<rsp>MSC(MSC12   ) MBR(HWS1            ) CC(   0) LINK(LINK12C ) PID(CC )
 SCL(MSC55555) RCL(MSC77777) LSTT(ACTIVE   ) </rsp>
<rsp>MSC(MSC12   ) MBR(HWS1            ) CC(   0) RPLK(MSC21   )
 LIMS(IMS1   ) RIMS(IMS2    ) GIMS(IMS      ) AFFIN(IMS1     )
```

```
    IMEM(ICON1          ) IMSPLX(PLEX1          ) RIC(CONNECT2)
    IP(010.100.200.002) HOST(ICON2.IBM.COM) PORT(5555) STT(ACTIVE  ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: MSC12 is a valid MSC physical link definition. It is currently active, with three associated logical links. Two of the logical links are currently active.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

MSC statement (System Definition)

QUERY IMSCON TYPE(ODBM) command

Use the QUERY IMSCON TYPE(ODBM) command to display the status, alias, and activity of one or more ODBMs defined to IMS Connect.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 217
- "Usage notes" on page 218
- "Equivalent WTOR and z/OS commands" on page 219
- "Output fields" on page 219
- "Return, reason, and completion codes" on page 220
- "Examples" on page 221

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
    ┌─────────────┐                    ┌────────────────┐
►───┤             ├──────────────────┬─┤                ├──────────────────►◄
    │  ┌─,────┐   │                  │ │  ┌─,──────────┐│
    └──SHOW(─▼──ALL───┬─)─┘          └─STATUS(─▼─REGISTERED─┬─)─┘
             ├──ALIAS──┤                       └─NOTACTIVE──┘
             ├──RRS────┤
             └──STATUS─┘
```
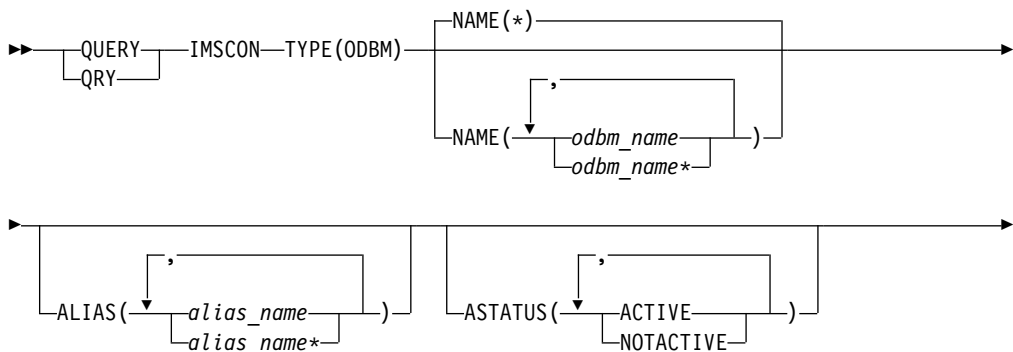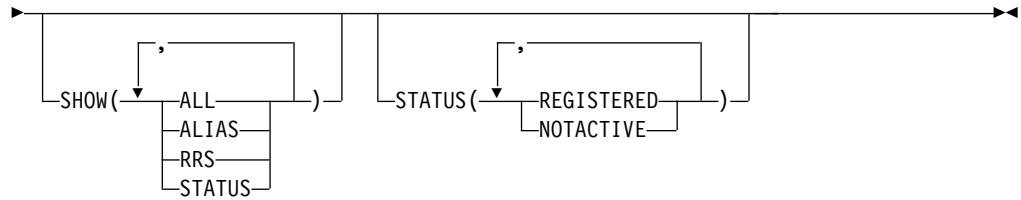
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(ODBM) command.

ALIAS

Selects ODBM resources for display that have one of the specified aliases defined on the port. You can specify a single alias name, or a list of alias names separated by commas. Wildcards can be used in the names.

Only those ODBM resources that have a specified alias are displayed. ODBMs that match the NAME() parameter, but do not have the specified alias, are not displayed.

When the ALIAS filter is specified, alias information is displayed even if SHOW(ALIAS) is not specified. Aliases that match the names specified by the ALIAS filter are displayed. Aliases that are associated with ODBM, but which do not match the filter, are not displayed.

ASTATUS

Selects ODBM resources for display that have at least one alias that is in one of the specified states. You can specify a single alias status, or a list of alias statuses separated by commas.

The filters supported with the ASTATUS keyword, which can be specified in any order, are:

ACTIVE

The alias is active.

NOTACTIVE

The alias is not active in IMS Connect, ODBM, or both.

Only those ODBM resources that have an alias that is in one of the specified states are displayed. ODBMs that match the NAME() parameter, but do not have such an alias, are not displayed.

When the ASTATUS filter is specified, alias information is displayed even if SHOW(ALIAS) is not specified. Aliases that are in one of the specified states are displayed.

NAME

Specifies one or more ODBM resources to be displayed. You can specify a single ODBM name or a list of ODBM names separated by commas. Wildcards can be used in the name.

You can specify NAME(*) to display all ODBMs. NAME(*) is the default.

SHOW

Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the ODBM name, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL
Displays all output fields.

ALIAS
Displays the alias name of an IMS data store defined to the instance of ODBM. If an ODBM has one or more aliases, the command output includes information about each alias. A separate output line is displayed for each alias. This output line is in addition to any output lines that display general ODBM information. If the ODBM resource has no aliases, no additional output line is displayed.

Information displayed for the alias includes:

- Name of the alias
- Status of the alias

For a description of the possible alias status returned, see the ALIAS keyword in Table 68 on page 219.

RRS
Displays whether ODBM is using the z/OS Resource Recovery Services (RRS). When IMS Connect registers with ODBM, the ODBM provides IMS Connect with its RRS setting.

STATUS
Displays the status of the ODBM. For a description of the possible status returned, see the STATUS keyword in Table 68 on page 219.

STATUS
Selects ODBMs for display that possess at least one of the specified statuses. When the STATUS filter is specified, status information is displayed even if SHOW(STATUS) is not specified.

The filters supported with the STATUS keyword, which can be specified in any order, are:

REGISTERED
Selects ODBMs that have a status of REGISTERED, which means that this instance of IMS Connect is registered with the ODBM.

NOTACTIVE
Selects ODBMs that have a status of NOTACTIVE, which means that the ODBM is not active.

Usage notes

You can issue the QUERY IMSCON TYPE(ODBM) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Typically, this command results in one output display line for each ODBM displayed. However, if the SHOW(ALIAS) keyword is specified, an additional line is displayed for each alias associated with the specified ODBM.

Equivalent WTOR and z/OS commands

There are no equivalent WTOR and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(ODBM) command.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
Provides a brief description of the output field.

Table 68. Output fields for the QUERY IMSCON TYPE(ODBM) command

Short label	Long label	Keyword	Meaning
ALIAS	AliasName	ALIAS	The alias name of an IMS data store defined to the instance of ODBM.
ASTT	AStatus	ALIAS	Status or state of the alias, which is one of the following: **ACTIVE** 　The alias is active. **NOTACTIVE(IMSCON)** 　The alias has been deactivated in IMS Connect by using the STOPIA or an equivalent command. **NOTACTIVE(ODBM)** 　The alias has been deactivated in ODBM by using the ODBM type-2 UPDATE ODBM STOP(CONNECTION) ALIAS command. **NOTACTIVE(IMSCON,ODBM)** 　The alias has been deactivated both in IMS Connect (by using the STOPIA or an equivalent command) and in ODBM (by using the ODBM type-2 UPDATE ODBM STOP(CONNECTION) ALIAS command).
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 70 on page 220.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
ODBM	ODBMName	N/A	Name of the ODBM. The ODBM name is always returned.

Table 68. Output fields for the QUERY IMSCON TYPE(ODBM) command (continued)

Short label	Long label	Keyword	Meaning
RRS	RRS	RRS	Indicates the RRS specification in an ODBM. When IMS Connect registers with ODBM, the ODBM provides IMS Connect with its RRS setting. The specification can be one of the following: **Y** ODBM is using RRS. **N** ODBM is not using RRS.
STT	Status	STATUS	Status of the ODBM. The status can be one of the following: **REGISTERED** This instance of IMS Connect is registered with the ODBM. **NOTACTIVE** ODBM is not active.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(ODBM) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 69. Return and reason codes for the QUERY IMSCON TYPE(ODBM) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(ODBM) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 70. Completion codes for the QUERY IMSCON TYPE(ODBM) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(ODBM) command completed successfully for the resources.

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for QUERY IMSCON TYPE(ODBM) command

TSO SPOC input:

```
QUERY IMSCON TYPE(ODBM) NAME(*) SHOW(ALL)
```

TSO SPOC output:

```
(Screen 1)
ODBMName MbrName CC  Status      RRS AliasName   AStatus
ODBMA    HWS1    0   REGISTERED  N
ODBMA    HWS1    0               IMS1        ACTIVE
ODBMA    HWS1    0               IMS2        NOTACTIVE(IMSCON)
ODBMB    HWS1    0   REGISTERED  N
ODBMB    HWS1    0               IMS1        ACTIVE
ODBMB    HWS1    0               IMS2        ACTIVE
ODBMC    HWS1    0   NOTACTIVE   N
ODBMD    HWS1    0   REGISTERED  N
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(ODBM) NAME(*) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 15:47:30.101469</statime>
<stotime>2010.298 15:47:30.102568</stotime>
<staseq>C6C8331A774DD430</staseq>
<stoseq>C6C8331A77928530</stoseq>
<rqsttkn1>USRT001 10084729</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON         </kwd>
<input>QRY IMSCON TYPE(ODBM) NAME(*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="ODBM" llbl="ODBMName" scope="LCL" sort="a" key="1"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
```

```
                 len="4" dtype="INT" align="right" skipb="no" />
            <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
             scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
            len="*" dtype="CHAR" align="left" skipb="no" />
            <hdr slbl="RRS" llbl="RRS" scope="LCL" sort="n" key="0" scroll="yes"
             len="1" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="ALIAS" llbl="AliasName" scope="LCL" sort="a" key="3"
             scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="ASTT" llbl="AStatus" scope="LCL" sort="n" key="0"
             scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
            </cmdrsphdr>
            <cmdrspdata>
            <rsp>ODBM(ODBMA   ) MBR(HWS1             ) CC(   0) ALIAS(IMS1)
             ASTT(ACTIVE) </rsp>
            <rsp>ODBM(ODBMA   ) MBR(HWS1             ) CC(   0) ALIAS(IMS2)
             ASTT(NOTACTIVE(IMSCON)) </rsp>
            <rsp>ODBM(ODBMA   ) MBR(HWS1             ) CC(   0) STT(REGISTERED
                 ) RRS(N) </rsp>
            <rsp>ODBM(ODBMB   ) MBR(HWS1             ) CC(   0) ALIAS(IMS1)
             ASTT(ACTIVE) </rsp>
            <rsp>ODBM(ODBMB   ) MBR(HWS1             ) CC(   0) ALIAS(IMS2)
             ASTT(ACTIVE) </rsp>
            <rsp>ODBM(ODBMB   ) MBR(HWS1             ) CC(   0) STT(REGISTERED
                 ) RRS(N) </rsp>
            <rsp>ODBM(ODBMC   ) MBR(HWS1             ) CC(   0) STT(NOTACTIVE
                 ) RRS(N) </rsp>
            <rsp>ODBM(ODBMD   ) MBR(HWS1             ) CC(   0) STT(REGISTERED
                 ) RRS(N) </rsp>
            </cmdrspdata>
            </imsout>
```

Explanation: There are four ODBM resources defined to IMS Connect. ODBMA and ODBMB have two aliases defined. ODBMC and ODBMD have no aliases. For each ODBM, the first line of output displays information and status related to the ODBM resource, while the output that follows displays information and status specific to each alias defined to the ODBM.

Related concepts:

⤷ How to interpret CSL request return and reason codes (System Programming APIs)

QUERY IMSCON TYPE(PORT) command

Use the QUERY IMSCON TYPE(PORT) command to display the status and activity of one or more ports defined to IMS Connect. The clients that are active on the specified ports can also be displayed.

Subsections:
- "Environment" on page 223
- "Syntax" on page 223
- "Keywords" on page 224
- "Usage notes" on page 227
- "Equivalent WTOR and z/OS commands" on page 228
- "Output fields" on page 228
- "Return and reason codes" on page 230
- "Completion codes" on page 231
- "Examples" on page 231

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
>>─┬─QUERY─┬──IMSCON──TYPE(PORT)──┬─NAME(*)──────────────────────────┬──>
   └─QRY───┘                      │          ┌─,───────────┐         │
                                  └─NAME(──▼─┬─LOCAL────┬──┴──)──────┘
                                            ├─portid───┤
                                            └─portid*──┘

>──┬──────────────────────────────┬──┬─────────────────────────────┬──>
   │       ┌─,───────────────┐     │  │          ┌─,──────┐         │
   └─CLIENT(──▼─┬─client_name──┬─┴──)─┘  └─CSTATUS(──▼─┬─CONN─┬──┴──)─┘
               └─client_name*─┘                      ├─CONV─┤
                                                     ├─READ─┤
                                                     ├─RECV─┤
                                                     ├─WFCM─┤
                                                     └─XMIT─┘

>──┬────────────────────────────────────┬──┬───────────────────────┬──>
   │          ┌─,─────────────────┐      │  │         ┌─,────────┐  │
   └─DATASTORE(──▼─┬─datastore_name──┬─┴──)─┘  └─SHOW(──▼─┬─ALL─────┬─┴──)─┘
                  └─datastore_name*─┘                    ├─CLIENT──┤
                                                         ├─EDITRTN─┤
                                                         ├─IDLETO──┤
                                                         ├─KEEPAV──┤
                                                         ├─NUMSOC──┤
                                                         ├─STATUS──┤
                                                         ├─SUMMARY─┤
                                                         └─TIMEOUT─┘

>──┬────────────────────────────┬──┬──────────────────────────────┬──>
   │        ┌─,──────────┐       │  │        ┌─,────────────┐       │
   └─STATUS(──▼─┬─ACTIVE────┬─┴──)─┘  └─TRAN(──▼─┬─trancode_name──┬─┴──)─┘
               └─NOTACTIVE─┘                    └─trancode_name*─┘

>──┬───────────────────────┬──><
   │        ┌─,───────┐     │
   └─USERID(──▼─┬─userid──┬─┴──)─┘
               └─userid*─┘
```
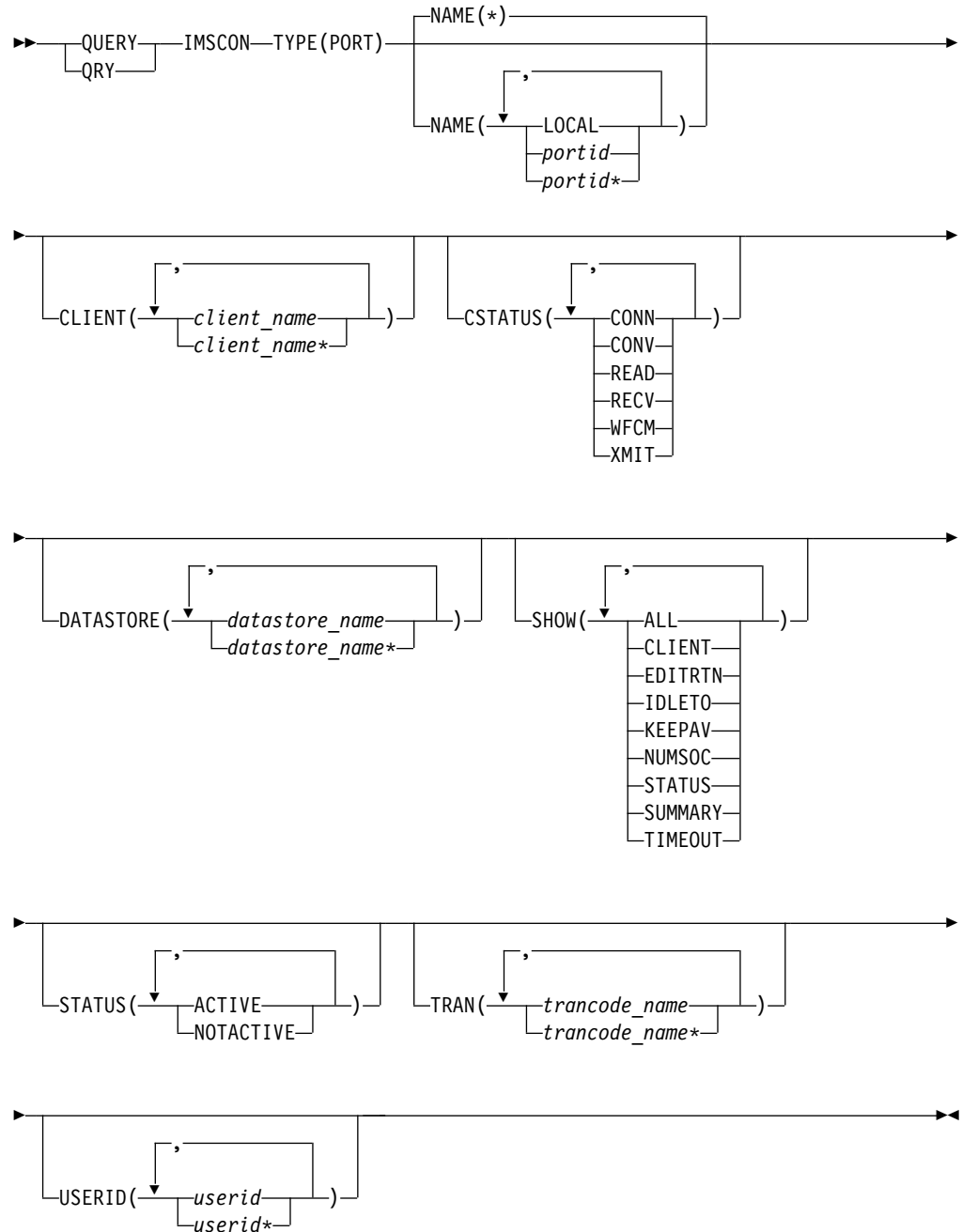
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(PORT) command.

CLIENT
> Selects ports for display that have one of the specified clients active on the port. You can specify a single client name or a list of client names separated by commas. Wildcards can be used in the names.
>
> Only those ports that have a specified client active are displayed. Ports that match the NAME() parameter, but do not have the specified client active, are not displayed.
>
> When the CLIENT filter is specified, client information is displayed even if SHOW(CLIENT) is not specified.
>
> Only those clients that match the names specified by the CLIENT filter are displayed. Clients that are active on the port, but which do not match the filter, are not displayed.

CSTATUS
> Selects ports for display that have at least one active client that is in one of the specified states. You can specify a single client status, or a list of client statuses separated by commas.
>
> The filters supported with the CSTATUS keyword, which can be specified in any order, are:
>
> **CONN**
> > Waiting for output from IMS.
>
> **CONV**
> > In a conversational state.
>
> **READ**
> > Reading an input message from the client.
>
> **RECV**
> > Waiting for input from the client (in other words, in a `receive` state).
>
> **WFCM**
> > Waiting for confirmation (ACK, NAK, or DEALLOCATE) from the client.
>
> **XMIT**
> > Sending data to the client.
>
> Only those ports that have an active client that is in one of the specified states are displayed. Ports that match the NAME() parameter, but do not have such an active client, are not displayed.
>
> When the CSTATUS filter is specified, client information is displayed even if SHOW(CLIENT) is not specified. Only those clients that are in one of the specified states are displayed.

DATASTORE
> Selects ports for display that have at least one active client with a transaction submitted to the specified data store. You can specify a single data store name or a list of data store names separated by commas. Wildcards can be used in the names.
>
> Only those ports that have an active client with a transaction submitted to the specified data store are displayed. Ports that match the NAME() parameter, but do not have such an active client, are not displayed.

When the DATASTORE filter is specified, client information is displayed even if SHOW(CLIENT) is not specified. Only those clients that have submitted a transaction to the specified data store are displayed.

NAME

Specifies one or more ports to be displayed. You can specify a single port number or a list of port numbers separated by commas. Wildcards can be used in the port numbers.

To display the local port used by the IMS TM Resource Adapter, specify NAME(LOCAL).

An SSL port is displayed with the character "S" appended to the end of the port number. To display the SSL port, specify the port number either with or without the character "S" appended to the end of the port number.

A port defined for ODBM use is displayed with the character "D" appended to the end of the port number. To display the ODBM port, specify the port number either with or without the character "D" appended to the end of the port number.

A port defined for an ISC link to a CICS subsystem is displayed with the character "C" appended to the end of the port number. To display a CICS port, specify the port number either with or without the character "C" appended to the end of the port number.

You can specify NAME(*) to display all ports. NAME(*) is the default.

SHOW

Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the port number, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL

Displays all output fields.

CLIENT

Displays the active client socket connections that are associated with the port. If a port has one or more active clients, the command output includes information about each client socket connection. A separate output line is displayed for each active client. This output line is in addition to any output lines that display general port information. If a port has no active clients, no additional output line is displayed.

To view client summary information, including the total number of clients active and in various states, specify the SUMMARY filter in addition to the CLIENT filter.

Information displayed for the client includes:
- Name of the client.
  IMS Connect uses the client name to identify the client socket connection. The name of an IMS Connect client can be provided by the client or, if the client does not provide it, IMS Connect randomly generates the name.
- User ID passed to IMS Connect.
- If the connection is used for MSC messages, the local MSC physical link ID.

- If a connection is used for ISC messages, the ISC link ID.
- Transaction code submitted by the client.
- Data store to which the transaction was submitted.
- Number of seconds that the client has been in the specified status.
- The client port number, which is a random number that TCP/IP generates to represent a connection from a client.
- The IP address being used by the connection of the client to IMS Connect.
- The APSB token for ODBM.
- Status or state of the thread of the client. For a description of the possible client status returned, see the CLIENT keyword in Table 72 on page 228.
- The alternate client ID specified by the client that issued the RESUME TPIPE request.
- The RESUME TPIPE token of the client that issued the RESUME TPIPE request.

EDITRTN
Displays the name of the Port Input/Output Edit Exit routine, as defined by the EDIT= parameter of the port definition in the IMS Connect configuration file.

IDLETO
Displays the amount of time open socket connections on this port can remain idle in RECV state before they are terminated due to inactivity.

The value displayed is the value of the IDLETO= parameter in the port definition in the IMS Connect configuration file.

The value shown is hundredths of seconds. Returned values can be from -1 to 2 147 483 647 (X'7FFFFFFF'). A value of 0 means this timeout function is disabled, which prevents inactive connections from timing out. A value of -1 means that the connections on the port use the global IDLETO value, because no IDLETO value was set for the port.

This port-level idle timeout value overrides the global-level idle timeout value.

KEEPAV
Displays the amount of time a connection remains idle before the z/OS TCP/IP layer sends a packet to maintain the connection. The value displayed is specified by the KEEPAV= keyword in either the TCPIP or the ODACCESS statement in the IMS Connect configuration file.

NUMSOC
Displays the number of active sockets that are used on a port. The number includes the active client sockets plus one listening socket.

STATUS
Displays the status of the port. For a description of the possible status returned, see the STATUS keyword in Table 72 on page 228.

SUMMARY
Displays summary information related to the clients active on the port. To view specific client information, specify the CLIENT filter in addition to the SUMMARY filter.

Information displayed includes:

- Total number of clients active on the port
- Total number of clients in RECV state
- Total number of clients in READ state
- Total number of clients in CONN state
- Total number of clients in XMIT state
- Total number of clients not in any of these states

TIMEOUT

Displays the amount of time IMS Connect waits before terminating a client connection if no messages are received from the client.

STATUS

Selects ports for display that possess at least one of the specified statuses. When the STATUS keyword is specified, status information is displayed even if SHOW(STATUS) is not specified.

The filters supported with the STATUS keyword, which can be specified in any order, are:

ACTIVE

Selects ports that are active.

NOTACTIVE

Selects ports that are not active.

TRAN

Selects ports for display that have at least one active client with a specified transaction submitted to a data store. You can specify a single transaction name or a list of transaction names separated by commas. Wildcards can be used in the names.

Only those ports that have an active client with a specified transaction are displayed. Ports that match the NAME() parameter, but do not have such an active client, are not displayed.

When the TRAN filter is specified, client information is displayed even if SHOW(CLIENT) is not specified. Only those clients that have submitted a specified transaction are displayed.

USERID

Selects ports for display that have one of the specified user IDs active on the port. You can specify a single *userid* name or a list of *userid* names separated by commas. Wildcards can be used in the user IDs.

Only those ports that have a specified user ID active are displayed. Ports that match the NAME() parameter, but do not have the specified user ID active, are not displayed.

When the USERID filter is specified, client information is displayed even if SHOW(CLIENT) is not specified. Only those clients that have a user ID specified by the CLIENT filter are displayed. Clients that are active on the port, but which do not have a user ID that matches the filter, are not displayed.

Usage notes

You can issue the QUERY IMSCON TYPE(PORT) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Typically, this command results in one output display line for each port displayed. However, if the SHOW(CLIENT) keyword is specified, an additional line is displayed for each client associated with the specified port.

Equivalent WTOR and z/OS commands

The following table lists WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(PORT) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 71. WTOR and IMS Connect z/OS equivalents for the QUERY IMSCON TYPE(PORT) command

QUERY IMSCON TYPE(PORT) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
QUERY IMSCON TYPE(PORT) NAME(*) SHOW(ALL | *show_parm*)	VIEWPORT ALL	QUERY PORT NAME(*) SHOW(ALL)
QUERY IMSCON TYPE(PORT) NAME(*portid*) SHOW(ALL | *show_parm*)	VIEWPORT *port_id*	QUERY PORT NAME(*port_id*) SHOW(ALL)
QUERY IMSCON TYPE(PORT) NAME(LOCAL) SHOW(ALL | *show_parm*)	VIEWPORT LOCAL	QUERY PORT NAME(LOCAL) SHOW(ALL)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 72. Output fields for the QUERY IMSCON TYPE(PORT) command

Short label	Long label	Keyword	Meaning
ALTCID	AltCID	ALTCID	The alternate client ID that is specified by a client that is issuing a RESUME TPIPE call.
APTK	ApsbToken	CLIENT	The APSB token for ODBM.
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 74 on page 231.

Table 72. Output fields for the QUERY IMSCON TYPE(PORT) command (continued)

Short label	Long label	Keyword	Meaning
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
CLID	ClientID	CLIENT	Name of the client.
CPORT	ClntPort	CLIENT	The port from which the client is sending messages to IMS Connect. Typically, the port is randomly assigned to the client by the TCP/IP stack at the client site.
CSTT	CStatus	CLIENT	Status of the client thread. The status can be one of the following: **CONN** Waiting for output from IMS. **CONV** In a conversational state. **READ** Reading an input message from the client. **RECV** Waiting for input from the client (in other words, in a receive state). **WFCM** Waiting for confirmation (ACK, NAK, or DEALLOCATE) from the client. **XMIT** Sending data to the client.
DS	DataStore	CLIENT	Data store to which the transaction was submitted by the client.
EDTR	EditRtn	EDITRTN	The name of the port input/output edit exit, as defined by the EDIT= parameter of the port definition in the IMS Connect configuration file.
IP	IpAddress	CLIENT	The IP address of the client that is connected to IMS Connect.
ISC	ISCName	CLIENT	The ID of the ISC configuration statement that defines the ISC link that the client is using.
ITO	IdleTO	IDLETO	Timeout value for an idle socket connection in RECV state.
KAV	KeepAv	KEEPAV	The amount of time a connection remains idle before the z/OS TCP/IP layer sends a packet to maintain the connection. The time is specified by the KEEPAV= keyword in either the TCPIP or the ODACCESS statement in the IMS Connect configuration file.
LPLK	LclPlkID	CLIENT	The local MSC physical link ID that is using this connection. The value is specified on the LCLPLKID parameter of the MSC statement in the IMS Connect configuration member. This field is valid for MSC messages only.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
NSOC	NumSoc	NUMSOC	The number of sockets used on each port.

Table 72. Output fields for the QUERY IMSCON TYPE(PORT) command (continued)

Short label	Long label	Keyword	Meaning
PORT	Port	N/A	The port number. The port number is always returned.
			If one of the following characters is appended to the end of the port number, it indicates that the port is dedicated to a particular purpose:
			C Identifies a CICS port.
			D Identifies an ODBM port.
			S Identifies an SSL port.
			If "LOCAL" is displayed instead of a port number, the port is a local port that is used by the IMS TM Resource Adapter.
RTTK	RTToken	RTTOKEN	The RESUME TPIPE token of the client that is issuing a RESUME TPIPE call.
SEC	Second	CLIENT	Number of seconds that the client has been in its current state or status.

STT	Status	STATUS	Status of the port. The status can be one of the following:
			
			ACTIVE
			    The port is active.
			
			NOTACTIVE
			    The port is not active.

TCL	TotClnts	SUMMARY	Total number of clients that are active on the port.
TCON	TotConn	SUMMARY	Total number of clients that have a status of "CONN" on the port.
TIMO	TimeOut	TIMEOUT	The amount of time that IMS Connect waits before terminating a client connection if no messages are received from the client.
TOTH	TotOther	SUMMARY	Total number of clients that have a status other than "CONN", "RECV" or "XMIT" on the port.
TRAN	Trancode	CLIENT	Transaction code submitted by the client.
TRCV	TotRecv	SUMMARY	Total number of clients that have a status of "RECV" on the port.
TREAD	TotRead	SUMMARY	Total number of clients that have a status of "READ" on the port.
TXMT	TotXmit	SUMMARY	Total number of clients that have a status of "XMIT" on the port.
UID	UserID	CLIENT	User ID for the client that is passed to IMS Connect.

Return and reason codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(PORT) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 73. Return and reason codes for the QUERY IMSCON TYPE(PORT) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(PORT) command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See the "Completion codes" table for details.
X'0C000008'	X'00002004'	An invalid keyword or keyword parameter was specified.
X'0C000008'	X'00002014'	An invalid character was specified in the NAME() parameter.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Completion codes

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 74. Completion codes for the QUERY IMSCON TYPE(PORT) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(PORT) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for QUERY IMSCON TYPE(PORT) command

In the following example, port 9999 has four active clients. In the output, the first line shows information and status related to the port. The final four lines of the output display information and status specific to each of the four clients active on the port.

TSO SPOC input:

```
QUERY IMSCON TYPE(PORT) NAME(9999) SHOW(ALL)
```

TSO SPOC output:

```
(Screen 1)
 Port MbrName CC   KeepAv NumSoc EditRtn     TimeOut Status
 9999 HWS1    0       0     5                    0 ACTIVE
 9999 HWS1    0
 9999 HWS1    0
 9999 HWS1    0
 9999 HWS1    0
(Screen 2)
 Port MbrName TotClnts TotRecv TotRead TotConn TotXmit TotOther
 9999 HWS1           4       2       0       2       0        0
 9999 HWS1
 9999 HWS1
 9999 HWS1
 9999 HWS1
(Screen 3)
 Port MbrName ClientID UserID  LclPlkID Trancode DataStore CStatus Second
 9999 HWS1
 9999 HWS1    CLIENT01 USRT003          ITOC04   IMS1      RECV      2468
 9999 HWS1    CLIENT12 USRT002          ITOC04   IMS1      RECV        15
 9999 HWS1    MSC33333          MSC12                      CONN        14
 9999 HWS1    MSC44444          MSC12                      CONN         9

(Screen 4)
 Port MbrName ClntPort IpAddress               ApsbToken
 9999 HWS1
 9999 HWS1        2363 0:0:0:0:0:FFFF:930:6E53
 9999 HWS1        2323 0:0:0:0:0:FFFF:930:6E53
 9999 HWS1        1739 0:0:0:0:0:FFFF:A64:C802
 9999 HWS1        2684 0:0:0:0:0:FFFF:A64:C802
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(PORT) NAME(9999) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 00:19:55.043283</statime>
<stotime>2010.298 00:19:55.044732</stotime>
<staseq>C6C763C5AA9D3667</staseq>
<stoseq>C6C763C5AAF7C5E7</stoseq>
<rqsttkn1>USRT001 10171955</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(PORT) NAME(9999) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left"skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
```

```
                  scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="KAV" llbl="KeepAv" scope="LCL" sort="n" key="0" scroll="yes"
len="7" dtype="INT" align="right" skipb="yes" />
<hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="EDTR" llbl="EditRtn" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TIMO" llbl="TimeOut" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TCL" llbl="TotClnts" scope="LCL" sort="d" key="3"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TRCV" llbl="TotRecv" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TREAD" llbl="TotRead" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TCON" llbl="TotConn" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TXMT" llbl="TotXmit" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CSTT" llbl="CStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="39" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) CLID(CLIENT12)
 UID(USRT002 ) TRAN(ITOC04  ) DS(IMS1    ) CSTT(RECV     ) SEC(15)
 CPORT(2323) IP(0:0:0:0:0:FFFF:930:6E53) APTK(              ) </rsp>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) CLID(CLIENT01)
 UID(USRT003 ) TRAN(ITOC04  ) DS(IMS1    ) CSTT(RECV     ) SEC(2468)
 CPORT(2363) IP(0:0:0:0:0:FFFF:930:6E53) APTK(              ) </rsp>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) CLID(MSC33333)
 LPLK(MSC12   ) CSTT(CONN      ) SEC(14) CPORT(1739)
 IP(0:0:0:0:0:FFFF:A64:C802) APTK(              ) </rsp>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) CLID(MSC44444)
 LPLK(MSC12   ) CSTT(CONN      ) SEC(9) CPORT(2694)
 IP(0:0:0:0:0:FFFF:A64:C802) APTK(              ) </rsp>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) KAV(0) NSOC(5)
 EDTR(        ) TIMO(0) STT(ACTIVE) TCL(4) TRCV(2) TREAD(0) TCON(2)
 TXMT(0) TOTH(0) </rsp>
</cmdrspdata>
</imsout>
```

Example 2 for QUERY IMSCON TYPE(PORT) command

In the following example, port 9999 has four active clients, but because only summary information is requested, only one line of output is displayed.

TSO SPOC input:

```
QUERY IMSCON TYPE(PORT) NAME(9999) SHOW(SUMMARY)
```

TSO SPOC output:

```
Port MbrName  CC  TotClnts TotRecv TotRead TotConn TotXmit TotOther
9999 HWS1      0         4       4       0       0       0        0
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(PORT) NAME(9999) SHOW(SUMMARY) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 00:30:21.422815</statime>
<stotime>2010.298 00:30:21.424144</stotime>
<staseq>C6C7661B074DF4F8</staseq>
<stoseq>C6C7661B07A10C38</stoseq>
<rqsttkn1>USRT001 10173021</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(PORT) NAME(9999) SHOW(SUMMARY) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left"skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="KAV" llbl="KeepAv" scope="LCL" sort="n" key="0" scroll="yes"
len="7" dtype="INT" align="right" skipb="yes" />
<hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="EDTR" llbl="EditRtn" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TIMO" llbl="TimeOut" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TCL" llbl="TotClnts" scope="LCL" sort="d" key="3"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TRCV" llbl="TotRecv" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TREAD" llbl="TotRead" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TCON" llbl="TotConn" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TXMT" llbl="TotXmit" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
```

```
                    scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
            <hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="4"
             scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
             len="8" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
             scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
             scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
             scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="CSTT" llbl="CStatus" scope="LCL" sort="n" key="0"
             scroll="yes" len="9" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
             len="10" dtype="INT" align="right" skipb="yes" />
            <hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
             scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
             scroll="yes" len="39" dtype="CHAR" align="left" skipb="yes" />
            <hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
             scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
         </cmdrsphdr>
         <cmdrspdata>
         <rsp>PORT(9999     ) MBR(HWS1           ) CC(    0) TCL(4) TRCV(4)
          TREAD(0) TCON(0) TXMT(0) TOTH(0) </rsp>
         </cmdrspdata>
         </imsout>
```

Example 3 for QUERY IMSCON TYPE(PORT) command

In the following example, the command requests any port that has a client active
with a client ID that begins with CLIENT1. Port 9999 has one such client active:
CLIENT12. Other clients might be active on the port, but they are not displayed
because their client IDs do not match the CLIENT keyword filter. Likewise, other
ports might be in use, but because no clients match the CLIENT keyword filter, the
port information is not displayed for those ports.

Although the SHOW keyword was not specified, SHOW(CLIENT) is assumed
because the CLIENT keyword was specified.

In the output, the first line displays information and status related to the port. In
this case, the first line only displays that it exists because no other SHOW keyword
parameter was specified. The remaining line of output displays information and
status for CLIENT12.

TSO SPOC input:

```
QUERY IMSCON TYPE(PORT) CLIENT(CLIENT1*)
```

TSO SPOC output:

```
(Screen 1)
 Port MbrName CC ClientID UserID  Trancode DataStore Second CStatus
 9999 HWS1     0
 9999 HWS1     0 CLIENT12 USRT002 FESTX2   IMS1          15 RECV
(Screen 2)
 Port MbrName ClntPort IpAddress              ApsbToken
 9999 HWS1
 9999 HWS1        2323 0:0:0:0:0:FFFF:930:6E53
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(PORT) CLIENT(CLIENT1*) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 00:37:11.954827</statime>
<stotime>2010.298 00:37:11.955822</stotime>
<staseq>C6C767A28AD8BD37</staseq>
<stoseq>C6C767A28B16EB37</stoseq>
<rqsttkn1>USRT001 10173711</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1     </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(PORT) CLIENT(CLIENT1*) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left"skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="KAV" llbl="KeepAv" scope="LCL" sort="n" key="0" scroll="yes"
len="7" dtype="INT" align="right" skipb="yes" />
<hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="EDTR" llbl="EditRtn" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TIMO" llbl="TimeOut" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TCL" llbl="TotClnts" scope="LCL" sort="d" key="3"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TRCV" llbl="TotRecv" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TREAD" llbl="TotRead" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TCON" llbl="TotConn" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TXMT" llbl="TotXmit" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CSTT" llbl="CStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
```

```
 scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="39" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) CLID(CLIENT12)
 UID(USRT002 ) TRAN(FESTX2  ) DS(IMS1    ) CSTT(RECV     ) SEC(15)
 CPORT(2323) IP(0:0:0:0:0:FFFF:930:6E53) APTK(                ) </rsp>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Example 4 for QUERY IMSCON TYPE(PORT) command

In the following example, the command requests any port that has a client active
with a user ID of USRT002. Port 9999 has one such client active: CLIENT12. Other
clients might be active on the port, but they are not displayed because their user
IDs do not match the USERID keyword filter. Likewise, other ports might be in
use, but because no clients match the USERID keyword filter, the port information
is not displayed for those ports.

In the output, the first line of output displays information and status related to the
port. In this case, the first line only displays that it exists because no other SHOW
keyword parameter was specified. The remaining line of output displays
information and status for CLIENT12.

Although the SHOW keyword was not specified, SHOW(CLIENT) is assumed
because the USERID keyword filter was specified.

TSO SPOC input:
```
QUERY IMSCON TYPE(PORT) USERID(USRT002)
```

TSO SPOC output:
```
(Screen 1)
 Port MbrName CC ClientID UserID  Trancode DataStore Second CStatus
 9999 HWS1     0
 9999 HWS1     0 CLIENT12 USRT002 FESTX2   IMS1          15 RECV
(Screen 2)
 Port MbrName ClntPort IpAddress              ApsbToken
 9999 HWS1
 9999 HWS1        2323 0:0:0:0:0:FFFF:930:6E53
```

OM API input:
```
CMD ( QUERY IMSCON TYPE(PORT) USERID(USRT002) )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 00:42:47.545738</statime>
<stotime>2010.298 00:42:47.567715</stotime>
<staseq>C6C768E29638ACEA</staseq>
<stoseq>C6C768E29B96379A</stoseq>
<rqsttkn1>USRT001 10174247</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
```

```
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(PORT) USERID(USRT002) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left"skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="KAV" llbl="KeepAv" scope="LCL" sort="n" key="0" scroll="yes"
len="7" dtype="INT" align="right" skipb="yes" />
<hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="EDTR" llbl="EditRtn" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TIMO" llbl="TimeOut" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TCL" llbl="TotClnts" scope="LCL" sort="d" key="3"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TRCV" llbl="TotRecv" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TREAD" llbl="TotRead" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TCON" llbl="TotConn" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TXMT" llbl="TotXmit" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CSTT" llbl="CStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="39" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(    0) CLID(CLIENT12)
 UID(USRT002 ) TRAN(FESTX2  ) DS(IMS1    ) CSTT(RECV    ) SEC(15)
 CPORT(2323) IP(0:0:0:0:0:FFFF:930:6E53) APTK(               ) </rsp>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(    0) </rsp>
</cmdrspdata>
</imsout>
```

Example 5 for QUERY IMSCON TYPE(PORT) command

In the following example, the command requests information about port 9876. Because the port is defined by the CICSPORT parameter of the TCPIP and ISC statements, a C is appended to the port number.

Two client connections are active on port 9876. The client IDs start with the characters ISC, which also indicate that these clients are using ISC connections. Both clients are using an ISC connection that is defined to IMS Connect by the ISC configuration statement, IMS1CIC4.

TSO SPOC input:
```
QUERY IMSCON TYPE(PORT) NAME(9876) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
Port      MbrName            CC  KeepAv     NumSoc EditRtn    TimeOut Status
9876C     HWS1                0       0          3                 0 ACTIVE
9876C     HWS1                0
9876C     HWS1                0
(screen 2)
Port      MbrName           TotClnts TotRecv TotRead TotConn TotXmit TotOther
9876C     HWS1                    2       2       0       0       0        0
9876C     HWS1
9876C     HWS1
(screen 3)
Port      MbrName           ClientID ISCName  CStatus      Second ClntPort
9876C     HWS1
9876C     HWS1              ISCC279A IMS1CIC4 RECV           1384 1057
9876C     HWS1              ISC41218 IMS1CIC4 RECV           1388 1055
(screen 4)
Port      MbrName           ClientID IpAddress
9876C     HWS1
9876C     HWS1              ISCC279A 0:0:0:0:0:FFFF:91E:72F3
9876C     HWS1              ISC41218 0:0:0:0:0:FFFF:91E:72F3
```

OM API input:
```
CMD ( QUERY IMSCON TYPE(PORT) NAME(9876) SHOW(ALL) )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2012.221 18:31:09.431501</statime>
<stotime>2012.221 18:31:09.433764</stotime>
<staseq>C9FD5994E6ACD414</staseq>
<stoseq>C9FD5994E73A4194</stoseq>
<rqsttkn1>USRT001 10113109</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON         </kwd>
<input>QUERY IMSCON TYPE(PORT) NAME(9876) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
```

```
                            len="8" dtype="CHAR" align="left" skipb="no" />
                   <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
                            len="8" dtype="CHAR" align="left" skipb="no" />
                   <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
                            len="4" dtype="INT" align="right" skipb="no" />
                   <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
                            scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="KAV" llbl="KeepAv" scope="LCL" sort="n" key="0" scroll="yes"
                            len="7" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
                            scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="EDTR" llbl="EditRtn" scope="LCL" sort="n" key="0"
                            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="TIMO" llbl="TimeOut" scope="LCL" sort="n" key="0"
                            scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
                            len="9" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="TCL" llbl="TotClnts" scope="LCL" sort="d" key="3"
                            scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="TRCV" llbl="TotRecv" scope="LCL" sort="n" key="0"
                            scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="TREAD" llbl="TotRead" scope="LCL" sort="n" key="0"
                            scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="TCON" llbl="TotConn" scope="LCL" sort="n" key="0"
                            scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="TXMT" llbl="TotXmit" scope="LCL" sort="n" key="0"
                            scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
                            scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="4"
                            scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
                            len="8" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
                            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="n" key="0"
                            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
                            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
                            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="CSTT" llbl="CStatus" scope="LCL" sort="n" key="0"
                            scroll="yes" len="9" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
                            len="10" dtype="INT" align="right" skipb="yes" />
                   <hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
                            scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
                            scroll="yes" len="39" dtype="CHAR" align="left" skipb="yes" />
                   <hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
                            scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
                   </cmdrsphdr>
                   <cmdrspdata>
                   <rsp>PORT(9876C  ) MBR(HWS1            ) CC(   0) CLID(ISCC279A)
                    ISC(IMS1CIC4) CSTT(RECV     ) SEC(1384) CPORT(1057)
                    IP(0:0:0:0:0:FFFF:91E:72F3) </rsp>
                   <rsp>PORT(9876C  ) MBR(HWS1            ) CC(   0) CLID(ISC41218)
                    ISC(IMS1CIC4) CSTT(RECV     ) SEC(1388) CPORT(1055)
                    IP(0:0:0:0:0:FFFF:91E:72F3) </rsp>
                   <rsp>PORT(9876C  ) MBR(HWS1            ) CC(   0) KAV(0) NSOC(3) EDTR(
                           ) TIMO(0) STT(ACTIVE) TCL(2) TRCV(2) TREAD(0) TCON(0) TXMT(0)
                    TOTH(0) </rsp>
                   </cmdrspdata>
                   </imsout>
```

Example 6 for QUERY IMSCON TYPE(PORT) command

In the following example, the command retrieves information about all the clients connected to port 9999. The command output shows that client CLIENT01 has issued a resume TPIPE because we can see the resume TPIPE token. Further more, CLIENT01 is retrieving output messages on behalf of the alternate client ID CLIENT02.

TSO SPOC input:

```
QUERY IMSCON TYPE(PORT) NAME(9999) SHOW(CLIENT)
```

TSO SPOC output:

```
(screen 1)
Port    MbrName          CC ClientID UserID   Trancode DataStore CStatus
9999    HWS1              0
9999    HWS1              0 CLIENT01 USRT003            IMS1      RECV WFCM
(screen 2)
Port    MbrName          ClientID    Second ClntPort IpAddress
9999    HWS1
9999    HWS1             CLIENT01     225 57063     0:0:0:0:0:FFFF:937:86B2
(screen 3)
Port    MbrName          ClientID AltCID   RTToken
9999    HWS1
9999    HWS1             CLIENT01 CLIENT02 CDDDA95551AFA509
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(PORT) NAME(9999) SHOW(CLIENT) )
```

OM API output:

```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2014.280 00:39:07.269359</statime>
<stotime>2014.280 00:39:07.269647</stotime>
<staseq>CDDDAB60A5CEF204</staseq>
<stoseq>CDDDAB60A5E0FC84</stoseq>
<rqsttkn1>USRT002 10173907</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(PORT) NAME(9999) SHOW(CLIENT) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="KAV" llbl="KeepAv" scope="LCL" sort="n" key="0" scroll="yes"
 len="7" dtype="INT" align="right" skipb="yes" />
<hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="EDTR" llbl="EditRtn" scope="LCL" sort="n" key="0"
```

```
                    scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="TIMO" llbl="TimeOut" scope="LCL" sort="n" key="0"
                     scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
                    <hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
                     len="9" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="TCL" llbl="TotClnts" scope="LCL" sort="d" key="3"
                     scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                    <hdr slbl="TRCV" llbl="TotRecv" scope="LCL" sort="n" key="0"
                     scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                    <hdr slbl="TREAD" llbl="TotRead" scope="LCL" sort="n" key="0"
                     scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                    <hdr slbl="TCON" llbl="TotConn" scope="LCL" sort="n" key="0"
                     scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                    <hdr slbl="TXMT" llbl="TotXmit" scope="LCL" sort="n" key="0"
                     scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                    <hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
                     scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
                    <hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="4"
                     scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
                     len="8" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="n" key="0"
                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="CSTT" llbl="CStatus" scope="LCL" sort="n" key="0"
                     scroll="yes" len="9" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
                     len="10" dtype="INT" align="right" skipb="yes" />
                    <hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
                     scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
                     scroll="yes" len="39" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
                     scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="ALTCID" llbl="AltCID" scope="LCL" sort="n" key="0"
                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                    <hdr slbl="RTTK" llbl="RTToken" scope="LCL" sort="n" key="0"
                     scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
                    </cmdrsphdr>
                    <cmdrspdata>
                    <rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) CLID(CLIENT01)
                     UID(USRT003 ) TRAN(        ) DS(IMS1    ) CSTT(RECV WFCM) SEC(225)
                     CPORT(57063) IP(0:0:0:0:0:FFFF:937:86B2) ALTCID(CLIENT02)
                     RTTK(CDDDA95551AFA509) </rsp>
                    <rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) </rsp>
                    </cmdrspdata>
                    </imsout>
```

Example 7 for QUERY IMSCON TYPE(PORT) command

The following example shows how the IDLETO value is displayed. In the example, the command requests all information about the clients that are connected to five different ports.

The idle timeout value is not applicable for CICS and DRDA ports. Therefore, the output lines for ports 5555 and 6600 show blanks for IDLETO value.

The IDLETO value of -1 means an idle timeout was not defined for the port. Socket connections on port 9999 use the global-level IDLETO value.

The IDLETO value of 1000 means the idle timeout for socket connections on port 7777 is enabled with a timeout value of 10 seconds.

The IDLETO value of 0 means the idle timeout value for socket connections on port 9998 is disabled.

TSO SPOC input:
```
QRY IMSCON TYPE(port) name(5555,6600,7777,9998,9999) show(all)
```

TSO SPOC output:
```
(screen 1)
Port       MbrName            CC   KeepAv     NumSoc EditRtn      TimeOut
5555D      HWS1                0        0          1              12000
6600C      HWS1                0        0          1                  0
7777       HWS1                0        0          1                  0
9998       HWS1                0        0          1                  0
9999       HWS1                0        0          1                  0

(screen 2)
Port       MbrName            Status        IdleTO
5555D      HWS1               ACTIVE
6600C      HWS1               ACTIVE
7777       HWS1               ACTIVE          1000
9998       HWS1               ACTIVE             0
9999       HWS1               ACTIVE            -1
```

OM API input:
```
QRY IMSCON TYPE(PORT) NAME(5555,6600,7777,9998,9999) SHOW(ALL)
```

OM API output:

```
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="KAV" llbl="KeepAv" scope="LCL" sort="n" key="0" scroll="yes"
 len="7" dtype="INT" align="right" skipb="yes" />
<hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="EDTR" llbl="EditRtn" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TIMO" llbl="TimeOut" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TCL" llbl="TotClnts" scope="LCL" sort="d" key="3"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TRCV" llbl="TotRecv" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TREAD" llbl="TotRead" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TCON" llbl="TotConn" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TXMT" llbl="TotXmit" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="4"
```

```
|            scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
|            len="8" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
|            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="n" key="0"
|            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="n" key="0"
|            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="DS" llbl="DataStore" scope="LCL" sort="n" key="0"
|            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="CSTT" llbl="CStatus" scope="LCL" sort="n" key="0"
|            scroll="yes" len="9" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
|            len="10" dtype="INT" align="right" skipb="yes" />
|          <hdr slbl="CPORT" llbl="ClntPort" scope="LCL" sort="n" key="0"
|            scroll="yes" len="5" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
|            scroll="yes" len="39" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="APTK" llbl="ApsbToken" scope="LCL" sort="n" key="0"
|            scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="ALTCID" llbl="AltCID" scope="LCL" sort="n" key="0"
|            scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="RTTK" llbl="RTToken" scope="LCL" sort="n" key="0"
|            scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
|          <hdr slbl="ITO" llbl="IdleTO" scope="LCL" sort="n" key="0" scroll="yes"
|            len="10" dtype="INT" align="right" skipb="yes" />
|        </cmdrsphdr>
|        <cmdrspdata>
|        <rsp>PORT(5555D  ) MBR(HWS1            ) CC(   0) KAV(0) NSOC(1) EDTR(
|              ) TIMO(12000) STT(ACTIVE) </rsp>
|        <rsp>PORT(6600C  ) MBR(HWS1            ) CC(   0) KAV(0) NSOC(1) EDTR(
|              ) TIMO(0) STT(ACTIVE) </rsp>
|        <rsp>PORT(7777   ) MBR(HWS1            ) CC(   0) ITO(1000) KAV(0)
|         NSOC(1) EDTR(        ) TIMO(0) STT(ACTIVE) </rsp>
|        <rsp>PORT(9998   ) MBR(HWS1            ) CC(   0) ITO(0) KAV(0)
|         NSOC(1) EDTR(        ) TIMO(0) STT(ACTIVE) </rsp>
|        <rsp>PORT(9999   ) MBR(HWS1            ) CC(   0) ITO(-1) KAV(0)
|         NSOC(1) EDTR(        ) TIMO(0) STT(ACTIVE) </rsp>
|        </cmdrspdata>
```

Related concepts:

➦ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➥ VIEWPORT command (Commands)

➥ IMS Connect QUERY PORT command (Commands)

QUERY IMSCON TYPE(RMTCICS) command

Use the QUERY IMSCON TYPE(RMTCICS) command to display the attributes and status of one or more ISC links to a remote CICS subsystem that is defined to IMS Connect.

Subsections:
- "Environment" on page 245
- "Syntax" on page 245
- "Usage notes" on page 245
- "Equivalent WTOR and z/OS commands" on page 246
- "Keywords" on page 246

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►─┬─QUERY─┬──IMSCON──TYPE(RMTCICS)──┬─NAME(*)───────────────────────────────┬─►
   └─QRY───┘                         │            ┌─,─────────────┐          │
                                     │            ▼               │          │
                                     └─NAME(──┬─RMTCICS_name───┬──)─┘
                                              └─RMTCICS_name*──┘

►─┬───────────────────────┬──┬────────────────────────────────┬──────────────►
  │       ┌─,──────┐       │  │          ┌─,──────────────────┐│
  │       ▼        │       │  │          ▼                    ││
  └─SCSTATUS(──┬─CONN─┬──)─┘  └─SENDCLNT(──┬─sendclient_name──┬──)─┘
              └─RECV─┘                    └─sendclient_name*─┘

►─┬──────────────────────────┬──┬───────────────────────────┬──►◄
  │         ┌─,────────┐     │  │         ┌─,─────────┐      │
  │         ▼          │     │  │         ▼           │      │
  └─SHOW(──┬─ALL──────┬──)─┘  └─STATUS(──┬─ACTIVE────┬──)─┘
          ├─HOSTNAME─┤                  ├─NOTACTIVE─┤
          ├─IPADDR───┤                  └─STOPPED───┘
          ├─NUMSOC───┤
          ├─PORT─────┤
          ├─SENDCLNT─┤
          ├─STATUS───┤
          └─SUMMARY──┘
```

Usage notes

You can issue the QUERY IMSCON TYPE(RMTCICS) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Typically, this command results in one output display line for each remote CICS definition being displayed. However, if the SHOW(SENDCLNT) keyword is specified, an additional line is displayed for each send client associated with the specified remote CICS definition.

Equivalent WTOR and z/OS commands

The QUERY IMSCON TYPE(RMTCICS) command does not have an equivalent WTOR or z/OS MODIFY command. However, the WTOR VIEWHWS command and the z/OS MODIFY QUERY MEMBER TYPE(IMSCON) command also display some of the same attribute and status fields as the QUERY IMSCON TYPE(RMTCICS).

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 75. WTOR and IMS Connect z/OS equivalents for the QUERY IMSCON TYPE(RMTCICS) command

QUERY IMSCON TYPE(RMTCICS) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
QUERY IMSCON TYPE(RMTCICS) NAME(*) SHOW(ALL | *show_parm*)	None	None
QUERY IMSCON TYPE(RMTCICS) NAME(*RMTCICS_name*) SHOW(ALL | *show_parm*)	None	None

Keywords

The following keywords are valid for the QUERY IMSCON TYPE(RMTCICS) command.

NAME
> Specify the ID values from one or more RMTCICS configuration statements to display the status and attributes of the corresponding remote CICS connections. You can specify a single RMTCICS ID or a list of RMTCICS IDs separated by commas. You can use wildcards.
>
> You can specify NAME(*) to display all RMTCICS resources. NAME(*) is the default.

SCSTATUS
> A filter that selects remote CICS resources for display that have at least one send client that is in one of the specified states. You can specify a single send client status, or a list of send client statuses separated by commas.
>
> The filters supported with the SCSTATUS keyword, which can be specified in any order, are:
>
> **CONN**
> > This connection is in connect state. It can send messages to the remote CICS.
>
> **RECV**
> > This connection is in receive state. It is waiting to receive a response from the remote CICS.
>
> Only those remote CICS resources that have a send client that is in one of the specified states are displayed. Remote CICS resources that match the NAME() parameter, but do not have such a send client, are not displayed.

When the SCSTATUS filter is specified, send client information is displayed even if SHOW(SENDCLNT) is not specified.

SENDCLNT
Selects remote CICS resources for display that have one of the specified send clients defined. You can specify a single send client name or a list of send client names separated by commas. Wildcards can be used in the names.

Only those remote CICS resources that have a specified send client are displayed. Remote CICS resources that match the NAME() parameter, but do not have the specified send client, are not displayed.

When the SENDCLNT filter is specified, send client information is displayed even if SHOW(SENDCLNT) is not specified. Only those clients that match the names specified by the SENDCLNT filter are displayed. Send clients that are associated with the remote CICS, but which do not match the filter, are not displayed.

SHOW
Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the remote CICS name, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL
Displays all output fields.

HOSTNAME
Displays the host name of the remote CICS subsystem. The host name of a remote CICS is defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.

IPADDR
Displays the IP address of the remote CICS subsystem.

NUMSOC
Displays the current number of sockets that are being used for this remote CICS connection.

PORT
Displays the port on which the remote CICS subsystem receives transactions and data replies from IMS Connect on this ISC link. This port is defined in CICS on the PORT keyword of the TCPIPSERVICE resource definition. This port is specified in IMS Connect on the PORT keyword of the RMTCICS statement that defines the connection to the remote CICS subsystem.

SENDCLNT
Displays information and status for the send sockets of the active parallel sessions (ISCUSERs) that are using the connection to the remote CICS subsystem. If one or more parallel sessions are active with a remote CICS, a separate output line is displayed for each parallel session. If there are no active parallel sessions, no additional output lines are displayed.

Information displayed for the send socket includes:
- The send client ID that IMS Connect randomly generates and assigns to the send socket.

- Status or state of the send socket connection, which is one of the following types:

  CONN
  > The send socket connection is in connect state. It can send messages to the remote CICS.

  RECV The send socket connection is in receive state. It is waiting to receive a response from the remote CICS.
- Number of seconds that this socket connection is in the specified state.
- The local port to which the send socket belongs.

STATUS
> Displays the state of the remote CICS connection. For a description of the status returned, see the STATUS keyword in Table 76 on page 249.

SUMMARY
> Displays summary information related to the send clients that are active on the remote CICS connection. To see specific send client information, specify the SENDCLNT parameter in addition to the SUMMARY parameter.
>
> Information displayed includes:
> - Total number of send clients that are active on the remote CICS connection
> - Total number of send clients in RECV state
> - Total number of send clients in CONN state
> - Total number of send clients in XMIT state
> - Total number of send clients that are not in any of these states

STATUS
> Selects remote CICS resources for display that possess at least one of the specified statuses. When the STATUS filter is specified, status information is displayed even if SHOW(STATUS) is not specified.
>
> The filters supported with the STATUS keyword, which can be specified in any order, are:
>
> **ACTIVE**
> > Selects remote CICS resources that have a status of ACTIVE, meaning that the connection to the remote CICS subsystem is active.
>
> **NOTACTIVE**
> > Selects remote CICS resources that have a status of NOTACTIVE, meaning that the connection to the remote CICS is not active.
>
> **STOPPED**
> > Selects remote CICS resources for which a command stopped communications between IMS Connect and the remote CICS subsystem that is identified on the NAME keyword. Any messages to be sent to the CICS subsystem that is specified on the NAME keyword are rejected and an error message is sent back to the requester.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 76. Output fields for the QUERY IMSCON TYPE(RMTCICS) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 78 on page 251.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
HOST	HostName	HOSTNAME	The host name of the remote CICS subsystem. Remote CICS subsystems are defined in the RMTCICS parameter of the ISC statement in the IMS Connect configuration member.
IP	IpAddress	IPADDR	The IP address of the remote CICS. The remote CICS is defined in the RMTCICS of the ISC statement in the IMS Connect configuration member.
ISC	ISCName	SENDCLNT	The ID of the ISC configuration statement in the IMS Connect configuration member.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
NSOC	NumSoc	NUMSOC	The current number of sockets that are being used for this CICS connection.
PORT	Port	PORT	The port of the associated remote CICS subsystem. The remote CICS is defined on the RMTCICS parameter of the ISC statement.
RCICS	RmtCICS	N/A	Remote CICS name. The CICS name is always returned.
SCL	SendClnt	SENDCLNT	The client name that IMS Connect assigns to the socket that is used to send the transactions and data replies to a remote CICS subsystem (RMTCICS) for the ISC parallel session that is identified in the corresponding IUSER output field.
SCSTT	SendStatus	SENDCLNT	Status of the send client connection. The status can be one of the following types: **CONN** This connection is in connect state. It can send messages to the remote CICS. **RECV** This connection is in receive state. It is waiting to receive a response from the remote CICS.
SEC	Second	SENDCLNT	Number of seconds that this connection is in the specified state.

Table 76. Output fields for the QUERY IMSCON TYPE(RMTCICS) command (continued)

Short label	Long label	Keyword	Meaning
SPORT	SendPort	SENDCLNT	The port that this instance of IMS Connect used to connect to the remote CICS.
STT	Status	STATUS	Status of the remote CICS connection. The status can be one of the following types: **ACTIVE** The connection to the remote CICS is active. A connection with this status has at least one socket connection to the CICS identified in the RMTCICS field. **NOTACTIVE** The connection to the remote CICS is not active. A connection with this status does not have any socket connections to the CICS that is identified in the RMTCICS field. **STOPPED** A command stopped communications between this IMS Connect and the CICS that is identified in the RMTCICS field. Any messages that are sent to the CICS that is shown in RMTCICS field are rejected and an error message is sent back to the sender.
TSCL	TotSClnts	SUMMARY	Total number of send clients that are active on the remote CICS connection.
TCON	TotConn	SUMMARY	Total number of send clients that have a status of CONN on the remote CICS connection.
TOTH	TotOther	SUMMARY	Total number of send clients that have a status other than CONN, RECV, or XMIT on the remote CICS connection.
TRCV	TotRecv	SUMMARY	Total number of send clients that have a status of RECV on the remote CICS connection.
TXMT	TotXmit	SUMMARY	Total number of send clients that have a status of XMIT on the remote CICS connection.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(RMTCICS) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 77. Return and reason codes for the QUERY IMSCON TYPE(RMTCICS) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(RMTCICS) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.

Table 77. Return and reason codes for the QUERY IMSCON TYPE(RMTCICS) command (continued)

Return code	Reason code	Meaning
X'0C000004'	X'00001010'	No RMTCICS resource was found that matched the NAME or the filters specified, if any.
X'0C000008'	X'00002004'	Invalid keyword or keyword combination for the command.
X'0C000008'	X'00002014'	Invalid character specified in the resource name.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005004'	The command processor failed to obtain storage for the output buffer.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 78. Completion codes for the QUERY IMSCON TYPE(RMTCICS) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(RMTCICS) command was completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

In the following example, the output shows the current status of the RMTCICS resource "ICONTT1". The status of ICONTT1 is ACTIVE, so it can be used for ISC communication between IMS and CICS. Two send clients are shown that are associated with ICONTT1: "ISC11111" and "ISC22222". Both send clients were created for the ISC resource "LU6TCP1" and are both in CONN state.

TSO SPOC input:

```
QUERY IMSCON TYPE(RMTCICS) NAME(ICONNT1) SHOW(ALL)
```

TSO SPOC output:

```
(Screen 1)
 RmtCICS   MbrName CC  IpAddress       HostName
 ICONNT1   HWS1    0   127.0.0.1       WWW.EXAMPLE.COM
 ICONNT1   HWS1    0
 ICONNT1   HWS1    0
```

```
(Screen 2)
RmtCICS    MbrName Port NumSoc Status
ICONNT1    HWS1    9999 2      ACTIVE
ICONNT1    HWS1
ICONNT1    HWS1

(Screen 3)
RmtCICS    MbrName TotSClnts TotRecv TotConn TotXmit TotOther
ICONNT1    HWS1    2         0       2       0       0
ICONNT1    HWS1
ICONNT1    HWS1

(Screen 4)
RmtCICS    MbrName SendClnt ISCName   Second SendPort
ICONNT1    HWS1
ICONNT1    HWS1    ISC11111 LU6TCP1      100     1234
ICONNT1    HWS1    ISC22222 LU6TCP1       89     5678

 (Screen 5)
RmtCICS    MbrName SendStatus
ICONNT1    HWS1
ICONNT1    HWS1    CONN
ICONNT1    HWS1    CONN
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(RMTCICS) NAME(ICONNT1) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2012.097 22:05:09.441459</statime>
<stotime>2012.097 22:05:09.443581</stotime>
<staseq>C961A224EEBB3C9E</staseq>
<stoseq>C961A224EF3FD01E</stoseq>
<rqsttkn1>USRT001 10150509</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1     </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON            </kwd>
<input>QUERY IMSCON TYPE(RMTCICS) NAME(CICS2) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="RCICS" llbl="RmtCICS" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
 scroll="yes" len="15" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="HOST" llbl="HostName" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
```

```
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TSCL" llbl="TotSClnts" scope="LCL" sort="d" key="4"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TRCV" llbl="TotRecv" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TCON" llbl="TotConn" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TXMT" llbl="TotXmit" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="a" key="3"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SPORT" llbl="SendPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SCSTT" llbl="SendStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RCICS(ICONTT1 ) MBR(HWS1            ) CC(   0) SCL(ISC11111)
 ISC(LU6TCP1 ) SEC(100) SPORT(1234) SCSTT(CONN) </rsp>
<rsp>RCICS(ICONTT1 ) MBR(HWS1            ) CC(   0) SCL(ISC22222)
 ISC(LU6TCP1 ) SEC(89) SPORT(5678) SCSTT(CONN) </rsp>
<rsp>RCICS(ICONTT1 ) MBR(HWS1            ) CC(   0) IP(127.0.0.1)
 HOST(WWW.EXAMPLE.COM) PORT(9999) NSOC(2)  ) STT(ACTIVE   ) TSCL(2)
 TRCV(0) TCON(2) TXMT(0) TOTH(0)
 </rsp>
</cmdrspdata>
</imsout>
```

Related reference:

QUERY IMSCON TYPE(RMTIMSCON) command

Use the QUERY IMSCON TYPE(RMTIMSCON) command to display the status of one or more remote IMS Connects that are defined to IMS Connect.

Subsections:

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
                                                        ┌─NAME(*)──────────────────────┐
►►─┬─QUERY─┬──IMSCON──TYPE(RMTIMSCON)──┤                                               ├──►
   └─QRY───┘                           │           ┌─,──────────────────┐             │
                                       └─NAME(─▼─┬─rmtimscon_name──┬───)─┘
                                                 └─rmtimscon_name*─┘

          ┌────────────────────┐      ┌──────────────────────────────┐
►──┬──────────────────────┬────┬──────────────────────────────────────┬──►
   │         ┌─,───┐       │    │              ┌─,────────────────┐     │
   └─SCSTATUS(─▼─┬─CONN─┬──)─┘    └─SENDCLNT(─▼─┬─sendclient_name──┬──)─┘
               └─RECV─┘                        └─sendclient_name*─┘

►──┬──────────────────────────┬──┬────────────────────────┬──►◄
   │         ┌─,──────────┐    │  │         ┌─,──────────┐  │
   └─SHOW(─▼─┬─ALL────────┬─)──┘  └─STATUS(─▼─┬─ACTIVE────┬─)─┘
            ├─APPL───────┤                    ├─NOTACTIVE─┤
            ├─AUTOCONN───┤                    ├─RETRYCONN─┤
            ├─HOSTNAME───┤                    └─STOPPED───┘
            ├─IDLETO─────┤
            ├─IPADDR─────┤
            ├─NUMSOC─────┤
            ├─PERSISTENT─┤
            ├─PORT───────┤
            ├─RESVSOC────┤
            ├─SENDCLNT───┤
            ├─STATUS─────┤
            ├─SUMMARY────┤
            └─USERID─────┘
```

Keywords

The following keywords are valid for the QUERY IMSCON TYPE(RMTIMSCON) command.

NAME
Specifies one or more remote IMS Connect resources to be displayed. You can specify a single remote IMS Connect name or a list of remote IMS Connect names separated by commas. Wildcards can be used in the names.

You can specify NAME(*) to display all remote IMS Connect resources. NAME(*) is the default.

SCSTATUS
Selects remote IMS Connect resources for display that have at least one send client that is in one of the specified states. You can specify a single send client status, or a list of send client statuses separated by commas.

The filters supported with the SCSTATUS keyword, which can be specified in any order, are:

CONN
   This connection is in connect state. It can send messages to the remote IMS Connect.

RECV
   This connection is in receive state. It is waiting to receive a response from the remote IMS Connect.

Only those remote IMS Connect resources that have a send client that is in one of the specified states are displayed. Remote IMS Connect resources that match the NAME() parameter, but do not have such a send client, are not displayed.

When the SCSTATUS filter is specified, send client information is displayed even if SHOW(SENDCLNT) is not specified. Only those send clients that are in one of the specified states are displayed.

SENDCLNT
   Selects remote IMS Connect resources for display that have one of the specified send clients defined. You can specify a single send client name or a list of send client names separated by commas. Wildcards can be used in the names.

   Only those remote IMS Connect resources that have a specified send client are displayed. Remote IMS Connect resources that match the NAME() parameter, but do not have the specified send client, are not displayed.

   When the SENDCLNT filter is specified, send client information is displayed even if SHOW(SENDCLNT) is not specified. Only those clients that match the names specified by the SENDCLNT filter are displayed. Send clients that are associated with the remote IMS Connect, but which do not match the filter, are not displayed.

SHOW
   Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the remote IMS Connect name, the name of the IMS Connect that processes the command, and the completion code.

   The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL
   Displays all output fields.

APPL
   Displays the APPL name that is used to generate the RACF PassTicket. The PassTicket is then sent to the remote IMS Connect and used to authenticate the user ID.

AUTOCONN
   Displays whether this instance of IMS Connect automatically connects to the remote IMS Connect when this instance of IMS Connect starts.

RESVSOC
   Displays the number of send sockets available for this remote IMS Connect connection. This value is defined in the RESVSOC parameter of the RMTIMSCON statement in the IMS Connect configuration member.

HOSTNAME
   Displays the host name of the remote IMS Connect.

IDLETO

Displays the idle timeout interval to keep this socket connection before closing it due to inactivity.

IPADDR

Displays the IP address of the remote IMS Connect.

NUMSOC

Displays the current number of sockets being used for this remote IMS Connect connection.

PERSISTENT

Displays whether the connections to the remote IMS Connect are persistent.

PORT

Displays the port number of the remote IMS Connect that this IMS Connect instance communicates on.

SENDCLNT

Displays information and status for active connections (send clients) to this remote IMS Connect. If this instance of IMS Connect has one or more active connections to this remote IMS Connect, the command output includes information about each connection. A separate output line is displayed for each connection. This output line is in addition to any output lines that display general remote IMS Connect information. If there are no active connections, no additional output line is displayed.

Information displayed for the send client includes:
- The client ID that this instance of IMS Connect used to connect to the remote IMS Connect.
- The user ID specified by the local IMS to be sent to the remote IMS for transaction authorization. This information is valid for OTMA messages only.
- The local MSC physical link ID that is using this connection.
- Status or state of the remote IMS Connect connection, which is one of the following:

  CONN

  This connection is in connect state. It can send messages to the remote IMS Connect.

  RECV This connection is in receive state. It is waiting to receive a response from the remote IMS Connect.
- Number of seconds that this connection is in the specified state.
- The port that this instance of IMS Connect used to connect to the remote IMS Connect.

STATUS

Displays the state of the remote IMS Connect. For a description of the status returned, see the STATUS keyword in Table 80 on page 258.

SUMMARY

Displays summary information related to the send clients that are active on the remote IMS Connect. To see specific send client information, specify the SENDCLNT parameter in addition to the SUMMARY parameter.

Information displayed includes:
- Total number of send clients that are active on the remote IMS Connect

- Total number of send clients in RECV state
- Total number of send clients in CONN state
- Total number of send clients in XMIT state
- Total number of send clients that are not in any of these states

USERID

Displays the user ID to use for generating the RACF PassTicket, which is sent to the remote IMS Connect and used to authenticate the user ID.

STATUS

Selects remote IMS Connect resources for display that possess at least one of the specified statuses. When the STATUS filter is specified, status information is displayed even if SHOW(STATUS) is not specified.

The filters supported with the STATUS keyword, which can be specified in any order, are:

ACTIVE

Selects remote IMS Connect resources that have a status of ACTIVE, meaning that the connection to the remote IMS Connect is active. IMS Connect with this status has at least one socket connection to the remote IMS Connect identified in the RMTIMSCON field.

NOTACTIVE

Selects remote IMS Connect resources that have a status of NOTACTIVE, meaning that the connection to the remote IMS Connect is not active. IMS Connect with this status does not have any socket connections to the remote IMS Connect identified in the RMTIMSCON field.

RETRYCONN

Selects remote IMS Connect resources that have a status of RETRYCONN, meaning that IMS Connect had failed to make a connection to the remote IMS Connect and is retrying to make a connection every two minutes.

STOPPED

Selects remote IMS Connect resources that have a status of STOPPED. A STOPRMT command has stopped communications between this IMS Connect and the remote IMS Connect identified in the RMTIMSCON field. Any messages to be sent to the IMS Connect that is shown in the RMTIMSCON field is rejected, and an error is sent back to the requester. The STOPRMT command is equivalent to the UPD IMSCON TYPE(RMTIMSCON) NAME(...) STOP(COMM) command.

Usage notes

You can issue the QUERY IMSCON TYPE(RMTIMSCON) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Typically, this command results in one output display line for each remote IMS Connect definition being displayed. However, if the SHOW(SENDCLNT) keyword is specified, an additional line is displayed for each send client associated with the specified remote IMS Connect definition.

Equivalent WTOR and z/OS commands

The following table lists WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(RMTIMSCON) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 79. WTOR and IMS Connect z/OS equivalents for the QUERY IMSCON TYPE(RMTIMSCON) command

QUERY IMSCON TYPE(RMTIMSCON) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
QUERY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL | *show_parm*)	VIEWRMT ALL	QUERY RMTIMSCON NAME(*)
QUERY IMSCON TYPE(RMTIMSCON) NAME(*rmtimscon_name*) SHOW(ALL | *show_parm*)	VIEWRMT *rmtimscon_name*	QUERY RMTIMSCON NAME(*rmtimscon_name*)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 80. Output fields for the QUERY IMSCON TYPE(RMTIMSCON) command

Short label	Long label	Keyword	Meaning
APPL	Appl	APPL	The APPL name to use for generating the RACF PassTicket, which is sent to the remote IMS Connect and be used for authenticating the user ID.
AUTC	AutoConn	AUTOCONN	Indicates whether this instance of IMS Connect automatically connects to the remote IMS Connect when this instance of IMS Connect starts. The value can be one of the following: **Y** Connection is automatic. **N** Connection is not automatic.

Table 80. Output fields for the QUERY IMSCON TYPE(RMTIMSCON) command (continued)

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 82 on page 261.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
HOST	HostName	HOSTNAME	The host name of the remote IMS Connect.
IP	IpAddress	IPADDR	The IP address of the remote IMS Connect.
ITO	IdleTO	IDLETO	The idle timeout interval to keep this socket connection before closing it due to inactivity.
LPLK	LclPlkID	SENDCLNT	The local MSC physical link ID that is using this connection, as specified on the LCLPLKID parameter of the MSC statement in the IMS Connect configuration member.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
NSOC	NumSoc	NUMSOC	The current number of sockets being used for this remote IMS Connect connection.
PERS	Persist	PERSISTENT	Indicates whether the connections to the remote IMS Connect are persistent. The value can be one of the following: **Y** Connection is persistent. **N** Connection is not persistent.
PORT	Port	PORT	The port number of the remote IMS Connect that this IMS Connect instance communicates on.
RIC	RmtImsCon	N/A	Remote IMS Connect name. The remote IMS Connect name is always returned.
RSOC	ResvSoc	RESVSOC	The number of send sockets available for this remote IMS Connect connection. The value is defined in the RESVSOC parameter of the RMTIMSCON statement in the IMS Connect configuration.
SCL	SendClnt	SENDCLNT	The client ID that this instance of IMS Connect used to connect to the remote IMS Connect.
SCSTT	SendStatus	SENDCLNT	Status of the send client connection. The status can be one of the following: **CONN** This connection is in connect state. It can send messages to the remote IMS Connect. **RECV** This connection is in receive state. It is waiting to receive a response from the remote IMS Connect.
SEC	Second	SENDCLNT	Number of seconds that this connection is in the specified state.
SPORT	SendPort	SENDCLNT	The port that this instance of IMS Connect used to connect to the remote IMS Connect.

Table 80. Output fields for the QUERY IMSCON TYPE(RMTIMSCON) command (continued)

Short label	Long label	Keyword	Meaning
STT	Status	STATUS	Status of the remote IMS Connect connection. The status can be one of the following:
			ACTIVE
			The connection to the remote IMS Connect is active. IMS Connect with this status has at least one socket connection to the remote IMS Connect identified in the RMTIMSCON field.
			NOTACTIVE
			The connection to the remote IMS Connect is not active. IMS Connect with this status does not have any socket connections to the remote IMS Connect identified in the RMTIMSCON field.
			RETRYCONN
			IMS Connect had failed to make a connection to the remote IMS Connect. The failure was due to the local TCP/IP, remote TCP/IP, or remote IMS Connect not being available. IMS Connect will wait 2 minutes and retry connecting again.
			When the connection to the remote IMS Connect is established, IMS Connect will notify OTMA to start sending remote IMS messages that uses this specific remote IMS Connect RMTIMSCON.
			STOPPED
			A STOPRMT command has stopped communications between this IMS Connect and the remote IMS Connect identified in the RMTIMSCON field. Any messages to be sent to the IMS Connect that is shown in RMTIMSCON field is rejected, and an error is sent back to the requester. The STOPRMT command is equivalent to the UPD IMSCON TYPE(RMTIMSCON) NAME(...) STOP(COMM) command.
SUID	SendUID	SENDCLNT	The user ID specified by the local IMS to be sent to the remote IMS for transaction authorization. This field is valid for OTMA messages only.
TSCL	TotSClnts	SUMMARY	Total number of send clients that are active on the remote IMS Connect.
TCON	TotConn	SUMMARY	Total number of send clients that have a status of CONN on the remote IMS Connect.
TOTH	TotOther	SUMMARY	Total number of send clients that have a status other than CONN, RECV, or XMIT on the remote IMS Connect.
TRCV	TotRecv	SUMMARY	Total number of send clients that have a status of RECV on the remote IMS Connect.

Table 80. Output fields for the QUERY IMSCON TYPE(RMTIMSCON) command (continued)

Short label	Long label	Keyword	Meaning
TXMT	TotXmit	SUMMARY	Total number of send clients that have a status of XMIT on the remote IMS Connect.
UID	UserID	USERID	The user ID to use for generating the RACF PassTicket, which is sent to the remote IMS Connect and be used for authenticating the user ID.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(RMTIMSCON) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 81. Return and reason codes for the QUERY IMSCON TYPE(RMTIMSCON) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(RMTIMSCON) command was completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 82. Completion codes for the QUERY IMSCON TYPE(RMTIMSCON) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(RMTIMSCON) command was completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for QUERY IMSCON TYPE(RMTIMSCON) command

TSO SPOC input:

```
QUERY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL)
```

TSO SPOC output:

```
(Screen 1)
 RmtImsCon MbrName CC  IpAddress       HostName      Port  AutoConn
 CONNECT2  HWS1    0   010.100.200.002 ICON2.IBM.COM 5555  N
 CONNECT2  HWS1    0
 CONNECT2  HWS1    0
 CONNECT3  HWS1    0   010.100.200.003 ICON3.IBM.COM 9999  Y
 CONNECT3  HWS1    0
 CONNECT3  HWS1    0
 CONNECT3  HWS1    0
 CONNECT3  HWS1    0
 (Screen 2)
 RmtImsCon MbrName Persist IdleTO ResvSoc NumSoc Appl   UserID Status
 CONNECT2  HWS1    Y         6000     10      2 APPL02 USER01 ACTIVE
 CONNECT2  HWS1
 CONNECT2  HWS1
 CONNECT3  HWS1    Y         6000      4      4 APPL03 USER01 ACTIVE
 CONNECT3  HWS1
 CONNECT3  HWS1
 CONNECT3  HWS1
 CONNECT3  HWS1
(Screen 3)
 RmtImsCon MbrName TotSClnts TotRecv TotConn TotXmit TotOther
 CONNECT2  HWS1    2         0       2       0       0
 CONNECT2  HWS1
 CONNECT2  HWS1
 CONNECT3  HWS1    4         1       3       0       0
 CONNECT3  HWS1
 CONNECT3  HWS1
 CONNECT3  HWS1
 CONNECT3  HWS1
(Screen 4)
 RmtImsCon MbrName SendClnt SendUID LclPlkID Second SendPort SendStatus
 CONNECT2  HWS1
 CONNECT2  HWS1    MSC11111         MSC12      100    1234 CONN
 CONNECT2  HWS1    MSC22222         MSC12       89    5678 CONN
 CONNECT3  HWS1
 CONNECT3  HWS1    OTM11111 USER01             100    1111 CONN
 CONNECT3  HWS1    OTM22222 USER01              89    2222 CONN
 CONNECT3  HWS1    OTM33333 USER02              81    3333 CONN
 CONNECT3  HWS1    OTM44444 USER03              23    4444 RECV
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 01:13:52.694135</statime>
<stotime>2010.298 01:13:52.695758</stotime>
<staseq>C6C76FD554B77EFA</staseq>
<stoseq>C6C76FD5551CEE7A</stoseq>
<rqsttkn1>USRT001 10181352</rqsttkn1>
```

```
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON        </kwd>
<input>QUERY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="RIC" llbl="RmtImsCon" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="IP" llbl="IpAddress" scope="LCL" sort="n" key="0"
scroll="yes" len="15" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="HOST" llbl="HostName" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left"skipb="yes" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="AUTC" llbl="AutoConn" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PERS" llbl="Persist" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ITO" llbl="IdleTO" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="RSOC" llbl="ResvSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="NSOC" llbl="NumSoc" scope="LCL" sort="n" key="0"
 scroll="yes" len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="APPL" llbl="Appl" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TSCL" llbl="TotSClnts" scope="LCL" sort="d" key="4"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TRCV" llbl="TotRecv" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TCON" llbl="TotConn" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TXMT" llbl="TotXmit" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="a" key="3"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SUID" llbl="SendUID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPLK" llbl="LclPlkID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SPORT" llbl="SendPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SCSTT" llbl="SendStatus" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RIC(CONNECT2) MBR(HWS1          ) CC(   0) SCL(MSC11111)
LPLK(MSC12   ) SEC(100) SPORT(1234) SCSTT(CONN) </rsp>
```

```
<rsp>RIC(CONNECT2) MBR(HWS1           ) CC(   0) SCL(MSC22222)
 LPLK(MSC12   ) SEC(89) SPORT(5678) SCSTT(CONN) </rsp>
<rsp>RIC(CONNECT2) MBR(HWS1           ) CC(   0) IP(010.100.200.002)
 HOST(ICON2.IBM.COM) PORT(5555) AUTC(N) PERS(Y) ITO(6000)
 RSOC(10) NSOC(2) APPL(APPL02  ) UID(USER01  ) STT(ACTIVE   ) TSCL(2)
 TRCV(0) TCON(2) TXMT(0) TOTH(0) </rsp>
<rsp>RIC(CONNECT3) MBR(HWS1           ) CC(   0) IP(010.100.200.003)
 HOST(ICON3.IBM.COM) PORT(9999) AUTC(Y) PERS(Y) ITO(6000)
 RSOC(4) NSOC(4) APPL(APPL03  ) UID(USER01  ) STT(ACTIVE   ) TSCL(4)
TRCV(1) TCON(3) TXMT(0) TOTH(0) </rsp>
<rsp>RIC(CONNECT3) MBR(HWS1           ) CC(   0) SCL(OTM11111) SUID(USER01  )
 SEC(100) SPORT(1111) SCSTT(CONN) </rsp>
<rsp>RIC(CONNECT3) MBR(HWS1           ) CC(   0) SCL(OTM22222) SUID(USER01  )
 SEC(89) SPORT(2222) SCSTT(CONN) </rsp>
<rsp>RIC(CONNECT3) MBR(HWS1           ) CC(   0) SCL(OTM33333) SUID(USER02  )
 SEC(81) SPORT(3333) SCSTT(CONN) </rsp>
<rsp>RIC(CONNECT3) MBR(HWS1           ) CC(   0) SCL(OTM44444) SUID(USER03  )
 SEC(23) SPORT(4444) SCSTT(CONN) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: There are two RMTIMSCON definitions in IMS Connect: CONNECT2 and CONNECT3. Each remote connection is active. CONNECT2 is used for MSC purposes and has two MSC links active. CONNECT3 is used for OTMA purposes and has four connections active.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

QUERY IMSCON TYPE(SENDCLNT) command

Use the QUERY IMSCON TYPE(SENDCLNT) command to display the status and activity of one or more active send client socket connections with another instance of IMS Connect or with a CICS subsystem.

IMS Connect uses a separate receive socket on a different port to receive transactions and reply data from another IMS Connect instance or from a CICS subsystem. The QUERY IMSCON TYPE(SENDCLNT) command does not display receive client socket connections. To display information about receive client socket connections, use the QUERY IMSCON TYPE(CLIENT) command.

Subsections:
- "Environment"
- "Syntax" on page 265
- "Keywords" on page 265
- "Usage notes" on page 267
- "Equivalent WTOR and z/OS commands" on page 267
- "Output fields" on page 267
- "Return, reason, and completion codes" on page 269
- "IMS to IMS connections example" on page 270

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).

- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax
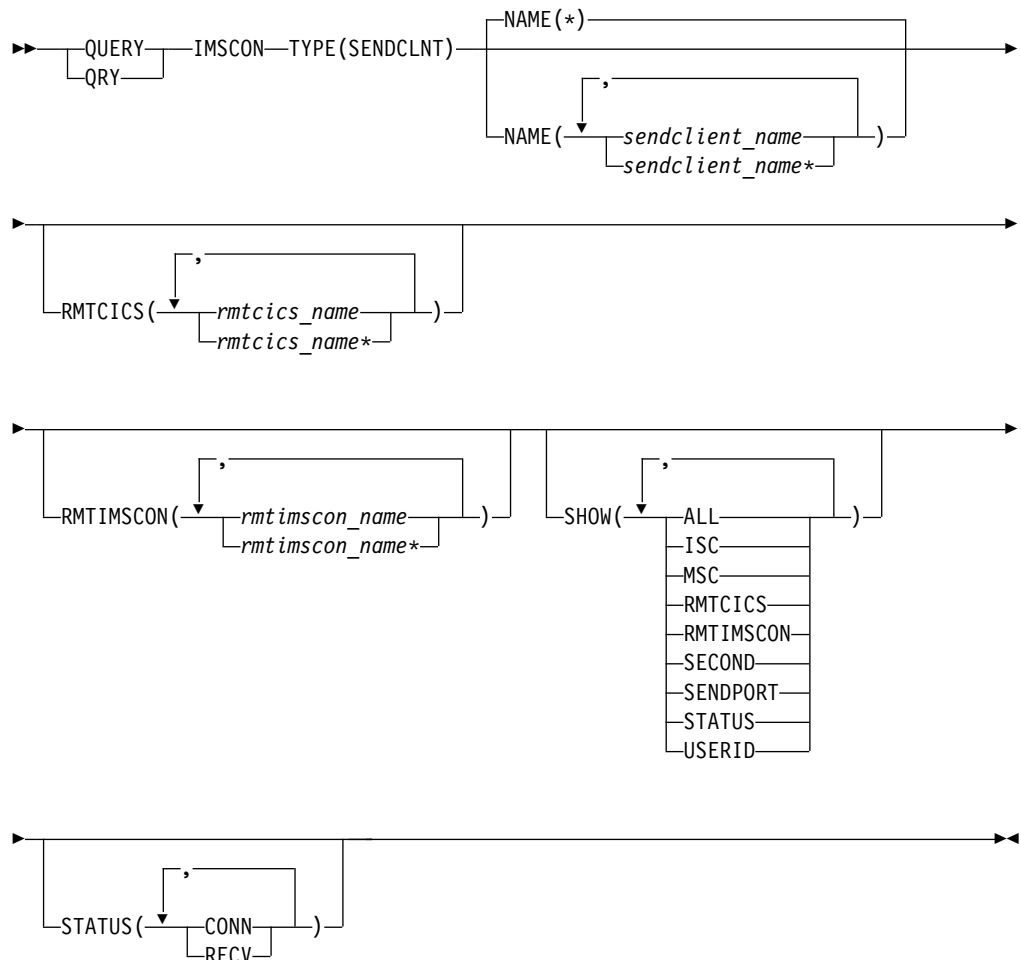
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(SENDCLNT) command.

NAME

Specifies one or more send client resources to be displayed. You can specify a single send client name or a list of send client names separated by commas. Wildcards can be used in the names.

You can specify NAME(*) to display all send client resources. NAME(*) is the default.

IMS Connect always generates the client name for connections with another IMS Connect instance or a CICS subsystem. In these cases, the first characters of the client name identify the IMS communication type. For example:

ISC ISC communication with a CICS subsystem

MSC MSC communication between IMS systems

OTM OTMA communication between IMS systems

To display the active socket connections for one of the preceding types of IMS communication, you can specify the character identifier followed by a wildcard character. For example, NAME(MSC*), returns all IMS-to-IMS send socket connections for MSC communications.

RMTCICS

Selects send client resources for display that are associated with the specified remote CICS name, as defined on the ID keyword of the RMTCICS statement in the IMS Connect configuration member.

You can specify a single remote CICS name or a list of remote CICS names separated by commas. Wildcard characters are supported.

Only the send client resources that are associated with the specified remote CICS name are displayed. Send clients that match the NAME() parameter, but that are not associated with the specified remote CICS name, are not displayed.

When the RMTCICS filter is specified, remote CICS resource information is displayed even if SHOW(RMTCICS) is not specified.

RMTIMSCON

Selects send client resources for display that are associated with the specified remote IMS Connect definition. You can specify a single remote IMS Connect name or a list of remote IMS Connect names separated by commas. Wildcards can be used in the names.

Only those send client resources that are associated with the specified remote IMS Connect resource are displayed. Send clients that match the NAME() parameter, but are not associated with the specified remote IMS Connect name, are not displayed.

When the RMTIMSCON filter is specified, remote IMS Connect resource information is displayed even if SHOW(RMTIMSCON) is not specified.

SHOW

Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the send client name, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL

Displays all output fields.

ISC

Displays the name of the Intersystem Communication (ISC) resource that this send client is associated with. This value is specified on the ID parameter of the ISC statement in the IMS Connect configuration member.

MSC

Displays the name of the Multiple Systems Coupling (MSC) physical link that this send client is associated with. This value is specified on the LCLPLKID parameter of the MSC statement in the IMS Connect configuration member.

RMTCICS

Displays the name of the remote CICS resource that this send client is associated with.

RMTIMSCON
Displays the name of the remote IMS Connect resource that this send client is associated with.

SECOND
Displays the number of seconds that this connection is in the specified state.

SENDPORT
Displays the local port of the send client socket connection.

STATUS
Displays the state of the send client connection. For a description of the status returned, see the STATUS keyword in Table 83 on page 268.

USERID
Displays the user ID specified by the local IMS to be sent to the remote IMS for transaction authorization. This field is valid for OTMA messages only.

STATUS
Selects send clients for display that possess at least one of the specified statuses. When the STATUS filter is specified, status information is displayed even if SHOW(STATUS) is not specified.

The filters supported with the STATUS keyword, which can be specified in any order, are:

CONN
Selects send client connections that have a status of CONN, meaning that the connection is in connect state.

RECV
Selects send client connections that have a status of RECV, meaning that the connection is in receive state.

Usage notes

The port number for each port displayed is repeated on each line of information that applies to that port. The first line of information for a port shows the status and statistics for the port. Each subsequent line for the port shows information about an active client socket on the port.

You can issue the QUERY IMSCON TYPE(SENDCLNT) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Equivalent WTOR and z/OS commands

There are no equivalent WTOR and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(SENDCLNT) command.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 83. Output fields for the QUERY IMSCON TYPE(SENDCLNT) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 85 on page 269.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
ISC	ISCName	ISC	Displays the name of the Intersystem Communication (ISC) resource that this send client is associated with. This value is specified on the ID parameter of the ISC statement in the IMS Connect configuration member.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
MSC	MscName	MSC	Displays the name of the Multiple Systems Coupling (MSC) physical link that this send client is associated with. This value is specified on the LCLPLKID parameter of the MSC statement in the IMS Connect configuration member.
RCICS	RmtCICS	RMTCICS	Name of the remote CICS resource associated with the send client.
RIC	RmtImsCon	RMTIMSCON	Name of the remote IMS Connect resource associated with the send client.
SCL	SendClnt	N/A	Name of the send client; that is, the client ID that this instance of IMS Connect used to connect to the remote IMS Connect. The send client name is always returned.
SEC	Second	SECOND	Number of seconds that this connection is in the specified state.
STT	Status	STATUS	Status of the send client connection, which is one of the following: **CONN** This connection is in connect state. It can send messages to the remote IMS Connect. **RECV** This connection is in receive state. It is waiting for a response from the remote IMS Connect.
SPORT	SendPort	SENDPORT	The port that this instance of IMS Connect used to connect to the remote IMS Connect.

Table 83. Output fields for the QUERY IMSCON TYPE(SENDCLNT) command (continued)

Short label	Long label	Keyword	Meaning
UID	UserID	USERID	The user ID specified by the local IMS to be sent to the remote IMS for transaction authorization. This field is valid for OTMA messages only.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(SENDCLNT) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 84. Return and reason codes for the QUERY IMSCON TYPE(SENDCLNT) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(SENDCLNT) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C000008'	X'00002004'	An invalid keyword or keyword parameter was specified.
X'0C000008'	X'00002014'	An invalid character was specified in the NAME() parameter.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 85. Completion codes for the QUERY IMSCON TYPE(SENDCLNT) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(SENDCLNT) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

IMS to IMS connections example

In the following example, there are two RMTIMSCON definitions in IMS Connect: CONNECT2 and CONNECT3. Each remote connection is active. CONNECT2 is used for MSC purposes and has two MSC links active. CONNECT3 is used for OTMA purposes and has four connections active.

TSO SPOC input:
```
QUERY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(ALL)
```

TSO SPOC output:
```
(Screen 1)
 SendClnt MbrName CC UserID MscName Second SendPort RmtImsCon Status
 MSC11111 HWS1    0         MSC12      100     1234 CONNECT2  CONN
 MSC22222 HWS1    0         MSC12       89     5678 CONNECT2  CONN
 OTM11111 HWS1    0  USER01             100     1111 CONNECT3  CONN
 OTM22222 HWS1    0  USER01              89     2222 CONNECT3  CONN
 OTM33333 HWS1    0  USER02              81     3333 CONNECT3  CONN
 OTM44444 HWS1    0  USER03              23     4444 CONNECT3  CONN
```

OM API input:
```
CMD ( QUERY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(ALL) )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 01:50:08.581654</statime>
<stotime>2010.298 01:50:08.582765</stotime>
<staseq>C6C777F06B41662C</staseq>
<stoseq>C6C777F06B86D1EC</stoseq>
<rqsttkn1>USRT001 10185008</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="lfet" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSC" llbl="MscName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SPORT" llbl="SendPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="RIC" llbl="RmtImsCon" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
```

```
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>SCL(MSC11111) MBR(HWS1            ) CC(   0) MSC(MSC12   )
 SEC(100) SPORT(1234) RIC(CONNECT2) STT(CONN) </rsp>
<rsp>SCL(MSC22222) MBR(HWS1            ) CC(   0) MSC(MSC12   )
 SEC(89) SPORT(5678) RIC(CONNECT2) STT(CONN) </rsp>
<rsp>SCL(OTM11111) MBR(HWS1            ) CC(   0) UID(USER01  ) SEC(100)
 SPORT(1111) RIC(CONNECT3) STT(CONN) </rsp>
<rsp>SCL(OTM22222) MBR(HWS1            ) CC(   0) UID(USER01  ) SEC(89)
 SPORT(2222) RIC(CONNECT3) STT(CONN) </rsp>
<rsp>SCL(OTM33333) MBR(HWS1            ) CC(   0) UID(USER02  ) SEC(81)
 SPORT(3333) RIC(CONNECT3) STT(CONN) </rsp>
<rsp>SCL(OTM44444) MBR(HWS1            ) CC(   0) UID(USER03  ) SEC(23)
 SPORT(4444) RIC(CONNECT3) STT(CONN) </rsp>
</cmdrspdata>
</imsout>
```

ISC TCP/IP connections example

In the following example, two send clients are identified by names that begin with
ISC. Both send clients are associated with the remote CICS resource CICS2 and the
ISC resource IMS1CIC1.

TSO SPOC input:
```
QUERY IMSCON TYPE(SENDCLNT) NAME(ISC*) SHOW(ALL)
```

TSO SPOC output:
```
SendClnt MbrName    CC ISCName      Second SendPort RmtCICS  Status
ISCC9890 HWS1        0 IMS1CIC1         41     1396 CICS2    CONN
ISC66F93 HWS1        0 IMS1CIC1         19     1397 CICS2    CONN
```

OM API input:
```
CMD ( QUERY IMSCON TYPE(SENDCLNT) NAME(ISC*) SHOW(ALL) )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2012.102 21:47:26.546443</statime>
<stotime>2012.102 21:47:26.548301</stotime>
<staseq>C967E78294E0BE1B</staseq>
<stoseq>C967E7829554D91B</stoseq>
<rqsttkn1>USRT001 10144726</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QUERY IMSCON TYPE(SENDCLNT) NAME(ISC*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
```

```
               len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSC" llbl="MscName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SEC" llbl="Second" scope="LCL" sort="n" key="0" scroll="yes"
 len="10" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SPORT" llbl="SendPort" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="RIC" llbl="RmtImsCon" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RCICS" llbl="RmtCICS" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
 len="9" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>SCL(ISC66F93) MBR(HWS1            ) CC(   0) ISC(IMS1CIC1) SEC(19)
 SPORT(1397) RCICS(CICS2   ) STT(CONN) </rsp>
<rsp>SCL(ISCC9890) MBR(HWS1            ) CC(   0) ISC(IMS1CIC1) SEC(41)
 SPORT(1396) RCICS(CICS2   ) STT(CONN) </rsp>
</cmdrspdata>
</imsout>
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

QUERY IMSCON TYPE(UOR) command

Use the QUERY IMSCON TYPE(UOR) command to display the status and activity of one or more unit of recovery (UOR) identifiers in IMS Connect.

Subsections:
- "Environment"
- "Syntax" on page 273
- "Keywords" on page 273
- "Usage notes" on page 275
- "Equivalent WTOR and z/OS commands" on page 275
- "Output fields" on page 275
- "Return, reason, and completion codes" on page 277
- "Examples" on page 278

Environment

The QUERY IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
>>-+-QUERY-+-IMSCON-TYPE(UOR)-+-NAME(*)------------------------+->
   +-QRY---+                  |           ,                    |
                             |          v----------+          |
                             +-NAME(-+--+-uor_name--+---+-)----+
                                     |  +-uor_name*--+   |
                                     +-----------------+

>--+---------------------+--+-----------------------------+-><
   |         ,           |  |            ,                |
   |        v------------+  |           v-----------------+
   +-SHOW(-+--+-ALL-----+-) +-STATE(-+--+-FORGOTTEN-----+-)
             +-STATE----+             +-INBACKOUT-----+
             +-SUMMARY--+             +-INCOMMIT------+
             +-XID------+             +-INCOMPLETION--+
                                      +-INDOUBT-------+
                                      +-INEND---------+
                                      +-INFLIGHT------+
                                      +-INFORGET------+
                                      +-INONLYAGENT---+
                                      +-INPREPARE-----+
                                      +-INRESET-------+
                                      +-INSTATECHECK--+
```
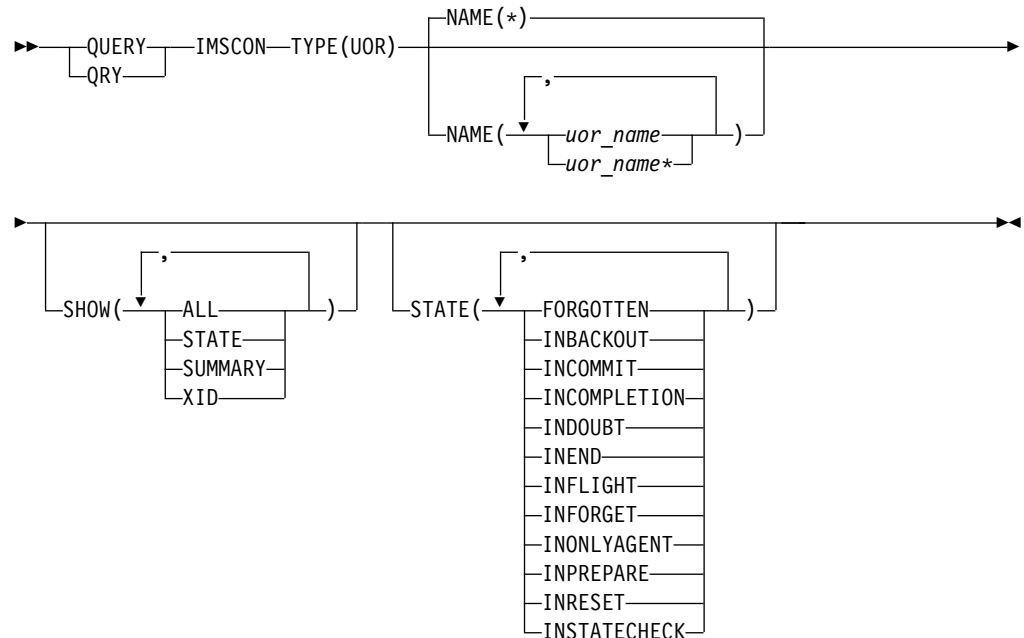
Keywords

The following keywords are valid for the QUERY IMSCON TYPE(UOR) command.

NAME
Specifies one or more UOR identifiers to be displayed. You can specify a single UOR ID or a list of UOR identifiers separated by commas. Wildcards can be used in the identifiers.

You can specify NAME(*) to display all UOR identifiers. NAME(*) is the default.

SHOW
Specifies the optional output fields to be displayed. Output fields that are always displayed, regardless of whether SHOW is specified, include the UOR identifier, the name of the IMS Connect that processes the command, and the completion code.

The filters that are supported with the SHOW keyword, which can be specified in any order, are:

ALL
Displays all output fields.

STATE
Displays the state of the UOR. For a description of the states returned, see the STATE keyword in Table 87 on page 276.

SUMMARY
Displays summary information related to all UORs specified by the NAME keyword. This information is displayed on its own output line. Specific UORs are displayed on separate output lines.

Information displayed includes:

- Total number of UORs in any state.
- Total number of UORs in IN_DOUBT state.
- Total number of UORs in IN_BACKOUT state.
- Total number of UORs in IN_COMMIT state.
- Total number of UORs not in one of these states.

XID
Displays the X/Open identifier, which identifies the distributed transaction used by the X/Open architecture. The XID consists of four parts:

FMID 4-byte fixed-format ID

GTRID
4-byte fixed GTRID length

BQUAL
4-byte fixed BQUAL length

XID 128-byte character XID

STATE
Selects UORs for display that are in at least one of the specified states.

The filters supported with the STATE keyword, which can be specified in any order, are:

FORGOTTEN
Selects UORs for display that have a status of FORGOTTEN.

INBACKOUT
Selects UORs for display that have a status of IN_BACKOUT.

INCOMMIT
Selects UORs for display that have a status of IN_COMMIT.

INCOMPLETION
Selects UORs for display that have a status of IN_COMPLETION.

INDOUBT
Selects UORs for display that have a status of IN_DOUBT.

INEND
Selects UORs for display that have a status of IN_END.

INFLIGHT
Selects UORs for display that have a status of IN_FLIGHT.

INFORGET
Selects UORs for display that have a status of IN_FORGET.

INONLYAGENT
Selects UORs for display that have a status of IN_ONLY_AGENT.

INPREPARE
Selects UORs for display that have a status of IN_PREPARE.

INRESET
Selects UORs for display that have a status of IN_RESET.

INSTATECHECK
Selects UORs for display that have a status of IN_STATE_CHECK.

Usage notes

You can issue the QUERY IMSCON TYPE(UOR) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Typically, this command results in one output display line for each UOR displayed. However, if the SHOW(SUMMARY) keyword is specified, one additional line is displayed for each specific or generic name specified by the NAME() keyword. In this line, summary totals for UORs are displayed in various states. For example, if NAME(*) is specified, IMS Connect builds one output line with a UOR name of '*' to display the totals, and one additional output line for each UOR found in IMS Connect.

Equivalent WTOR and z/OS commands

The following table lists WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the QUERY IMSCON TYPE(UOR) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 86. WTOR and IMS Connect z/OS equivalents for the QUERY IMSCON TYPE(UOR) command

QUERY IMSCON TYPE(UOR) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
QUERY IMSCON TYPE(UOR) NAME(*) SHOW(ALL | *show_parm*)	VIEWUOR ALL	QUERY UOR NAME(*) SHOW(ALL)
QUERY IMSCON TYPE(UOR) NAME(*uor_id*) SHOW(ALL | *show_parm*)	VIEWUOR *uor_id*	QUERY UOR NAME(*uor_id*) SHOW(ALL)
QUERY IMSCON TYPE(UOR) STATE(*state*)	None	None

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
Provides a brief description of the output field.

Table 87. Output fields for the QUERY IMSCON TYPE(UOR) command

Short label	Long label	Keyword	Meaning
BQUAL	BqualLen	XID	4-byte fixed BQUAL length, which is part of the XID.
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 90 on page 278.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
FMID	Fmid	XID	4-byte fixed-format ID, which is part of the XID.
GTRID	GtridLen	XID	4-byte fixed GTRID length, which is part of the XID.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
STATE	State	STATE	State of the UOR. For a description of the possible states that are returned, see Table 88.
TIBO	TotInBackout	SUMMARY	Total number of UORs that have a status of IN_BACKOUT.
TIC	TotInCommit	SUMMARY	Total number of UORs that have a status of IN_COMMIT.
TID	TotInDoubt	SUMMARY	Total number of UORs that have a status of IN_DOUBT.
TOTH	TotOther	SUMMARY	Total number of UORs that have a status other than IN_DOUBT, IN_BACKOUT, or IN_COMMIT.
TUOR	TotalUor	SUMMARY	Total number of UORs displayed for the NAME specified.
URID	Urid	N/A	The 16-byte character string that identifies a specific unit of recovery. The UOR identifier is always returned.
XID	Xid	XID	128-byte character XID.

Table 88. States of the UOR for the QUERY IMSCON TYPE(UOR) command

State	Meaning
FORGOTTEN	The UOR has completed, and z/OS Resource Recovery Services (RRS) has deleted its log records.
IN_BACKOUT	One of the following actions occurred: • One or more PREPARE exit routines replied NO. • The application issued a backout. • The DSRM or SDSRM told RRS to back out an IN_DOUBT UOR. • The installation used the RRS panels to back out an IN_DOUBT UOR. • Before phase 2 of the two-phase-commit protocol, the system, application, RRS, or a resource manager failed.

Table 88. States of the UOR for the QUERY IMSCON TYPE(UOR) command (continued)

State	Meaning
IN_COMMIT	One of the following actions occurred: • The PREPARE exit routines replied YES. • The DSRM or SDSRM told RRS to commit an IN_DOUBT UOR. • The installation used the RRS panels to commit an IN_DOUBT UOR.
IN_COMPLETION	The resources have been updated, and RRS has completed processing the UOR.
IN_DOUBT	RRS is waiting for the resource manager to tell it whether to resolve the UOR by a commit or by a backout.
IN_END	The resources have been updated.
IN_FLIGHT	The UOR can access resources and has the potential to change resources, but the changes are not committed.
IN_FORGET	During distributed processing, the UOR has completed, but RRS is waiting for the SDSRM to indicate how long it takes to process the log records for the UOR.
IN_ONLY_AGENT	Only one resource manager expressed interest in the UOR.
IN_PREPARE	The UOR in the proper state issues a commit, and RRS invokes the PREPARE exit routine.
IN_RESET	The UOR is starting and has not yet changed any resources.
IN_STATE_CHECK	The UOR issues a commit and waits for the STATE_CHECK exit routine of the resource manager to check if the resources are in the correct state.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSCON TYPE(UOR) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 89. Return and reason codes for the QUERY IMSCON TYPE(UOR) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSCON TYPE(UOR) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 90. Completion codes for the QUERY IMSCON TYPE(UOR) command

Completion code	Completion code text	Meaning
0		The QUERY IMSCON TYPE(UOR) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for QUERY IMSCON TYPE(UOR) command

TSO SPOC input:
```
QUERY IMSCON TYPE(UOR) SHOW(STATE)
```

TSO SPOC output:
```
Urid                             MbrName CC State
C3A3DE827DE5500000000000601010000 HWS1     0 IN_FLIGHT
```

OM API input:
```
CMD ( QUERY IMSCON TYPE(UOR) SHOW(STATE) )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 16:17:22.927883</statime>
<stotime>2010.298 16:17:22.928933</stotime>
<staseq>C6C839C83D10BAAC</staseq>
<stoseq>C6C839C83D5252AC</stoseq>
<rqsttkn1>USRT001 10091722</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QRY IMSCON TYPE(UOR) SHOW(STATE) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="URID" llbl="Urid" scope="LCL" sort="a" key="2" scroll="no"
 len="32" dtype="CHAR" align="left"skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TUOR" llbl="TotalUor" scope="LCL" sort="n" key="0"
scroll="yes" len="7" dtype="INT" align="right" skipb="yes" />
```

```
<hdr slbl="TID" llbl="TotInDoubt" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TIBO" llbl="TotInBackout" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TIC" llbl="TotInCommit" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="STATE" llbl="State" scope="LCL" sort="n" key="0"
 scroll="yes" len="14" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="FMID" llbl="Fmid" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="LEFT" skipb="yes" />
<hdr slbl="GTRID" llbl="GtridLen" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="BQUAL" llbl="BqualLen" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="XID" llbl="Xid" scope="LCL" sort="n" key="0" scroll="yes"
 len="256" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>URID(C3A3DE827DE550000000000006601010000) MBR(HWS1          ) CC(
 0) STATE(IN_FLIGHT     ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: There is one UOR active in IMS Connect. There is only one display
line because the SHOW(SUMMARY) keyword was not specified. The NAME
keyword is omitted, so IMS Connect displays all data stores (the default is
NAME(*)).

Example 2 for QUERY IMSCON TYPE(UOR) command

TSO SPOC input:
```
QUERY IMSCON TYPE(UOR) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
Urid                                MbrName CC TotalUor TotInDoubt
*                                   HWS1     0        1          0
C3A3DE827DE550000000000006601010000 HWS1     0

(screen 2)
Urid                                MbrName TotInBackout TotInCommit
*                                   HWS1               0           0
C3A3DE827DE550000000000006601010000 HWS1

(screen 3)
Urid                                MbrName TotOther State
*                                   HWS1           1
C3A3DE827DE550000000000006601010000 HWS1             IN_FLIGHT

(screen 4)
Urid                                MbrName Fmid     GtridLen BqualLen
*                                   HWS1
C3A3DE827DE550000000000006601010000 HWS1    57415344       26       25

(screen 5)
Urid                                MbrName Xid
*                                   HWS1
C3A3DE827DE550000000000006601010000 HWS1    000000180114B9767775F58D0A517C90

(screen 6)
Urid                                MbrName Xid
*                                   HWS1
```

```
C3A3DE827DE55000000000006601010000 HWS1    3AD5C4901BAB55D42C0701B9767775F5

(screen 7)
Urid                              MbrName Xid
*                                 HWS1
C3A3DE827DE55000000000006601010000 HWS1    8D0A517C903AD5C4901BAB55D42C075B

(screen 8)
Urid                              MbrName Xid
*                                 HWS1
C3A3DE827DE55000000000006601010000 HWS1    AA8D12000000000000000000000000000

(screen 9)
Urid                              MbrName Xid
*                                 HWS1
C3A3DE827DE55000000000006601010000 HWS1    000000000000000000000000000000000

(screen 10)
Urid                              MbrName Xid
*                                 HWS1
C3A3DE827DE55000000000006601010000 HWS1    000000000000000000000000000000000

(screen 11)
Urid                              MbrName Xid
*                                 HWS1
C3A3DE827DE55000000000006601010000 HWS1    000000000000000000000000000000000

(screen 12)
Urid                              MbrName Xid
*                                 HWS1
C3A3DE827DE55000000000006601010000 HWS1    000000000000000000000000000000000
```

OM API input:

```
CMD ( QUERY IMSCON TYPE(UOR) SHOW(ALL) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.300 04:59:35.813592</statime>
<stotime>2010.300 04:59:35.814615</stotime>
<staseq>C6CA2603FA5D84EA</staseq>
<stoseq>C6CA2603FA9D79AA</stoseq>
<rqsttkn1>USRT001 10215935</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>IMSCON          </kwd>
<input>QRY IMSCON TYPE(UOR) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="URID" llbl="Urid" scope="LCL" sort="a" key="2" scroll="no"
 len="32" dtype="CHAR" align="left"skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TUOR" llbl="TotalUor" scope="LCL" sort="n" key="0"
```

```
                     scroll="yes" len="7" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TID" llbl="TotInDoubt" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TIBO" llbl="TotInBackout" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TIC" llbl="TotInCommit" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TOTH" llbl="TotOther" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="yes" />
<hdr slbl="STATE" llbl="State" scope="LCL" sort="n" key="0"
 scroll="yes" len="14" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="FMID" llbl="Fmid" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="LEFT" skipb="yes" />
<hdr slbl="GTRID" llbl="GtridLen" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="BQUAL" llbl="BqualLen" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="XID" llbl="Xid" scope="LCL" sort="n" key="0" scroll="yes"
 len="256" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>URID(C3A3DE827DE550000000000006601010000) MBR(HWS1            ) CC(
 0) STATE(IN_FLIGHT    ) FMID(57415344) GTRID(26) BQUAL(25)
XID(000000180114B9767775F58D0A517C903AD5C4901BAB55D42C0701B9767775F58D
0A517C903AD5C4901BAB55D42C075BAA8D1200000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000)</rsp>
<rsp>URID(*                                 ) MBR(HWS1            ) CC(
 0) TUOR(1) TID(0) TIBO(0) TIC(0) TOTH(1) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: There is one UOR active in IMS Connect. The first display line
includes the summary total information for all UORs that follow. The second
display line represents the UOR, which includes its state and XID information. The
NAME keyword is omitted, so IMS Connect displays all data stores (the default is
NAME(*)).

Related concepts:

How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

VIEWUOR command (Commands)

IMS Connect QUERY UOR command (Commands)

QUERY IMSPLEX command

Use the QUERY IMSPLEX command, which is a type-2 command, to display
information about one or more IMSplex members.

Subsections:
- "Environment" on page 282
- "Syntax" on page 282
- "Keywords" on page 282
- "Usage notes" on page 284
- "Output fields" on page 286
- "Return, reason, and completion codes" on page 286
- "Examples" on page 288

Environment

The QUERY IMSPLEX command does not run in the address space of any IMS control or dependent region. QUERY IMSPLEX is processed in an OM command processing environment.
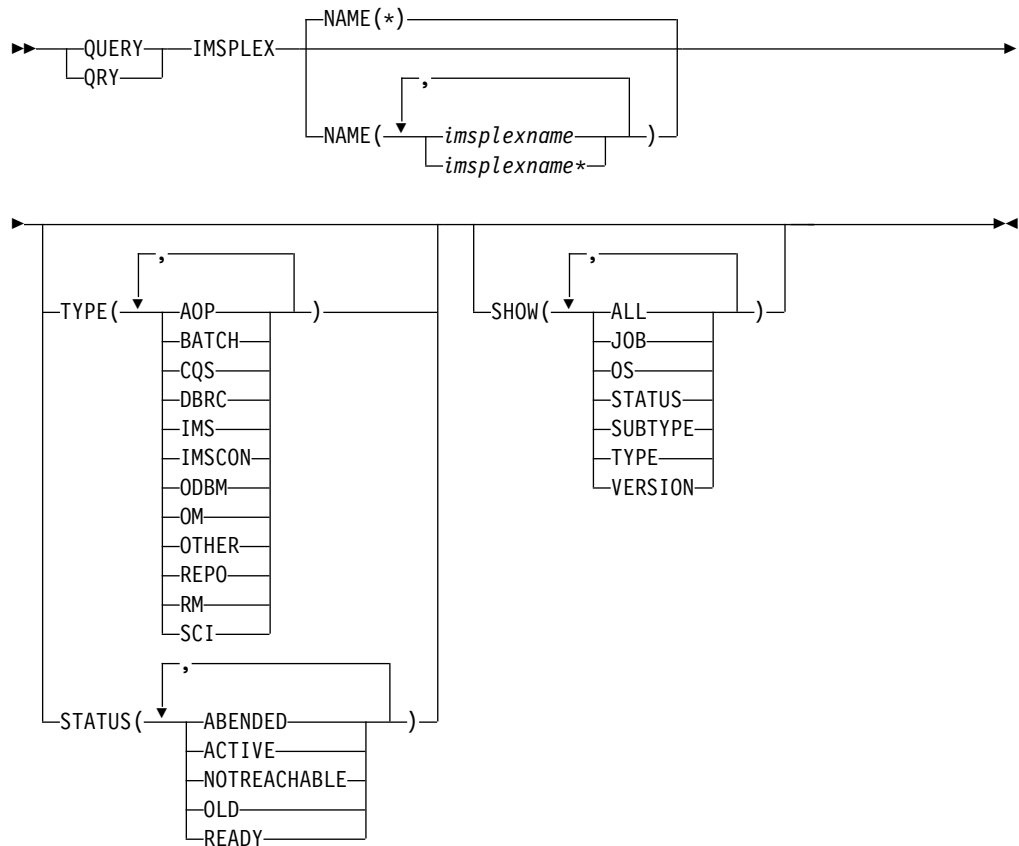
Syntax

```
►►─┬─QUERY─┬─IMSPLEX─┬──────────NAME(*)──────────┬─────────────────────►
   └─QRY───┘         │              ┌─,─────────┐ │
                     │              ▼           │ │
                     └─NAME(──┬──imsplexname──┬──)─┘
                             └─imsplexname*──┘
```

```
►─┬───────────────────────────┬──┬────────────────────────┬──►◄
  │        ┌─,───────┐         │  │        ┌─,────────┐    │
  │        ▼         │         │  │        ▼          │    │
  ├─TYPE(──┬─AOP────┬──)───────┤  └─SHOW(──┬─ALL─────┬──)──┘
  │        ├─BATCH──┤          │           ├─JOB─────┤
  │        ├─CQS────┤          │           ├─OS──────┤
  │        ├─DBRC───┤          │           ├─STATUS──┤
  │        ├─IMS────┤          │           ├─SUBTYPE─┤
  │        ├─IMSCON─┤          │           ├─TYPE────┤
  │        ├─ODBM───┤          │           └─VERSION─┘
  │        ├─OM─────┤          │
  │        ├─OTHER──┤          │
  │        ├─REPO───┤          │
  │        ├─RM─────┤          │
  │        └─SCI────┘          │
  │          ┌─,─────────────┐ │
  │          ▼               │ │
  └─STATUS(──┬─ABENDED──────┬──)─┘
             ├─ACTIVE───────┤
             ├─NOTREACHABLE─┤
             ├─OLD──────────┤
             └─READY────────┘
```

Keywords

The following keywords are valid for the QUERY IMPSLEX command:

NAME()
Specifies the name of the IMSplexes for which member information is to be returned. The IMSplex name can be a generic parameter, to allow easy specification of a group of IMSplexes whose names match a generic parameter mask. For example, QUERY IMSPLEX NAME(*PLEX*).

You must include the prefix, CSL, to the name of the IMSplex for which you want member information. Add CSL in front of the IMSplex name that you specified in the IMSPLEX= parameter in the DFSCGxxx PROCLIB member. For example, if you specified IMSPLEX=PLEX1 in your DFSCGxxx member, you must specify QUERY IMSPLEX NAME(CSLPLEX1).

SHOW()
Specifies the output fields to be returned. If SHOW is not specified, only the IMSplex names, IMSplex member names, IMSplex member that builds the output line, and completion codes are returned. This provides a method for a

system management application to obtain a list of all IMSplex member names that are currently known in the IMSplexes.

ALL
Returns all output fields.

JOB
Job name of the IMSplex member.

OS Name of the OS image on which the IMSplex member is executing.

STATUS
IMSplex member status.

SUBTYPE
IMSplex member subtype.

TYPE
IMSplex member type.

VERSION
IMSplex member version.

STATUS()
Displays IMSplex members that have at least one of the specified statuses.

ABENDED
IMSplex member has ended abnormally.

ACTIVE
IMSplex member is active.

NOTREACHABLE
The local SCI responsible for the member is not currently active. The status displayed is the current status for the member.

Note: The status output is NOT-REACHABLE (with a hyphen).

OLD
The SCI responsible for the member is not currently active. The status displayed is the last known status for the member. The actual status might be different.

READY
IMSplex member is ready to receive messages and requests that are routed to it by any method, including by TYPE.

TYPE()
Displays IMSplex members that possess at least one of the specified member types.

AOP
Automated Operator Program. Examples of AOPs are a SPOC application that an operator uses to interact with an IMSplex or a program that is monitoring an IMSplex.

BATCH
IMS batch job.

CQS
Common Queue Server address space.

DBRC
DBRC address space.

IMS
> IMS region.

IMSCON
> An address space that serves as an interface between IMS and a protocol that is not directly supported by IMS (for example, TCP/IP).

ODBM
> Open Database Manager address space.

OM Operations Manager address space.

OTHER
> Other non-IMS address space or job.

REPO
> Repository Server (RS) address space. The Repository Server information is returned if the Repository Server is registered to the Structured Call Interface (SCI) address space.

RM Resource Manager address space.

SCI
> Structured Call Interface address space.

Usage notes

This command can be issued only through the OM API.

QUERY IMSPLEX status

The following table shows the possible IMSplex member status. The table contains information about status such as the STATUS keyword to specify to select members with the specified status, the status that is returned, and the meaning of the status.

Table 91. QUERY IMSPLEX status table

Status keyword	Status	Meaning
ABENDED	ABENDED	IMSplex member has abended.
ACTIVE	ACTIVE	IMSplex member is active.
NOTREACHABLE	NOT-REACHABLE	The local SCI responsible for the member is currently not active. The status displayed is the current status for the member.
OLD	OLD	The SCI responsible for the member is not currently active. The status displayed is the last known status for the member. The actual status might be different.
READY	READY	IMSplex member is ready to receive messages and requests that are routed to it by any method, including by TYPE.

QUERY IMSPLEX types

The following table shows the possible IMSplex member types. The table contains information about member types such as the TYPE keyword to specify to select members with the specified type, the type that is returned, and the meaning of the member type.

Table 92. QUERY IMSPLEX member types

Type keyword	Member type	Meaning
AOP	aop	Automated Operator Program. An example of an AOP is a SPOC (Single Point of Control) that an operator uses to interact with the IMSplex. Another example of an AOP is a program that is monitoring the IMSplex.
BATCH	batch	IMS batch job.
CQS	cqs	Common Queue Server address space. CQS manages shared queues and may also manage resources on a resource structure.
DBRC	dbrc	DBRC address space.
IMS	ims	IMS region.
IMSCON	imscon	IMS connect. An address space that serves as an interface between IMS and a protocol that is not directly supported by IMS.
ODBM	odbm	Open Database Manager address space. ODBM provides distributed access to IMS database resources.
OM	om	Operations Manager address space. Operations Manager supports IMS operations in an IMSplex.
OTHER	other	Other non-IMS address space or job.
REPO	repo	Repository Server (RS) address space.
RM	rm	Resource Manager address space. Resource manager supports global resources in an IMSplex.
SCI	sci	Structured Call Interface address space.

QUERY IMSPLEX subtypes

The following table shows the possible IMSplex member subtypes. The table contains information about member types, the member subtypes associated with them, and the meaning of the member subtype. These are the only subtypes that are defined and used by members supplied by IMS.

Table 93. QUERY IMSPLEX member subtypes

Member type	Member subtype	Meaning
DBRC	DBRC group_id	DBRC instances that share the same RECON in an IMSplex environment.
IMS	DBDC	IMS DB/DC address space.
IMS	DBCTL	IMS DBCTL address space. DBCTL supports database functions.
IMS	DCCTL	IMS DCCTL address space. DCCTL supports data communications functions.
IMS	FDBR	IMS Fast Database Recovery. An IMS control region that recovers database resources when an IMS database manager fails.
RM	SNGLRM	RM is defined without a resource structure, so only a single RM is allowed in the IMSplex.

Table 93. QUERY IMSPLEX member subtypes (continued)

Member type	Member subtype	Meaning
RM	MULTRM	RM is defined with a resource structure, so multiple RMs are allowed in the IMSplex.

Output fields

The following table shows the output fields for the QUERY IMSPLEX command. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 94. Output fields for QUERY IMSPLEX command

Short label	Long label	Keyword	Meaning
IMSMBR	Member	N/A	IMSplex member name. The IMSplex member name is always returned.
IMSPLX	IMSplex	N/A	IMSplex name. The IMSplex name is always returned.
MBR	MbrName	N/A	IMSplex member that built the output line. The OM identifier of the OM that built the output line.
CC	CC	N/A	Completion code for the line of output. The completion code is always returned.
STT	Status	STATUS	IMSplex member status. For more information, see Table 91 on page 284.
JOB	JobName	JOB	Job name of IMSplex member.
OS	OSName	OS	Name of OS image on which the IMSplex member is executing.
STYP	Subtype	SUBTYPE	Subtype of IMSplex member. See Table 93 on page 285 for an explanation of the possible subtypes.
TYP	Type	TYPE	IMSplex member type.
VER	Version	VERSION	IMSplex member version.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY IMSPLEX command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 95. Return and reason codes for the QUERY IMSPLEX command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY IMSPLEX command completed successfully.
X'02000008'	X'00002048'	The QUERY IMSPLEX command has more than one filter value specified. Only one of the TYPE or STATUS filters can be specified.
X'02000008'	X'0000203C'	An invalid keyword parameter value was specified.
X'0200000C'	X'00003000'	The QUERY IMSPLEX command is successful for at least one resource name. The QUERY IMSPLEX command is not successful for one or more resource names. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY IMSPLEX command are listed in the QUERY IMSPLEX completion code table.
X'0200000C'	X'00003004'	No resources were found to be returned. The resource names specified might be invalid or there were no resources that match the filter specified.
X'0200000C'	X'00003008'	The QUERY IMSPLEX command was routed to multiple clients. None of the clients that processed the command returned a return code and reason code to the OM. At least one command client returned either command response data or a response message.
X'02000014'	X'00005020'	The QUERY IMSPLEX command processing terminated. OM was unable to obtain storage for a system AWE while processing the command.
X'02000014'	X'0000502C'	The QUERY IMSPLEX command processing terminated. OM was unable to obtain storage for the command output header.
X'02000014'	X'00005030'	The QUERY IMSPLEX command processing terminated. OM was unable to obtain storage for the command output response.
X'02000014'	X'00005040'	The QUERY IMSPLEX command processing terminated because of an SCI error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains completion codes can be returned on a QUERY IMSPLEX command.

Table 96. Completion codes for the QUERY IMSPLEX command

Completion code	Meaning
0	The QUERY IMSPLEX command completed successfully for the resource.
4	The IMSplex name is unknown to the client that is processing the request. The IMSplex name might have been typed in error or the IMSplex might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

The following are examples of the QUERY IMSPLEX command:

Example 1 for QUERY IMSPLEX command

TSO SPOC input:
```
QRY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)
```

TSO SPOC output:
```
Response for: QUERY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS)
IMSplex  MbrName   CC Member    JobName   Type Subtype Status
CSLPLEX1 OM1OM      0 IMS2      IMS2      IMS  DBDC    READY,ACTIVE
CSLPLEX1 OM1OM      0 CQS1CQS   CQSRE1    CQS          ACTIVE
CSLPLEX1 OM1OM      0 PRAQJOB4  PRAQJOB4  DBRC 001     READY,ACTIVE
CSLPLEX1 OM1OM      0 DBRACSAH  DBRACSAH  DBRC 001     READY,ACTIVE
CSLPLEX1 OM1OM      0 SYS3      IMS1      IMS  DBDC    READY,ACTIVE
CSLPLEX1 OM1OM      0 OM1OM     OM1       OM           READY,ACTIVE
CSLPLEX1 OM1OM      0 IMS3      IMS3      IMS  DBDC    READY,ACTIVE
CSLPLEX1 OM1OM      0 PRAQJOB2  PRAQJOB2  DBRC 001     READY,ACTIVE
CSLPLEX1 OM1OM      0 USRT011   USRT011   AOP          ACTIVE
CSLPLEX1 OM1OM      0 RM1RM     RM1       RM   MULTRM  READY,ACTIVE
CSLPLEX1 OM1OM      0 SCI1SC    SCI1      SCI          READY,ACTIVE
CSLPLEX1 OM1OM      0 ODBM1OD   ODBM1     ODBM         READY,ACTIVE
```

OM API input:
```
CMD(QRY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2002.163 15:05:18.859217</statime>
<stotime>2002.163 15:05:18.860443</stotime>
<staseq>B7C4A41E663D11C3</staseq>
<stoseq>B7C4A41E6689B9C3</stoseq>
<rqsttkn1>USRT011 10080518</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>IMSPLEX        </kwd>
<input>QUERY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left"  />
<hdr slbl="MBR"    llbl="MbrName" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left"  />
<hdr slbl="CC"     llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="IMSMBR" llbl="Member"  scope="LCL" sort="N" key="0" scroll="NO"  len="8"
dtype="CHAR" align="left"  />
<hdr slbl="JOB"    llbl="JobName" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left"  />
<hdr slbl="TYP"    llbl="Type"    scope="LCL" sort="N" key="0" scroll="YES" len="5"
dtype="CHAR" align="left"  />
<hdr slbl="STYP"   llbl="Subtype" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left"  />
<hdr slbl="STT"    llbl="Status"  scope="GBL" sort="N" key="0" scroll="YES" len="*"
dtype="CHAR" align="left"  />
```

```
</cmdrsphdr>
<cmdrspdata>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(IMS2)     CC(  0) JOB(IMS2)     TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(CQS1CQS)   CC(  0) JOB(CQSRE1)   TYP(CQS)
STYP( ) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(SYS3)      CC(  0) JOB(IMS1)     TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(PRAQJOB4)  CC(  0) JOB(PRAQJOB4) TYP(DBRC)
STYP(001) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(DBRACSAH)  CC(  0) JOB(DBRACSAH) TYP(DBRC)
STYP(001) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(OM1OM)     CC(  0) JOB(OM1)      TYP(OM)
STYP( ) STT(EADY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(IMS3)      CC(  0) JOB(IMS3)     TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(PRAQJOB2)  CC(  0) JOB(PRAQJOB2) TYP(DBRC)
STYP(001) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(USRT011)   CC(  0) JOB(USRT011)  TYP(AOP)
STYP( ) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(RM1RM)     CC(  0) JOB(RM1)      TYP(RM)
STYP(MULTRM) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(SCI1SC)    CC(  0) JOB(SCI1)     TYP(SCI)
STYP( ) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM)  IMSMBR(ODBM1OD)   CC(  0) JOB(ODBM1)    TYP(ODBM)
STYP( ) STT(READY,ACTIVE)</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY IMSPLEX command displays the IMSplex members that
compose IMSplex CSLPLEX1. This IMSplex contains three IMS systems (IMS1,
IMS2, and IMS3), a TSO SPOC (USRT011), a CQS (CQSRE1), RM (RM1), and OM
(OM1). OM1 is the command master that built the output.

Example 2 for QUERY IMSPLEX command

TSO SPOC input:
```
QRY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)
```

TSO SPOC output:
```
 Response for: QUERY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)
IMSplex  MbrName   CC Member   JobName  Type  Subtype  Status
CSLPLEX1 OM1OM      0 IMS2     IMS2     IMS   DBDC     READY,ACTIVE
CSLPLEX1 OM1OM      0 CQS1CQS  CQSRE1   CQS            ACTIVE
CSLPLEX1 OM1OM      0 SYS3     IMS1     IMS   DBDC     READY,ACTIVE
CSLPLEX1 OM1OM      0 OM1OM    OM1      OM             READY,ACTIVE
CSLPLEX1 OM1OM      0 IMS3     IMS3     IMS   DBDC     READY,ACTIVE
CSLPLEX1 OM1OM      0 USRT011  USRT011  AOP            ACTIVE
CSLPLEX1 OM1OM      0 RM1RM    RM1      RM    MULTRM   READY,ACTIVE
CSLPLEX1 OM1OM      0 SCI1SC   SCI1     SCI            READY,ACTIVE
CSLPLEX1 OM1OM      0 ODBM1OD  ODBM1    ODBM           READY,ACTIVE
```

OM API input:
```
CMD (QRY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname><omvsn>1.1.0</omvsn>
<xmlvsn>1</xmlvsn>
<statime>2002.163 15:05:18.859217</statime>
<stotime>2002.163 15:05:18.860443</stotime>
<staseq>B7C4A41E663D11C3</staseq>
<stoseq>B7C4A41E6689B9C3</stoseq>
```

```
<rqsttkn1>USRT011 10080518</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>IMSPLEX          </kwd>
<input>QUERY IMSPLEX NAME(CSLPLEX1) SHOW(JOB,SUBTYPE,STATUS,TYPE)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="MBR"    llbl="MbrName" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"     llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="IMSMBR" llbl="Member"  scope="LCL" sort="N" key="0" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="JOB"    llbl="JobName" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="TYP"    llbl="Type"    scope="LCL" sort="N" key="0" scroll="YES" len="5"
dtype="CHAR" align="left" />
<hdr slbl="STYP"   llbl="Subtype" scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="STT"    llbl="Status"  scope="GBL" sort="N" key="0" scroll="YES" len="*"
dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(IMS2)    CC(  0) JOB(IMS2)    TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(CQS1CQS) CC(  0) JOB(CQSRE1) TYP(CQS)
STYP(   ) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(SYS3)    CC(  0) JOB(IMS1)    TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(OM1OM)   CC(  0) JOB(OM1)     TYP(OM)
STYP(   ) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(IMS3)    CC(  0) JOB(IMS3)    TYP(IMS)
STYP(DBDC) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(USRT011) CC(  0) JOB(USRT011) TYP(AOP)
STYP(   ) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(RM1RM)   CC(  0) JOB(RM1)     TYP(RM)
STYP(MULTRM) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(SCI1SC)  CC(  0) JOB(SCI1)    TYP(SCI)
STYP(   ) STT(READY,ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(ODBM1OD) CC(  0) JOB(ODBM1)   TYP(ODBM)
STYP(   ) STT(READY,ACTIVE)</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY IMSPLEX command displays the IMSplex members that
compose IMSplex CSLPLEX1. This IMSplex contains three IMS systems (IMS1,
IMS2, and IMS3), a TSO SPOC (USRT011), a CQS (CQSRE1), RM (RM1), and OM
(OM1). OM1 is the command master that built the output.

Example 3 for QUERY IMSPLEX command

TSO SPOC input:
```
QUERY IMSPLEX TYPE(REPO) SHOW(ALL)
```

TSO SPOC output:

```
IMSplex   MbrName    CC Member   JobName  Type  Subtype  Version  OSName   Status
CSLPLEX1  OM1OM       0 REPO2RP   REPO2    REPO           1.2.0    ECDVL40  ACTIVE
CSLPLEX1  OM1OM       0 REPO3RP   REPO3    REPO           1.2.0    EC01221  ACTIVE
CSLPLEX1  OM1OM       0 REPO1RP   REPO1    REPO           1.2.0    EC01589  READY,ACTIVE
```

OM API input:

```
CMD(QUERY IMSPLEX TYPE(REPO) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.186 22:29:24.146714</statime>
<stotime>2011.186 22:29:24.147544</stotime>
<staseq>C806A4CD0761AED2</staseq>
<stoseq>C806A4CD07958A92</stoseq>
<rqsttkn1>USRT005 10152924</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>IMSPLEX        </kwd>
<input>QUERY IMSPLEX TYPE(REPO) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="A" key="1"
 scroll="NO" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="N" key="0"
 scroll="YES" len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="N" key="0" scroll="YES"
 len="4" dtype="INT" align="right" />
<hdr slbl="IMSMBR" llbl="Member" scope="LCL" sort="N" key="0"
 scroll="NO" len="8" dtype="CHAR" align="left" />
<hdr slbl="JOB" llbl="JobName" scope="LCL" sort="N" key="0"
 scroll="YES" len="8" dtype="CHAR" align="left" />
<hdr slbl="TYP" llbl="Type" scope="LCL" sort="N" key="0" scroll="YES"
 len="5" dtype="CHAR" align="left" />
<hdr slbl="STYP" llbl="Subtype" scope="LCL" sort="N" key="0"
 scroll="YES" len="8" dtype="CHAR" align="left" />
<hdr slbl="VER" llbl="Version" scope="LCL" sort="N" key="0"
 scroll="YES" len="8" dtype="CHAR" align="left" />
<hdr slbl="OS" llbl="OSName" scope="LCL" sort="N" key="0" scroll="YES"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="N" key="0" scroll="YES"
 len="*" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(REPO2RP) CC(   0) JOB(REPO2)
 TYP(REPO) STYP() VER(1.2.0) OS(ECDVL40) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(REPO3RP) CC(   0) JOB(REPO3)
 TYP(REPO) STYP() VER(1.2.0) OS(EC01221) STT(ACTIVE)</rsp>
<rsp>IMSPLX(CSLPLEX1) MBR(OM1OM) IMSMBR(REPO1RP) CC(   0) JOB(REPO1)
 TYP(REPO) STYP() VER(1.2.0) OS(EC01589) STT(READY,ACTIVE)</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY IMSPLEX TYPE(REPO) SHOW(ALL) command returns
information about the Repository Server address spaces registered to SCI. The
REPO1RP member is the master server and the REPO2RP server is the subordinate
server because REPO2RP has not yet issued the SCI Ready request.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY LE command

Use the QUERY LE command to display Language Environment® (LE) runtime parameter overrides defined by a previous UPDATE LE command. The query can use filters on transaction code, LTERM, user ID, or program name.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 293
- "Usage notes" on page 294
- "Output fields" on page 294
- "Return, reason, and completion codes" on page 294
- "Examples" on page 295

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY LE command and keywords can be issued.

Table 97. Valid environments for the QUERY LE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY LE	X	X	X
LTERM	X	X	X
PGM	X	X	X
SHOW	X	X	X
TRAN	X	X	X
USERID	X	X	X

Syntax

```
►►─┬─QUERY─┬─LE──────────────────────────────────────────────────────►
   └─QRY───┘   └─TRAN(tranname)─┘  └─LTERM(ltermname)─┘
```

```
                                          ┌─── , ───────────┐
                                          ▼                 │
►──┬──────────────────┬──┬──────────────────────┬──SHOW(──┬──ALL────────┬──)──►◄
   └─USERID(userid)─┘  └─PGM(programname)─┘              ├─LERUNOPTS─┤
                                                          ├─LTERM─────┤
                                                          ├─PGM───────┤
                                                          ├─TRAN──────┤
                                                          └─USERID────┘
```
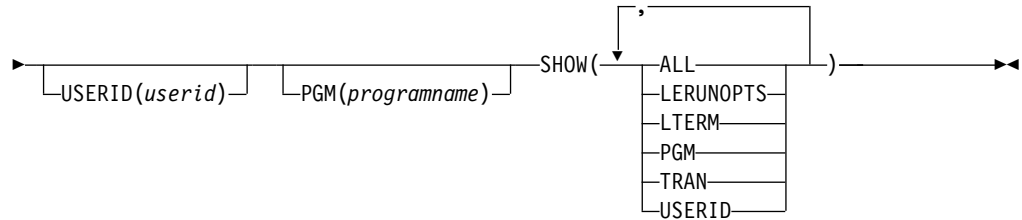
Keywords

The following keywords are valid for the QUERY LE command:

The parameters support a generic or wildcard parameter. A generic parameter is a
1-8 character name that includes an asterisk or a percent sign. An asterisk can be
replaced by a zero or more characters to create a valid resource name. A percent
sign can be replaced by exactly one character to create a valid resource name.

LTERM()

Specifies the 1-8 character name of the LTERM or LTERMs matching the
generic or wildcard parameter.

PGM()

Specifies the 1-8 character name of the program or programs matching the
generic or wildcard parameter.

SHOW()

Specifies the output fields to be returned. At least one SHOW field is required
on the command.

ALL

Returns all the output fields. This is the same as if the following was
specified: SHOW(TRAN,LTERM,USERID,PGM,LERUNOPTS).

LERUNOPTS

Returns all of the LE override parameters associated with the transaction,
LTERM, user ID, or program name.

LTERM

Returns the logical terminal name field.

PGM

Returns the program name field.

TRAN

Returns the transaction name field.

USERID

Returns the user identifier field.

TRAN()

Specifies the 1-8 character name of the transaction or transactions matching the
generic or wildcard parameter. If the TRAN, LTERM, USERID, or PGM
resource filters are not specified, all parameter overrides are returned.

USERID()

Specifies the 1-8 character name of the userid or userids matching the generic
or wildcard parameter. If the TRAN, LTERM, USERID, or PGM resource filters
are not specified, all parameter overrides are returned.

Usage notes

Any combination of parameters can be used to qualify the application instance. All entries found that match the criteria are returned. Specify on the command which output fields should be returned in the command response. You can ask for all information that includes transaction code, LTERM name, user ID, program name, and runtime parameters.

This command can be specified only through the Operations Manager API.

The command syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

The following table shows the QUERY LE output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 98. Output fields for QUERY LE command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code for the line of output. Completion code is always returned.
LTRM	Lterm	LTERM	LTERM Name requested by the QUERY.
MBR	MbrName	N/A	IMSplex member (IMS identifier) that built the output line. Member name is always returned.
PGM	Program	PGM	Program Name requested by the QUERY.
PRM	LERunOpts	LERUNOPTS	The LE override parameters for the specified resource filters.
TRAN	Trancode	TRAN	Transaction Name requested by the QUERY.
UID	Userid	USERID	Userid requested by the QUERY.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY LE command. The OM return and reason codes that may be returned as a result of the QUERY LE command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 99. Return and reason codes for the QUERY LE command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY LE command completed successfully.
X'00000008'	X'0000200C'	No resources found to return. Either the entry was previously deleted or a keyword filter was typed incorrectly.
X'00000008'	X'00002014'	An invalid character was specified in the filter name.
X'00000010'	X'00004040'	The parameter override header has not been initialized. Retry the command after restart is complete.
X'00000014'	X'00005000'	Unable to get storage from IMODULE GETSTOR.
X'00000014'	X'00005010'	Unable to obtain latch.
X'00000014'	X'00005FFF'	Internal IMS Error - Should not occur.

The following table includes an explanation of the completion code.

Table 100. Completion code for the QUERY LE command

Completion code	Meaning
0	The QUERY LE command completed successfully for the specified resource.

Examples

The following are examples of the QUERY LE command:

Example 1 for QUERY LE command

Assume the following filters are specified on QRY LE commands:
1. TRAN(PART) SHOW(ALL) Returns entries #1, 2, 3, 5, 6, 8.
2. TRAN(PART) LTERM(TERM1) SHOW(ALL) Returns entries #3, 5, 6.
3. LTERM(TERM2) USERID(BETTY) SHOW(ALL) Returns entry #7.
4. TRAN(PART) LTERM(TERM1) USERID(BETTY) SHOW(ALL) Does not return any entries.
5. TRAN(PART) LTERM(TERM*) SHOW(ALL) Returns entries #3, 5, 6, 8.
6. USERID(B*) SHOW(ALL) Returns entries #2, 5, 6, 7.

Rules for matching an entry which results in it being returned on QUERY command:
- If a filter is specified on the command for a particular resource it must match the resource filter defined in the entry. The resource in the QUERY LE command may be specified with wildcards as defined previously.
- A resource filter that is not specified on a QUERY LE command will match on any filter for the specific resource defined in the entry. A non-specified filter is treated as a wildcard. For instance if the LTERM filter is not specified on a QRY LE command it will match on any LTERM resource defined in an entry, as if LTERM(*) was specified on the command.

The following table is a logical representation of the parameter override table entries prior to any of the above query commands being processed.

Table 101. Parameter override table entries for example 1

Entry#	TRAN	LTERM	USERID	PROGRAM	LERUNOPTS
1	PART			DFSSAM02	aaaa
2	PART		BETTY		bbbb
3	PART	TERM1			cccc
4				DFSSAM02	dddd
5	PART	TERM1	BARBARA		eeee
6	PART	TERM1	BOB		ffff
7		TERM2	BETTY		gggg
8	PART	TERM2			iiii

Example 2 for QUERY LE command

TSO SPOC input:

```
QRY LE SHOW(ALL)
```

TSO SPOC output:

```
SYS3 0                         IAPMDI29 CCCC
SYS3 0 IAPMDI26     USRT001      RPTOPTS=((ON),NOOVR),RPTSTG=(((OFF),NOOVR)
SYS3 0 IAPMDI27 IMS1 USRT001  IAPMDI27 AAAA
```

OM API input:

```
CMD(QRY LE SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 17:34:01.196902</statime>
<stotime>2002.163 17:34:01.197368</stotime>
<staseq>B7C4C55B67566505</staseq>
<stoseq>B7C4C55B67738365</stoseq>
<rqsttkn1>USRT002 10103401</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>LE           </kwd>
<input>QRY LE SHOW(ALL)        </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"  llbl="MbrName"   scope="LCL" sort="A" key="1" scroll="NO" len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"   llbl="CC"        scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT" align="right" />
<hdr slbl="TRAN" llbl="Trancode"  scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="LTRM" llbl="Lterm"     scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
<hdr slbl="UID"  llbl="Userid"    scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
<hdr slbl="PGM"  llbl="Program"   scope="LCL" sort="N" key="0" scroll="YES" len="8"
```

```
dtype="CHAR" align="left" />
<hdr slbl="PRM"  llbl="LERunOpts" scope="LCL" sort="N" key="0" scroll="YES" len="*"
dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3    ) CC(   0) TRAN(         ) LTRM(     ) UID(         ) PGM(IAPMDI29)
PRM(CCCC    ) </rsp>
<rsp>MBR(SYS3    ) CC(   0) TRAN(IAPMDI26) LTRM(     ) UID(USRT001 ) PGM(        )
PRM(RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR)          )</rsp>
<rsp>MBR(SYS3    ) CC(   0) TRAN(IAPMDI27) LTRM(IMS1 ) UID(USRT001 ) PGM(IAPMDI27)
PRM(AAAA    ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The SHOW(ALL) parameter is specified, so all four filters and the runtime option string are shown for each table entry. Furthermore, no filters are specified in the command, so all table entries are shown. In this example, there are three table entries. The first specifies one filter (program) and the parameter string for this entry is CCCC. The second entry specifies two filters, trancode and user ID, and its parameter string is RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR). The last entry specifies all four filters and a parameter string of AAAA.

Example 3 for QUERY LE command

TSO SPOC input:

```
QRY LE SHOW(LTERM,USERID)
```

TSO SPOC output:

```
MbrName    CC Lterm    Userid
SYS3        0
SYS3        0          USRT001
SYS3        0 IMS1     USRT001
```

OM API input:

```
CMD(QRY LE SHOW(LTERM,USERID))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 17:36:27.588393</statime>
<stotime>2002.163 17:36:27.589261</stotime>
<staseq>B7C4C5E703729D6F</staseq>
<stoseq>B7C4C5E703A8D467</stoseq>
<rqsttkn1>USRT002 10103627</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>LE           </kwd>
<input>QRY LE SHOW(LTERM,USERID)         </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"  llbl="MbrName" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"   llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
```

```
                dtype="INT" align="right" />
                <hdr slbl="LTRM" llbl="Lterm"   scope="LCL" sort="N" key="0" scroll="YES" len="8"
                dtype="CHAR " align="left" />
                <hdr slbl="UID"  llbl="Userid"  scope="LCL" sort="N" key="0" scroll="YES" len="8"
                dtype="CHAR " align="left" />
                </cmdrsphdr>
                <cmdrspdata>
                <rsp>MBR(SYS3     ) CC(   0) LTRM(          ) UID(         ) </rsp>
                <rsp>MBR(SYS3     ) CC(   0) LTRM(          ) UID(USRT001 ) </rsp>
                <rsp>MBR(SYS3     ) CC(   0) LTRM(IMS1      ) UID(USRT001 ) </rsp>
                </cmdrspdata>
                </imsout>
```

Explanation: This command uses the SHOW parameter to limit the amount of data
that is shown for each entry in the table. All three table entries are shown, but only
the LTERM and TRAN filters are shown for each one. The first entry has neither
an LTERM filter nor a USERID filter defined, so it is blank except for the
MbrName and CC.

Example 4 for QUERY LE command

TSO SPOC input:
```
QRY LE USERID(USRT*) SHOW(LTERM,USERID)
```

TSO SPOC output:
```
MbrName    CC Lterm    Userid
SYS3        0          USRT001
SYS3        0 IMS1     USRT001
```

OM API input:
```
CMD(QRY LE USERID(USRT*) SHOW(LTERM,USERID))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 17:50:24.925819</statime>
<stotime>2002.163 17:50:24.926381</stotime>
<staseq>B7C4C9058F87B484</staseq>
<stoseq>B7C4C9058FAAD324</stoseq>
<rqsttkn1>USRT002 10105024</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>LE    </kwd>
<input>QRY LE USERID(USRT*) SHOW(LTERM,USERID)  </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"  llbl="MbrName" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"   llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT" align="right" />
<hdr slbl="LTRM" llbl="Lterm"   scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
<hdr slbl="UID"  llbl="Userid"  scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR " align="left" />
```

```
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3   ) CC(  0) LTRM(        ) UID(USRT001 ) </rsp>
<rsp>MBR(SYS3   ) CC(  0) LTRM(IMS1    ) UID(USRT001 ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: This command uses both the USERID filter and the SHOW parameter to limit the amount of data shown in the output. The USERID(USRT*) parameter limits the output to only those table entries that define a USERID filter that fits the specified pattern (USRT*). The SHOW parameter limits how much information is returned about each table entry In this case, only the LTERM and USERID are shown.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

"UPDATE LE command" on page 1194

QUERY LTERM command

Use the QUERY LTERM command to display information about logical terminals (LTERMs). This command can be specified only through the OM API and is valid on an XRF alternate.

Subsections:
- "Environment"
- "Syntax" on page 300
- "Keywords" on page 301
- "Usage notes" on page 308
- "Equivalent IMS type-1 commands" on page 309
- "Output fields" on page 310
- "QUERY LTERM status" on page 313
- "Return, reason, and completion codes" on page 314
- "Examples" on page 316

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 102. Valid environments for the QUERY LTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY LTERM	X		X
NAME	X		X
MSGAGE	X		X
QCNT	X		X
OPTION	X		X
SHOW	X		X

Table 102. Valid environments for the QUERY LTERM command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
STATUS	X		X

Syntax

```
►►─┬─QUERY─┬─LTERM─┬─NAME(*)──────────────────────────────────┬─►
   └─QRY───┘       │              ┌─,────────────┐            │
                   │              ▼              │            │
                   └─NAME(─┬─ltermname──┬─)──────┘
                          └─ltermname*─┘
```

```
►─┬──────────────────────────────────────────────────┬─►
  │  ┌─,──────────────────┐                           │
  │  ▼                    │  ┌─OPTION(MSGQ)─┐          │
  ├─MSGAGE(number)────────┤  ├──────────────┤          │
  └─QCNT(─┬─LT─┬─,number)─┘  └─OPTION(EMHQ)─┘
          ├─LE─┤
          ├─GT─┤
          ├─GE─┤
          ├─EQ─┤
          └─NE─┘
```

```
►─┬─SHOW(GLOBAL,LOCAL)──────────────────┬─┬──────────────────────────────┬─►◄
  │                                     │ │         ┌─,─────────┐         │
  │        ┌─GLOBAL,LOCAL,─┐            │ │         ▼           │         │
  ├─SHOW(──┴───────────────┴─┤ A ├─)────┤ └─STATUS(─┬─LOCK─────┬─)────────┘
  │      ┌─,──────────┐                 │           ├─MTO──────┤
  │      ▼            │                 │           ├─QERR─────┤
  ├─SHOW(─┬─GLOBAL─┬──┴─┬──────────┬─)──┤           ├─QLOCK────┤
  │       └─LOCAL──┘    └─,─┤ A ├──┘    │           ├─REMOTE───┤
  └─SHOW(EXPORTNEEDED)──────────────────┘           ├─RM───────┤
                                                    ├─RMACTIVE─┤
                                                    ├─RMOWNED──┤
                                                    ├─SMTO─────┤
                                                    ├─STATIC───┤
                                                    ├─STOQ─────┤
                                                    └─STOSEND──┘
```

A:

```
                        ┌─,──────────────────┐
    │    ┌─▼──ALL──────────────┐
├────┴──────COMPONENT───────┴──────────────────────────────────────────┤
           ├─DEFNTYPE────────┤
           ├─DEFN────────────┤
           ├─EMHQ────────────┤
           ├─IMSID───────────┤
           ├─MSGAGE──────────┤
           ├─MSNAME──────────┤
           ├─NODE────────────┤
           ├─OWNER───────────┤
           ├─QCNT────────────┤
           ├─STATUS──────────┤
           ├─TIMESTAMP───────┤
           ├─USER────────────┤
           ├─VERSION─────────┤
           └─EXPORTNEEDED────┘
```
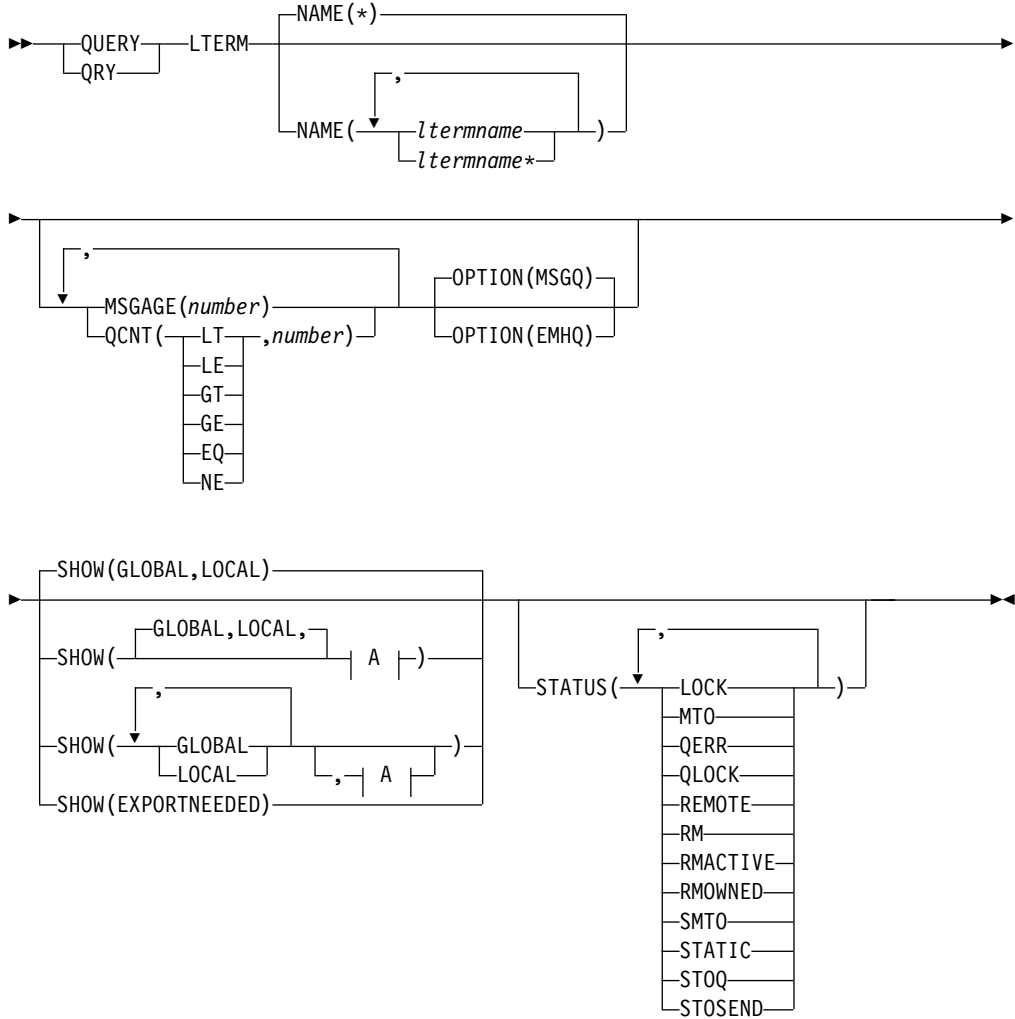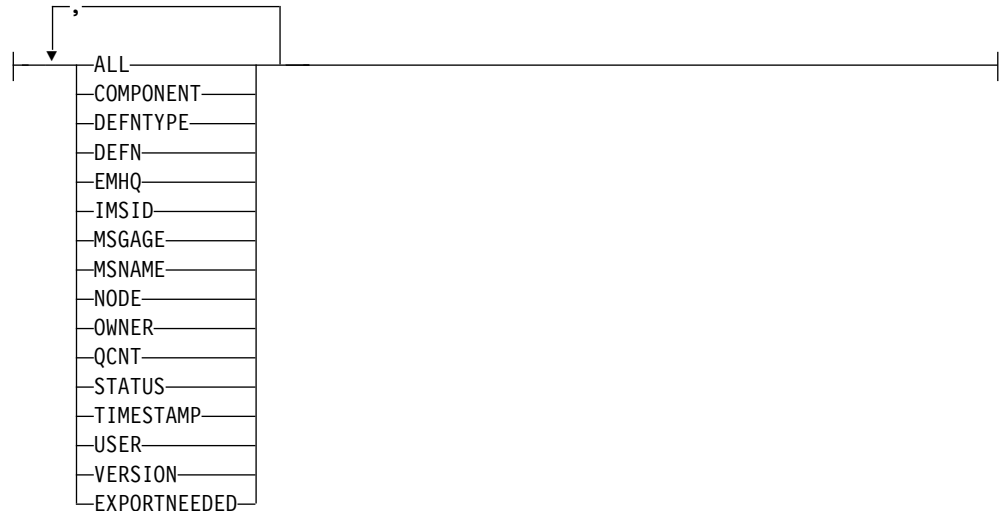
Keywords

The following keywords are valid for the QUERY LTERM command:

MSGAGE()

Selects only those logical terminals that have at least one message whose message age is greater than the message age value specified. This applies only to messages in the shared message queues. The value specified is the number of days. Valid values are 0-365.

The information displayed includes the total count of messages on the queue, the count of messages with a message age greater than or equal to the message age specified, the time stamp of the oldest message, and the time stamp of the newest message.

When the MSGAGE filter is specified, the output returned includes the queue count of the LTERM even if SHOW(QCNT) is not specified, and the message age information even if SHOW(MSGAGE) is not specified.

MSGAGE( ) applies only to a shared-queues environment, and is processed only by the command master IMS. It is valid only when SHOW(GLOBAL) is specified.

The OPTION keyword tells IMS whether the message queue structure (OPTION(MSGQ)) or the EMH queue structure (OPTION(EMHQ)) is to be read. OPTION(MSGQ) is the default. OPTION(EMHQ) is valid only if shared EMH is used.

The performance implication is that in order to obtain message age information, all of the shared-queues LTERM messages on the Coupling Facility must be read.

If MSGAGE( ) is specified with the QCNT( ) or STATUS( ) filters, then IMS selects those LTERMs that match any of the specified filters (they do not need to match all specified filters).

NAME()

Specifies the names of one or more logical terminals that are to be displayed. Valid names are 1-8 characters, and wildcards can be specified. To display all logical terminals, specify NAME(*). NAME(*) is the default.

There are some special considerations when specifying certain specific LTERM names.

The primary and secondary MTO LTERM names may have generic names associated with them, as specified by the PMTOG and SMTOG keywords in the DFSDCxxx PROCLIB member. To display the primary or secondary MTO LTERMs on all systems, the operator should specify the generic LTERM name. When this is done, the output displayed will show the real LTERM names on each system, not the generic name.

The system console LTERM has a generic name of 'WTOR'. When shared queues are not enabled, 'WTOR' is also the real name of the system console LTERM. When shared queues are enabled in a non-XRF system, the real name is the IMS system ID (IMSID). When shared queues are enabled in an XRF system, the real name is the Recoverable Service Element (RSENAME). To display the system console LTERMs on all systems, the operator should specify the generic LTERM name 'WTOR'. When this is done, the output displayed will show the real LTERM names on each system, not the generic name.

Logical terminals associated with VTAM nodes might exist either locally, in the resource structure, or both. Logical terminals associated with the system console, SPOOL or SYSOUT devices will exist only in local systems.

OPTION()
Specifies additional information to process the command. This keyword is valid only with the MSGAGE() or QCNT() filter.

MSGQ
Specifies that queue count information should be retrieved from the shared message queue structure, or the local message queues if shared queues are not enabled (for QCNT filter only).

EMHQ
Specifies that queue count information should be retrieved from the expedited message handler queue structure. EMHQ is valid only when EMH queues are used.

QCNT()
Selects only those logical terminals that have a queue count less than (LT), less than or equal to (LE), greater than (GT), greater than or equal to (GE), equal to (EQ), or not equal to (NE) the specified number. The specified number cannot be a 1 when LT is specified, and cannot be a 0 when EQ, GE, LE, or LT is specified. Regardless of the condition specified, only LTERMs with a queue count greater than 0 are returned when the QCNT filter is specified. Lterms with a queue count of 0 are not returned.

The QCNT filter is valid in both a shared-queues environment and in a non-shared-queues environment.

In a shared-queues environment, the global queue count values are used to determine the LTERMs to be displayed. Therefore, SHOW(LOCAL) is invalid with the QCNT filter. In this environment, only the command master processes the command. All other IMS systems ignore the command. The OPTION keyword tells IMS whether the message queue structure (OPTION(MSGQ)) or the EMH queue structure (OPTION(EMHQ)) is to be read. OPTION(MSGQ) is the default. OPTION(EMHQ) is only valid if shared EMH is used. If QCNT is specified with a wildcard LTERM name, the performance implication is that all of the shared-queues LTERM messages on the Coupling Facility must be read.

In a non-shared-queues environment, the local queue count values are used to determine the LTERMs to be displayed. In this environment, the command is

processed by each IMS that the command is routed to because the queues are local. Each IMS returns all of the LTERMs that it finds locally that match the queue count filter specified. Because there is no local EMH queue count, OPTION(EMHQ) is invalid in this environment, and if specified, the command will be rejected.

When the QCNT filter is specified, the output returned includes the queue count of the LTERM, even if SHOW(QCNT) is not specified.

If QCNT() is specified with the MSGAGE() or STATUS() filters, then IMS selects those LTERMs that match any of the specified filters (they do not need to match all specified filters).

SHOW()

This specifies the LTERM output fields to be returned. If SHOW is not specified, and if none of the three keywords MSGAGE, QCNT, or STATUS is specified, then only the LTERM names are returned. This provides a method for a system management application to obtain a list of all LTERM names that are currently known in the IMSplex.

Two parameters, GLOBAL and LOCAL, are used to specify the location (global resources or local resources) where IMS should obtain the information that is to be displayed. The default is both GLOBAL and LOCAL.

The rest of the parameters are used to specify what information is displayed.

The parameters supported with the SHOW keyword, which can be specified in any order, are:

ALL
Returns all of the output fields.

COMPONENT
Displays the input and output component for the LTERM. The component numbers are defined in the NAME macro, user descriptor, signon user exit (DFSSGNX0), or the /ASSIGN LTERM command, and represent the component of a terminal that is used for input or output.

DEFNTYPE
Displays the definition type that the resource was defined with. This parameter is applicable only to the MSC remote LTERM.

DEFN
Specifies that the resource definitions are to be returned.

The following attributes for MSC remote LTERMs can be returned:
- MSNAME
- The repository create and update time stamps
- The IMS runtime create and update time stamps

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition that is obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS

systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE or STATUS. The LclStat and LDefnType columns, which are returned on the QRY LTERM SHOW(ALL) command, are not returned with SHOW(DEFN).
- SHOW(DEFN) cannot be specified with filter STATUS.
- SHOW(DEFN) cannot be specified with global shared queues, global status, or sysplex terminal management keywords, including EMHQ, MSGAGE, OWNER, QCNT, and VERSION.
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).

Resource definitions that are stored in the repository are used to determine the response lines with the repository information, and the runtime resource definitions are used to determine the response lines with the IMS runtime resource information.

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

EXPORTNEEDED
Specifies that only the MSC remote LTERM resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command. SHOW(EXPORTNEEDED) behaves like a filter by only displaying resources that need to be exported, as if QUERY LTERM EXPORTNEEDED(Y) were supported.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the MSC remote LTERMs that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each MSC remote LTERM name specified on the command. If the specified MSC remote LTERM does not need to be exported to the repository, an N is returned in the ExportNeeded column.

If NAME is specified with specific or generic parameters that only match local LTERMs, they will be rejected with completion code 10 "NO RESOURCES FOUND", because automatic export does not apply to local LTERMs.

Restrictions:
- SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords.
- SHOW(EXPORTNEEDED) is valid only when IMS is enabled with dynamic resource definition for MSC resources (MSCRSCS=DYN) and the IMSRSC repository.
- The QUERY LTERM SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources that have not been exported to the IMSRSC repository, you can issue the /CHECKPOINT command to automatically write the resources to the repository so that they are available for the next IMS cold start.

EMHQ

Displays the LTERM message queue count in the Expedited Message Handler (EMH) queues.

EMHQ is valid only when the GLOBAL parameter is specified on the SHOW keyword. If GLOBAL is not specified, then the EMHQ parameter is ignored.

EMHQ is processed only by the command master. It is ignored by all other IMS systems.

EMHQ is valid only if shared EMH is used in a shared queues environment. Otherwise, this parameter is ignored.

GLOBAL

The command master displays global information, depending on the other SHOW parameters specified. This includes information from shared queues, the resource structure, and the repository.

The GLOBAL parameter is processed by the command master only. All other IMS systems ignore this parameter. If LOCAL is not also specified, then all IMS systems other than the command master ignore the command.

GLOBAL is applicable only when the command master is using shared queues, sysplex terminal management, the IMSRSC repository, or a combination.

GLOBAL is not applicable when the command master is not using shared queues, sysplex terminal management, or the IMSRSC repository. In this environment, if LOCAL is also specified, then GLOBAL is ignored. Otherwise, the command master rejects the command.

If repository is enabled, the command master makes requests to RM to determine the repository attribute definitions.

SHOW(DEFN,GLOBAL,LOCAL) is equivalent to SHOW(DEFN).

If shared queues are enabled, and global queue counts are requested, then the command master will make requests to CQS to determine the appropriate queue counts. This includes both MSGQ and EMHQ.

If sysplex terminal management is enabled, then the command master will make requests to RM to determine the appropriate global status.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

IMSID

Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only when the command master IMS is enabled to use the repository.

IMSID cannot be specified with any other SHOW keywords other than DEFN or DEFN and GLOBAL.

When SHOW(DEFN,IMSID) is specified, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN keyword, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

LOCAL

All IMS systems including the command master display local information, depending on the other SHOW parameters specified. This includes information local to the IMS processing the command.

The LOCAL parameter is processed by all IMS systems, including the command master.

LOCAL is applicable in any environment, regardless of whether shared queues, sysplex terminal management, or the IMSRSC repository are used.

SHOW(DEFN,GLOBAL,LOCAL) is equivalent to SHOW(DEFN).

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

MSGAGE

Displays the count of messages whose ages are greater than the value specified by the MSGAGE() filter. If the MSGAGE() filter is not specified, then the value is assumed to be 0, and the count displayed is the total message queue count for the LTERM. The time stamp of the oldest and newest message on the LTERM queue is also displayed.

SHOW(QCNT) and SHOW(EMHQ) determine whether IMS reads the shared message queues, or the EMH queues. If neither is specified, then SHOW(QCNT) is assumed.

The performance implication is that in order to obtain message age information, all of the shared-queues LTERM messages on the Coupling Facility must be read.

The MSGAGE parameter is valid only in a shared-queues environment, and is ignored in all other environments. It is processed by the command master only. All other IMS systems ignore this parameter.

MSNAME

Displays the MSC logical link path name (msname) associated with the remote LTERM, and the remote and local system identifiers (SYSIDs). This applies only to LTERMs that are defined as MSC remote LTERMs.

NODE

Displays the node, if any, associated with the LTERM. For VTAM terminals, this is the terminal name. For LTERMs associated with non-VTAM devices (system console, SPOOL, SYSOUT, and TCO), IMS displays a node name of DFSLN*xxx*, where *xxx* is the line number, and the line and PTERM number of the non-VTAM device.

OWNER

Displays the owner of the associated user or node resource in the resource structure. This applies only when sysplex terminal management is enabled, and is only processed by the command master. All other IMS systems ignore this parameter.

The owner is the IMSID (or RSENAME for XRF systems) of the IMS system that owns the associated user or node. An IMS system owns a user or node resource if the resource is active (the user is signed on, or the node is logged on), or an IMS system is maintaining significant status for that resource.

QCNT

Displays the LTERM message queue count.

The local queue counts value returned on this command represents the messages being processed by the IMS system where this command is issued. In a shared-queues environment, do not use the local queue counts for queue evaluation purposes. The global queue counts, which are returned on this command, represent the current message queue counts on the shared queues structure.

If the LOCAL parameter is also specified on the SHOW keyword, then all IMS systems that process the command, including the command master, display the local queue count. This is valid whether or not shared message queues are enabled.

If the GLOBAL parameter is also specified on the SHOW keyword, and shared message queues are enabled, then the command master displays the global queue count on the shared message queues (MSGQ).

The local and global queue counts are displayed as separate output fields.

STATUS

Returns local and global status of the LTERM. See "QUERY LTERM status" on page 313 for a list and meaning of possible status that can be returned.

TIMESTAMP

Displays the creation time (TimeCreate), last update time (TimeUpdate), last access time (TimeAccess), and last import time (TimeImport) timestamps are returned. This parameter is applicable only to the MSC remote LTERM. The time is returned in local time in the format YYYY.JJJ HH:MM:SS:TH:

YYYY

The year

JJJ

The Julian day (001 - 365)

The hour (01 - 24)

The minute (00 - 59)

The seconds (00 - 59)

The tenths and hundredths of a second (00 - 99)

USER
Displays the dynamic or ISC user, if any, associated with the LTERM.

VERSION
Displays the RM version number of the LTERM resource. This is the version number assigned to the LTERM, which is assigned by MVS, and maintained by RM, when the resource is created or updated in the resource structure. VERSION applies only when sysplex terminal management is enabled. VERSION is ignored when sysplex terminal management is not enabled.

STATUS()
Selects LTERMs for display that possess at least one of the specified LTERM statuses. The status might exist locally or globally if sysplex terminal management (STM) is enabled.

The STATUS filter is valid in both a sysplex terminal management environment and in a non sysplex terminal management environment.

In a sysplex terminal management environment, the status selected might exist locally, globally, or both. If sysplex terminal management is not enabled, then the status only exists locally.

If SHOW(LOCAL) is specified, then IMS will select only those LTERMs with the appropriate status in the local system. The command is processed by all IMS systems, including the command master.

If SHOW(GLOBAL) is specified, and sysplex terminal management is enabled, then IMS will select only those LTERMs with the appropriate status in the resource structure. The command is processed only by the command master.

If SHOW(GLOBAL) is specified, but sysplex terminal management is not enabled, then the command is rejected.

If SHOW(GLOBAL,LOCAL) is specified, which is the default, then IMS will select those LTERMs with the appropriate status either locally or in the resource structure (if sysplex terminal management is enabled). The command is processed by all IMS systems. The command master processes both global and local information.

The output returned when the status filter is specified includes the status of the LTERM, even if SHOW(STATUS) is not specified.

If STATUS() is specified with the MSGAGE() or QCNT() filters, then IMS selects those LTERMs that match any of the specified filters (they do not need to match all specified filters).

See "QUERY LTERM status" on page 313 to determine which filters can be used to select nodes with corresponding status.

Usage notes

The QUERY LTERM command can be specified only through the OM API.

QUERY LTERM can be issued on an XRF alternate system, but SHOW(GLOBAL) is not supported. Only local information can be displayed.

The processing of the QUERY LTERM command is different depending on whether IMS sysplex terminal management is enabled.

- If IMS sysplex terminal management is not enabled, processing is local for each system. The results of type-1 and type-2 commands are similar.
- If IMS sysplex terminal management is enabled, type-1 and type-2 command processing is similar when displaying local information. However, they differ in how global information is displayed.
- For type-1 /DISPLAY commands with IMS sysplex terminal management enabled, the command master displays information from either the resource structure or the local system, but not both. If the resource being displayed is not owned by any system or is owned by the command master, the command master displays the global resource. However, if the resource is owned by a system other than the command master, the command master displays only the local resource, and the owning system is responsible for displaying the global resource.
- For type-2 QUERY commands with IMS sysplex terminal management enabled, the command master is the only system that displays global resource information, regardless of whether the resource is owned. In addition, the command master displays local resource information. All other IMS systems that process the command display local resource information only. This approach allows more flexibility in displaying all information in an IMSplex.

The SHOW keyword determines which IMS systems process the command, and what information is displayed.

- If SHOW(GLOBAL) is specified, then the command master displays global information, which can include the global queue count if shared queues are enabled, and status from the resource structure if sysplex terminal management is enabled (STM=YES defined in DFSDCxxx PROCLIB member). This is true whether or not the LTERM is active on any particular IMS system. All other IMS systems ignore the GLOBAL parameter with return code X'00000004' and reason code X'00001000'.
- If SHOW(LOCAL) is specified, then each IMS system to which OM routes the command (including the command master) processes the command, and displays information that is local to each system.
- If both GLOBAL and LOCAL are specified, which is the default, then both global and local information are displayed. Each IMS system to which OM routes the command, including the command master, processes the command, and displays local information. In addition to local information, the command master displays global information.

To display information about resource definitions, specify SHOW(DEFN). To display both the IMS systems that have the resource defined and the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). To display the IMS systems that have the resource defined, specify SHOW(IMSID).

Equivalent IMS type-1 commands

The following table shows variations of the QUERY LTERM command and the type-1 IMS commands that perform similar functions.

Table 103. Type-1 equivalents for the QUERY LTERM command

QUERY LTERM command	Similar IMS type-1 command
QUERY LTERM SHOW(COMPONENT)	/DISPLAY ASMT LTERM ltermname
QUERY LTERM SHOW(EMHQ)	/DISPLAY LTERM ltermname QCNT EMHQ
QUERY LTERM SHOW(MSNAME)	/DISPLAY LTERM ltermname
QUERY LTERM SHOW(NODE)	/DISPLAY ASMT LTERM ltermname
QUERY LTERM SHOW(QCNT)	/DISPLAY LTERM ltermname /DISPLAY LTERM ltermname QCNT
QUERY LTERM SHOW(STATUS)	/DISPLAY LTERM ltermname /DISPLAY STATUS LTERM
QUERY LTERM SHOW(USER)	/DISPLAY ASMT LTERM ltermname
QUERY LTERM STATUS(MTO,SMTO)	/DISPLAY MASTER /RDISPLAY MASTER
QUERY LTERM MSGAGE(x)	/DISPLAY QCNT LTERM MSGAGE x
QUERY LTERM STATUS(status)	/DISPLAY STATUS LTERM

Output fields

The following table shows the QUERY LTERM output fields. The columns in the table are:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the column heading for the output field in the formatted output.

SHOW parameter
> Identifies the parameter on the SHOW keyword that caused the field to be generated. *Error* appears for output fields that are returned for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field. GBL indicates that the field can be generated only by the command master when displaying global information for SHOW(GLOBAL). LCL indicates that the field can be generated by any IMS displaying local information for SHOW(LOCAL). N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 104. Output fields for the QUERY LTERM command

Short label	Long label	SHOW parameter	Scope	Meaning
CC	CC	N/A	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 314 for more information. The completion code is always returned.
CCTXT	CCText	*Error*	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.

Table 104. Output fields for the QUERY LTERM command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
DFNT	LDefnType	DEFNTYPE	LCL	Definition type, which can be one of the following: **CREATE** Defined by the CREATE LTERM command. **DFSDSCMx** Defined by an MSC descriptor in the DFSDSCMx member of the IMS PROCLIB data set. **DFSDSCTx** Defined by an MSC descriptor in the DFSDSCTx member of the IMS PROCLIB data set. **DFSINSX0** Defined by the user exit DFSINSX0 to represent an LTERM that is defined remotely in an IMS system in a shared-queues group. **MSCGEN** Defined by system definition in the DFSCLRxy member of the IMS RESLIB. This field is applicable only to the MSC remote LTERM.
EAGE	EmhqAged	MSGAGE	GBL	Count of EMHQ messages with a message age greater than or equal to the message age specified by the MSGAGE() filter. If no filter is specified, the count is the total number of messages queued.
EMHQ	EMHQCnt	EMHQ	GBL	Global LTERM queue count in the EMH (Expedited Message Handler) queues. EMHQ is displayed only if shared EMH is used.
ETNEW	EmhqTStmpNew	MSGAGE	GBL	The time stamp of the newest EMHQ message for the LTERM on the shared queues.
ETOLD	EmhqTStmpOld	MSGAGE	GBL	The time stamp of the oldest EMHQ message for the LTERM on the shared queues.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the MSC remote logical terminal has been exported to the IMSRSC repository. The value can be Y or N.
GBL	Gbl	GLOBAL	GBL	If 'Y', then the output reflects the status found globally in shared queues or RM. If blank, then the output reflects the status found locally.
ICMP	InCmp	COMPONENT	GBL	The input component that the LTERM is assigned to in the resource structure.
IMSID	IMSid	IMSID	GBL	Returns from the repository the IMSIDs that have the resource defined.
LICMP	LInCmp	COMPONENT	LCL	The input component that the LTERM is assigned to in the local system.
LLINE	LLine	NODE	LCL	Identifies the line number for system console, SPOOL, SYSOUT, or TCO device.
LMSN	LMSName	MSNAME	LCL	The associated logical link path name when the LTERM is an MSC remote LTERM.
LNODE	LNode	NODE	LCL	Identifies the dynamic or static node associated with the LTERM on the local system.
LOCMP	LOutCmp	COMPONENT	LCL	The output component that the LTERM is assigned to in the local system.
LPTERM	LPTerm	NODE	LCL	Identifies the PTERM number for the system console, SPOOL, SYSOUT or TCO device.
LQ	LQCnt	QCNT	LCL	Local queue count.
LSIDL	LSIDL	MSNAME	LCL	Local system identification of the associated logical link path name when the LTERM is an MSC remote LTERM.
LSIDR	LSIDR	MSNAME	LCL	Remote system identification of the associated logical link path name when the LTERM is an MSC remote LTERM.
LSTT	LclStat	STATUS	LCL	Local logical terminal status. See "QUERY LTERM status" on page 313 for information about the logical terminal status that can be returned.
LTERM	Lterm	N/A	N/A	Logical terminal name. The logical terminal name is always returned.
LUSER	LUser	USER	LCL	Identifies the dynamic or ISC user associated with the LTERM on the local system.

Table 104. Output fields for the QUERY LTERM command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LVER	LVersion#	VERSION	LCL	Version number for the LTERM resource being maintained in the local system. This field applies only when STM is enabled.
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. The IMS identifier is always returned.
NODE	Node	NODE	GBL	Identifies the dynamic or static node associated with the LTERM in the resource structure.
OCMP	OutCmp	COMPONENT	GBL	The output component that the LTERM is assigned to in the resource structure.
OWNER	Owner	OWNER	GBL	The IMSID (or RSENAME for XRF systems) of the IMS system that "owns" the associated node or user resource. This field is returned only by the command master, and applies only when sysplex terminal management is enabled.
QAGE	QCntAged	MSGAGE	GBL	Count of MSGQ messages with a message age greater than or equal to the message age specified by the MSGAGE() filter. If no filter is specified, the count is the total number of messages queued.
QCNT	QCnt	QCNT	GBL	Global queue count on the shared queues. Global queue count can be displayed only if shared queues are used.
QTNEW	TStmpNew	MSGAGE	GBL	The time stamp of the newest MSGQ message for the LTERM on the shared queues.
QTOLD	TStmpOld	MSGAGE	GBL	The time stamp of the oldest MSGQ message for the LTERM on the shared queues.
REPO	Repo	DEFN	GBL	Indicates whether the line shows the stored resource definitions. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
RMSN	MSName	DEFN, MSNAME	GBL	The associated logical link path name when the LTERM is an MSC remote LTERM. The output is returned from the repository.
RTMCR	TimeCreate	DEFN	GBL	Create time from the repository. This is the time the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN	GBL	Update time from the repository. This is the time the resource was last updated in the repository.
STT	Status	STATUS	GBL	Global logical terminal status from the resource structure. See "QUERY LTERM status" on page 313 for information about the logical terminal status that can be returned.
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the resource was last accessed. This is not yet supported.
TMCR	LTimeCreate	TIMESTAMP	LCL	The time that the resource was created with a CREATE LTERM command, an IMPORT command that creates the MSC remote LTERM, or IMS initialization. The create time is retained across IMS warm start, emergency restart, export, and import. The output is returned from the local IMS. This field is only applicable to the MSC remote LTERM.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the resource was last imported, if applicable. The local import timestamp is set to be the IMS cold start time, if the resource was automatically imported from the IMSRSC repository during IMS cold start. The import time is retained across IMS warm start and emergency restart. The output is returned from the local IMS. This field is applicable only to the MSC remote LTERM.
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated. This is not yet supported.
USER	User	USER	GBL	Identifies the dynamic or ISC user associated with the LTERM in the resource structure.
VER	Version#	VERSION	GBL	Version number for the LTERM resource being maintained in the resource structure. This field applies only when STM is enabled.

QUERY LTERM status

The following table shows the possible LTERM status that can be displayed. The columns in the table are:

Status The LTERM status that is displayed.

STATUS parameter
The STATUS() filter that will select LTERMs with the specified status.

Scope The scope of the status. GBL indicates that the status can be global (it exists in the resource structure when STM is enabled), and is returned with the STT short label. LCL indicates that the status can be local, and is returned with the LSTT short label.

Meaning
Provides a brief description of the status.

Table 105. QUERY LTERM status

Status	STATUS parameter	Scope	Meaning
LOCK	LOCK	LCL	Logical terminal was locked by a /LOCK LTERM command.
MTO	MTO	LCL	Logical terminal is the master terminal for the local IMS.
QERR	QERR	LCL	I/O error has occurred on the queue for this logical terminal.
QLOCK	QLOCK	LCL	Logical terminal is locked from sending any further output or from receiving input that could create additional output for the same logical terminal until the state is reset by a specific request received on the session.
REMOTE	REMOTE	LCL	Logical terminal is defined remotely to this IMSplex, accessible via MSC.
RM	RM	GBL	The LTERM exists in the resource structure managed by RM.
RMACTIVE	RMACTIVE	GBL	The associated user or node is active (signed-on or logged-on) in the IMSplex, as indicated in the RM structure (RM active).
RMOWNED	RMOWNED	GBL	The associated user or node is owned by an IMS system in the IMSplex, as indicated in the RM structure (RM owned).
SMTO	SMTO	LCL	The logical terminal is the secondary master terminal for the local IMS.
STATIC	STATIC	LCL and GBL	Logical terminal was defined during system definition.
STOQ	STOQ	LCL and GBL	Input is stopped for the logical terminal.
STOSEND	STOSEND	LCL and GBL	Output is stopped for the logical terminal.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY LTERM command. The OM return and reason codes that may be returned as a result of the QUERY LTERM command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 106. Return and reason codes for the QUERY LTERM command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001010'	No resources were found to be returned because of one of the following reasons: • The resource names specified might be invalid. • There were no resources that match the filter specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000004'	X'00001000'	The command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000008'	X'00002014'	An invalid character was specified in the resource name.
X'00000008'	X'00002040'	An invalid parameter value was specified. An invalid SHOW or STATUS value might have been specified. A value of 0 might have been specified for QCNT with LE, GE or, EQ. Or a value of 1 might have been specified for QCNT with LT.
X'0000000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See Table 107 on page 316 for details.
X'0000000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code. See Table 107 on page 316 for details.
X'00000010'	X'00004004'	Command processing terminated because CQS was not active.
X'00000010'	X'00004005'	Command processing terminated because CQS was not connected to the queue structure.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	Command processing terminated because the resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'0000402D'	Command is not allowed because dynamic resource definition for MSC resources is not enabled (DFSDFxxx is not defined with MSCRSCS=DYN).

Table 106. Return and reason codes for the QUERY LTERM command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004104'	Command processing terminated because RM is not available.
X'00000010'	X'00004108'	Command processing terminated because SCI is not available.
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000010'	X'00004502'	The repository is not available.
X'00000010'	X'00004503'	The repository is stopped.
X'00000010'	X'00004504'	The repository spare recovery is in process.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Repository access is denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server (RS) is being shut down.
X'00000010'	X'0000450B'	RS is not available.
X'00000010'	X'0000450C'	RS is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields related to the IMSRSC repository.
X'00000014'	X'00005004'	A DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	An RM error occurred.
X'00000014'	X'00005104'	A CQS error occurred.
X'00000014'	X'00005108'	An SCI error occurred.
X'00000014'	X'00005110'	A repository error occurred.
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY LTERM command is rejected because the resource list is unusable. Contact IBM Software Support. Send the output from the command in error, and a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the IMSRSC repository RID and RMD data sets.
X'00000014'	X'00005FFF'	Command processing terminated because of an internal IMS error.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 107. Completion codes for the QUERY LTERM command

Completion code	Completion code text	Meaning
0		The QUERY LTERM command completed successfully for the resource.
1	COMMAND NOT APPLICABLE TO THIS IMS	QUERY LTERM SHOW(EXPORTNEEDED) was specified for a local LTERM, which is not applicable. SHOW(EXPORTNEEDED) is applicable only to remote LTERMs. This command is allowed on the IMSs where this LTERM is defined as remote.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
98	CQS REQUEST ERROR	Global queue counts could not be obtained because of a CQS error.
1A0	Lterm resource is in error	The LTERM resource was found in the resource structure. An associated resource was needed, but it was either not found or appeared to be in error. This is normally an error condition, but it could be a temporary condition caused by terminal or command activity. Retry the command.

Examples

The following are examples of the QUERY LTERM command:

Example 1 for QUERY LTERM command

TSO SPOC input:
```
QUERY LTERM NAME(LTERM20B,LTERM24M,LTERM23C,HOG0C,LTERM01) SHOW(DEFN)
```

TSO SPOC output:
```
(screen 1)
Lterm     MbrName   CC Repo MSName   LMSName   LSIDR LSIDL
HOG0C     IMS1       0 Y    LINK32T1
HOG0C     IMS1       0              LINK32T1   86    117
LTERM01   IMS1       0 Y    MSNV001
LTERM01   IMS1       0              MSNV001    56    706
LTERM20B  IMS1       0 Y    LINK32B1
LTERM20B  IMS1       0              LINK32B1   21    31
LTERM23C  IMS1       0 Y    LINK32C1
LTERM23C  IMS1       0              LINK32C1   23    33
LTERM24M  IMS1       0 Y    LINK32M1
```

```
                              LTERM24M IMS1          0                LINK32M1    24    34

                              (screen 2)
                              Lterm     MbrName  Repo SIDL TimeCreate          LTimeCreate          TimeUpdate
                              HOG0C     IMS1     Y         2015.310 14:57:33.98
                              HOG0C     IMS1          117                       2015.310 14:57:33.98
                              LTERM01   IMS1     Y         2015.310 14:58:42.79
                              LTERM01   IMS1          706                       2015.310 14:58:42.79
                              LTERM20B  IMS1     Y         2015.310 14:57:33.98
                              LTERM20B  IMS1           31                       2015.310 14:57:33.98
                              LTERM23C  IMS1     Y         2015.310 14:57:33.98
                              LTERM23C  IMS1           33                       2015.310 14:57:33.98
                              LTERM24M  IMS1     Y         2015.310 14:57:33.98
                              LTERM24M  IMS1           34                       2015.310 14:57:33.98

                              (screen 3)
                              Lterm     MbrName  Repo LTimeUpdate          LTimeAccess
                              HOG0C     IMS1     Y
                              HOG0C     IMS1
                              LTERM01   IMS1     Y
                              LTERM01   IMS1
                              LTERM20B  IMS1     Y
                              LTERM20B  IMS1
                              LTERM23C  IMS1     Y
                              LTERM23C  IMS1
                              LTERM24M  IMS1     Y
                              LTERM24M  IMS1
```

OM API input:

```
CMD(QRY LTERM NAME(LTERM20B,LTERM24M,LTERM23C,HOG0C,LTERM01) SHOW(DEFN)
```

OM API output:

```
<cmdrsphdr>
<hdr slbl="LTERM" llbl="Lterm" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LNODE" llbl="LNode" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LLINE" llbl="LLine" scope="LCL" sort="d" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LPTERM" llbl="LPterm" scope="LCL" sort="d" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LICMP" llbl="LInCmp" scope="LCL" sort="d" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LOCMP" llbl="LOutCmp" scope="LCL" sort="d" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LUSER" llbl="LUser" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RMSN" llbl="MSName" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LMSN" llbl="LMSName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LSIDR" llbl="LSIDR" scope="LCL" sort="d" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LSIDL" llbl="LSIDL" scope="LCL" sort="d" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="RTMCR" llbl="TimeCreate" scope="GBL" sort="n" key="0"
```

```
scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMCR" llbl="LTimeCreate"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMIM" llbl="LTimeImport"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RTMUP" llbl="TimeUpdate"scope="GBL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMUP" llbl="LTimeUpdate"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMAC" llbl="LTimeAccess"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="EXPN" llbl="ExportNeeded" scope="LCL" sort="y" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>LTERM(LTERM20B) MBR(IMS1) CC(   0) LQ(0) LMSN(LINK32B1) LSIDR(21)
 LSIDL(31) LVER(0) LSTT(REMOTE) DFNT(MSCGEN  ) TMCR(2015.310
 14:57:33.98) </rsp>
<rsp>LTERM(LTERM24M) MBR(IMS1) CC(   0) LQ(0) LMSN(LINK32M1) LSIDR(24)
 LSIDL(34) LVER(0) LSTT(REMOTE) DFNT(MSCGEN  ) TMCR(2015.310
 14:57:33.98) </rsp>
<rsp>LTERM(LTERM23C) MBR(IMS1) CC(   0) LQ(0) LMSN(LINK32C1) LSIDR(23)
 LSIDL(33) LVER(0) LSTT(REMOTE) DFNT(MSCGEN  ) TMCR(2015.310
 14:57:33.98) </rsp>
<rsp>LTERM(HOG0C  ) MBR(IMS1) CC(   0) LQ(0) LMSN(LINK32T1) LSIDR(86)
 LSIDL(117) LVER(0) LSTT(REMOTE) DFNT(MSCGEN  ) TMCR(2015.310
 14:57:33.98) </rsp>
<rsp>LTERM(LTERM01 ) MBR(IMS1) CC(   0) LQ(0) LMSN(MSNV001 ) LSIDR(56)
 LSIDL(706) LVER(0) LSTT(REMOTE) DFNT(CREATE  ) TMCR(2015.310
 14:58:42.79) </rsp>
<rsp>LTERM(LTERM20B) MBR(IMS1    ) CC(   0) REPO(Y) IMSID()
 RMSN(LINK32B1) RTMCR(2015.310 14:57:33.98) </rsp>
<rsp>LTERM(LTERM24M) MBR(IMS1    ) CC(   0) REPO(Y) IMSID()
 RMSN(LINK32M1) RTMCR(2015.310 14:57:33.98) </rsp>
<rsp>LTERM(LTERM23C) MBR(IMS1    ) CC(   0) REPO(Y) IMSID()
 RMSN(LINK32C1) RTMCR(2015.310 14:57:33.98) </rsp>
<rsp>LTERM(HOG0C  ) MBR(IMS1    ) CC(   0) REPO(Y) IMSID()
 RMSN(LINK32T1) RTMCR(2015.310 14:57:33.98) </rsp>
<rsp>LTERM(LTERM01 ) MBR(IMS1    ) CC(   0) REPO(Y) IMSID()
 RMSN(MSNV001 ) RTMCR(2015.310 14:58:42.79) </rsp>
</cmdrspdata>
```

Explanation: This QUERY command displays several remote LTERMs, with their
only attribute, MSNAME.

Example 2 for QUERY LTERM command

TSO SPOC input:
```
QRY LTERM NAME(LTERM0*)
```

TSO SPOC output:
```
Lterm     MbrName CC  Gbl
LTERM01   IMS1    0
LTERM02   IMS1    0   Y
LTERM02   IMS1    0
LTERM02   IMS2    0
LTERM03   IMS2    0
LTERM04   IMS1    0   Y
LTERM04   IMS1    0
LTERM04   IMS2    0
LTERM05   IMS1    0   Y
LTERM05   IMS1    0
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information only.

- LTERM01 exists on IMS1 only.
- LTERM02 exists on IMS1 and IMS2, and in the resource structure.
- LTERM03 exists on IMS2 only.
- LTERM04 exists on IMS1, IMS2, and in the resource structure.
- LTERM05 exists on IMS1 and in the resource structure.

Example 3 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(LTERM0*) SHOW(GLOBAL,STATUS,QCNT,OWNER)
```

TSO SPOC output:

```
Lterm    MbrName CC  Gbl   QCnt  Owner   Status
LTERM02  IMS1     0  Y        0  IMS2    STATIC,RM,RMACTIVE,RMOWNED
LTERM03  IMS1     0  Y        1
LTERM04  IMS1     0  Y        3  IMS2    STATIC,RM,RMOWNED
LTERM05  IMS1     0  Y        0          STOQ,STOSEND,RM
LTERM06  IMS1     0  Y        2
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are enabled. IMS1, the command master, displays global information. IMS2 ignores the command (RC=4, RSN=x1000) because SHOW(GLOBAL) is specified.

- LTERM02 exists in the resource structure, is currently active on IMS2, and has no messages in the shared queues.
- LTERM03 does not exist in the resource structure, but has one message in the shared queues.
- LTERM04 exists in the resource structure, is not currently active, but is owned by IMS2 which indicates status exists on IMS2, and has three messages in the shared queues.
- LTERM05 exists in the resource structure, is stopped, is not currently active or owned, and has no messages in the shared queues.
- LTERM06 does not exist in the resource structure, but has two messages in the shared queues.

Example 4 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(LTERM01,LTERM02,LTERM04) SHOW(QCNT,EMHQ)
```

TSO SPOC output:

```
Lterm    MbrName CC  CCText             Gbl  QCnt  EMHQCnt  LQCnt
LTERM01  IMS1     0                     Y       0        0
LTERM01  IMS1     0                                               0
LTERM01  IMS2    10  NO RESOURCES FOUND
LTERM02  IMS1     0                     Y       0        1
LTERM02  IMS1     0                                               0
LTERM02  IMS2     0                                               0
LTERM04  IMS1     0                     Y       3        0
LTERM04  IMS1     0                                               0
LTERM04  IMS2     0                                               0
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are enabled. IMS1, the command master, displays global and local information. IMS2 displays local information only.
- LTERM01 exists on IMS1 only, and has no messages in shared queues.
- LTERM02 exists on IMS1 and IMS2, and has one message in shared EMH.
- LTERM04 exists on IMS1 and IMS2, and has three messages in shared queues.

Example 5 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM MSGAGE(7) QCNT(GE,3)
```

TSO SPOC output:

```
Lterm    MbrName CC Gbl   QCnt  QCntAged TStmpOld    TStmpNew
LTERM04  IMS1    0  Y       3         0 07128/140758 07128/141005
LTERM06  IMS1    0  Y       2         2 07120/110113 07120/110240
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is irrelevant because global status other than queue counts is not requested. Shared queues are enabled. IMS1, the command master, displays global information. IMS2 ignores the command (RC=4, RSN=x1000) because QCNT() in a shared-queues environment is processed by the command master only.
- LTERM04 has three messages queued, which satisfies the QCNT filter, and all messages are less than 7 days old.
- LTERM06 has two messages queued that are older than 7 days, which satisfies the MSGAGE filter.

Example 6 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(*) QCNT(GE,2)
```

TSO SPOC output:

```
Lterm    MbrName CC Gbl   QCnt
LTERM04  IMS1    0  Y       3
LTERM06  IMS1    0  Y       2
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM is irrelevant because only queue information is requested. Shared queues are enabled. IMS1, the command master, displays global information. IMS2 ignores the command (RC=4, RSN=x1000) because QCNT() in a shared-queues environment is processed by the command master only. The QCnt output is displayed even if SHOW(QCNT) was not specified.
- LTERM04 has three messages in the shared queues.
- LTERM06 has two messages in the shared queues.

Example 7 for QUERY LTERM command

TSO SPOC input:

```
QRY LTERM NAME(LTERM04,LTERM05,LTERM11) SHOW(LOCAL,ALL)
```

TSO SPOC output:

(screen 1)
```
Lterm    MbrName CC CCText              LQCnt  LNode   LInCmp LOutCmp
LTERM04  IMS1    0                          0  NODE04       1       1
LTERM04  IMS2    0                          0  NODE04       1       1
```

```
| LTERM05  IMS1     0                              0              1        1
| LTERM05  IMS2    10 NO RESOURCES FOUND
| LTERM11  IMS1    10 NO RESOURCES FOUND
| LTERM11  IMS2     0                              0
|
| (scrolled right to screen 2)
| Lterm     MbrName LUser   LMSName   LSIDR LSIDL LVersion# LclStat
| LTERM04  IMS1                                            0 STATIC
| LTERM04  IMS2                                            7 STATIC
| LTERM05  IMS1    USER05                                  2 STOQ,STOSEND
| LTERM05  IMS2
| LTERM11  IMS1
| LTERM11  IMS2            LINK2111   21    11            0 STATIC,REMOTE
| (scrolled right to screen 3)
| Lterm     MbrName LTimeCreate
| LTERM04  IMS1    2014.283 15:29:52:12
| LTERM04  IMS2    2014.283 11:08:14:31
| LTERM05  IMS1    2014.283 15:29:52:12
| LTERM05  IMS2    2014.283 11:08:14:31
| LTERM11  IMS1    2014.283 15:29:52:12
| LTERM11  IMS2    2014.283 11:08:14:31
| (scrolled right to screen 4)
| Lterm     MbrName LTimeUpdate     LTimeAccess              LDefnType
| LTERM04  IMS1                                              MSCGEN
| LTERM04  IMS2                                              MSCGEN
| LTERM05  IMS1                                              MSCGEN
| LTERM05  IMS2                                              MSCGEN
| LTERM11  IMS1                                              MSCGEN
| LTERM11  IMS2                                              MSCGEN
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM is
enabled. Shared queues are irrelevant because global queue counts are not
requested. IMS1, the command master, displays local information only because
only LOCAL is specified. IMS2 displays local information only.
- LTERM04 exists on IMS1 and IMS2. The LTERM is statically allocated to
  NODE04 on all systems.
- LTERM05 exists on IMS1 only. The LTERM is dynamic, allocated to USER05, and
  is stopped. There is no node because the user is not signed on.
- LTERM11 exists on IMS2 only. It is defined as a remote LTERM. The associated
  logical link name is LINK2111, and the remote and local SIDs are 21 and 11.

Example 8 for QUERY LTERM command

TSO SPOC input:
```
QRY LTERM NAME(LTERM04,LTERM05,LTERM11) SHOW(ALL)
```

TSO SPOC output:
```
| (screen 1)
| Lterm     MbrName CC CCText              Gbl   QCnt  EMHQCnt  Owner  Node
| LTERM04  IMS1     0                      Y        0        0  IMS2   NODE04
| LTERM04  IMS1     0
| LTERM04  IMS2     0
| LTERM05  IMS1     0                      Y        0        0
| LTERM05  IMS1     0
| LTERM05  IMS2    10 NO RESOURCES FOUND
| LTERM11  IMS1    10 NO RESOURCES FOUND Y
| LTERM11  IMS1    10 NO RESOURCES FOUND
| LTERM11  IMS2     0
|
| (scrolled right to screen 2)
| Lterm     MbrName Gbl   InCmp OutCmp User     Version# QcntAged
| LTERM04  IMS1     Y         1      1              7        3
| LTERM04  IMS1
```

```
|             LTERM04  IMS2
|             LTERM05  IMS1   Y        1        1 USER05         2          0
|             LTERM05  IMS1
|             LTERM05  IMS2
|             LTERM11  IMS1   Y
|             LTERM11  IMS1
|             LTERM11  IMS2
|
|             (scrolled right to screen 3)
|             Lterm    MbrName Gbl  TStmpOld     TStmpNew     EmhqAged
|             LTERM04  IMS1   Y    07128/140758 07128/141005        0
|             LTERM04  IMS1
|             LTERM04  IMS2
|             LTERM05  IMS1   Y                                     0
|             LTERM05  IMS1
|             LTERM05  IMS2
|             LTERM11  IMS1   Y
|             LTERM11  IMS1
|             LTERM11  IMS2
|
|             (scrolled right to screen 4)
|             Lterm    MbrName Gbl  EmhqTStmpOld EmhqTStmpNew Status
|             LTERM04  IMS1   Y                               STATIC,RM,RMOWNED
|             LTERM04  IMS1
|             LTERM04  IMS2
|             LTERM05  IMS1   Y                               STOQ,STOSEND,RM
|             LTERM05  IMS1
|             LTERM05  IMS2
|             LTERM11  IMS1   Y
|             LTERM11  IMS1
|             LTERM11  IMS2
|
|             (scrolled right to screen 5)
|             Lterm    MbrName Gbl  LQCnt LNode   LInCmp LOutCmp LUser   LMSName
|             LTERM04  IMS1   Y
|             LTERM04  IMS1        0 NODE04      1       1
|             LTERM04  IMS2        0 NODE04      1       1
|             LTERM05  IMS1   Y
|             LTERM05  IMS1        0             1       1 USER05
|             LTERM05  IMS2
|             LTERM11  IMS1   Y
|             LTERM11  IMS1
|             LTERM11  IMS2        0                               LINK2111
|
|             (scrolled right to screen 6)
|             Lterm    MbrName Gbl  LSIDR LSIDL LVersion# LclStat
|             LTERM04  IMS1   Y
|             LTERM04  IMS1                        0 STATIC
|             LTERM04  IMS2                        7 STATIC
|             LTERM05  IMS1   Y
|             LTERM05  IMS1                        2 STOQ,STOSEND
|             LTERM05  IMS2
|             LTERM11  IMS1   Y
|             LTERM11  IMS1
|             LTERM11  IMS2          21   11       0 STATIC,REMOTE
|             (scrolled right to screen 7)
|             Lterm    MbrName Gbl  LTimeCreate         LTimeUpdate         LTimeAccess
|             LTERM04  IMS1   Y
|             LTERM04  IMS1
|             LTERM04  IMS2
|             LTERM05  IMS1   Y
|             LTERM05  IMS1
|             LTERM05  IMS2
|             LTERM11  IMS1   Y
|             LTERM11  IMS1
|             LTERM11  IMS2          2014.183 16:14:17:12
|             (scrolled right to screen 8)
```

```
|              Lterm    MbrName Gbl  LDefnType
|              LTERM04  IMS1    Y
|              LTERM04  IMS1
|              LTERM04  IMS2
|              LTERM05  IMS1    Y
|              LTERM05  IMS1
|              LTERM05  IMS2
|              LTERM11  IMS1    Y
|              LTERM11  IMS1
|              LTERM11  IMS2           MSCGEN
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM and shared queues are enabled. IMS1, the command master, displays global and local information. IMS2 displays local information only.

- LTERM04 exists on IMS1, IMS2, and in the resource structure. Global status indicates it is not currently active, but is owned by IMS2 which indicates significant status exists for the node on IMS2. The LTERM is statically allocated to NODE04 on all systems.
- LTERM05 exists on IMS1 and in the resource structure. LTERM05 is dynamic. Global status indicates it is stopped, but is not active or owned on any system. The local status on IMS1 indicates it is stopped. The LTERM is allocated to user USER05 in both IMS1 and RM. There is no node because the user is not signed on. LTERM11 exists on IMS2 only, so there is no global status. It is defined as a remote LTERM. The associated logical link name is LINK2111, and the remote and local SIDs are 21 and 11.

| *Example 9 for QUERY LTERM command*

| TSO SPOC input:

| `QRY LTERM NAME(RLTE*) SHOW(TIMESTAMP,MSNAME,DEFNTYPE)`

| TSO SPOC output:

```
| (screen 1)
| Lterm    MbrName    CC LMSName  LSIDR LSIDL LTimeCreate
| RLTE0001 IMS1        0 LINK12V1    20    10 2015.061 16:10:42.15
| RLTE0002 IMS1        0 LINK12V1    20    10 2015.061 16:11:23.77
| RLTE0003 IMS1        0 LINK12V1    20    10 2015.061 16:11:23.77
|
| (scrolled right to screen 2)
| Lterm    MbrName LTimeUpdate            LTimeAccess            LDefnType
| RLTE0001 IMS1                                                  CREATE
| RLTE0002 IMS1                                                  CREATE
| RLTE0003 IMS1                                                  CREATE
```

| OM API input:

| `CMD(QRY LTERM NAME(RLTE*) SHOW(TIMESTAMP,MSNAME,DEFNTYPE))`

| OM API output:

```
| <imsout>
| <ctl>
| <omname>OM1OM    </omname>
| <omvsn>1.7.0</omvsn>
| <xmlvsn>20   </xmlvsn>
| <statime>2015.062 00:17:28.048737</statime>
| <stotime>2015.062 00:17:28.049106</stotime>
| <staseq>CE9678B45F46150C</staseq>
| <stoseq>CE9678B45F5D2B8C</stoseq>
| <rqsttkn1>USRT011 10161728</rqsttkn1>
| <rc>00000000</rc>
| <rsn>00000000</rsn>
| </ctl>
```

```
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>LTERM          </kwd>
<input>QRY LTERM NAME(RLTE*) SHOW(DEFNTYPE,TIMESTAMP,MSNAME) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="LTERM" llbl="Lterm" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LMSN" llbl="LMSName" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LSIDR" llbl="LSIDR" scope="LCL" sort="d" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="LSIDL" llbl="LSIDL" scope="LCL" sort="d" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="TMCR" llbl="LTimeCreate"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMIM" llbl="LTimeImport"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TMUP" llbl="LTimeUpdate"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMAC" llbl="LTimeAccess"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="DFNT" llbl="LDefnType"scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>LTERM(RLTE0003) MBR(IMS1) CC(   0) LMSN(LINK12V1) LSIDR(20)
 LSIDL(10) DFNT(CREATE ) TMCR(2015.061 16:11:23.77) </rsp>
<rsp>LTERM(RLTE0002) MBR(IMS1) CC(   0) LMSN(LINK12V1) LSIDR(20)
 LSIDL(10) DFNT(CREATE ) TMCR(2015.061 16:11:23.77) </rsp>
<rsp>LTERM(RLTE0001) MBR(IMS1) CC(   0) LMSN(LINK12V1) LSIDR(20)
 LSIDL(10) DFNT(CREATE ) TMCR(2015.061 16:10:42.15) </rsp>
</cmdrspdata>
</imsout>
```

Three remote LTERMS that were created dynamically are displayed. They all have
a definition type of CREATE, because they were created by a CREATE LTERM
command.

Related concepts:

How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

/DISPLAY ASMT command (Commands)

/DISPLAY LTERM command (Commands)

/DISPLAY MASTER command (Commands)

/DISPLAY QCNT command (Commands)

/DISPLAY STATUS command (Commands)

Command keywords and their synonyms (Commands)

QUERY MEMBER command

Use the QUERY MEMBER command to display status or attribute information about one or more members of the IMSplex.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 326
- "Output fields" on page 337
- "Return, reason, and completion codes" on page 338
- "Examples" on page 339

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY MEMBER command and keywords can be issued.

Table 108. Valid environments for the QUERY MEMBER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY MEMBER	X	X	X
ALL	X	X	X
ATTRIB	X	X	X
SHOW	X	X	X
STATUS	X	X	X
TYPE	X	X	X

Syntax
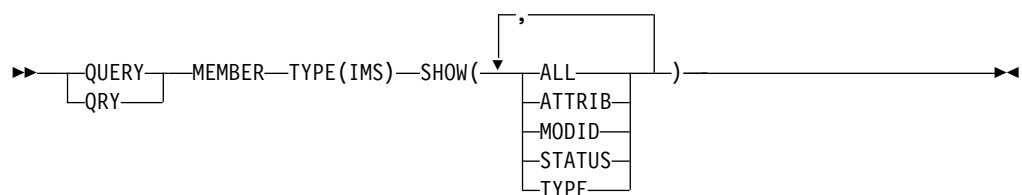
Keywords

The following keywords are valid for the QUERY MEMBER command:

SHOW()
Specifies the output fields to be returned.

> **ALL**
> Returns all the output fields.
>
> **ATTRIB**
> Displays the IMSplex member attributes. These are static definitions.

LOCAL

>Displays the local status information about one or more members of the IMSplex.

GLOBAL

>Displays the global status information about one or more members of the IMSplex.

MODID

>Displays the online change modify id. The modify id is incremented by each successful online change. During a global online change, an IMS's local modify id indicates whether the IMS has reached the online change commit phase 2 and is synchronized with the OLCSTAT data set. If the modify id is initialized to zero by the Global Online Change utility, the modify id represents the number of global online changes that have successfully completed.

>The local modify id on an XRF alternate system indicates whether the XRF alternate has gotten the X'70' log record, performed online change, and is synchronized with the OLCSTAT data set. If the XRF alternate's modid matches the OLCSTAT data set modid as displayed by the QUERY OLC LIBRARY (OLCSTAT) command, the XRF alternate is synchronized with the OLCSTAT data set.

STATUS

>Displays the IMSplex member status. IMSplex member status can change dynamically.

TYPE

>Shows the IMSplex member type. The IMSplex member type can be IMS, for the IMS address space.

TYPE()

Specifies the IMSplex member type for which information will be displayed. IMS is the only type available.

Usage notes

The QUERY MEMBER command can be specified only through the OM API.

The QUERY MEMBER command shows whether dynamic resource definition is enabled for MSC and MODBLKS resources. When the QUERY MEMBER command is issued with either SHOW(ALL) or SHOW(ATTRIB) specified, the LclAttr column displays `DYNMODBLKS` if dynamic resource definition is enabled for MODBLKS resources and displays `DYNMSC` if dynamic resource definition is enabled for MSC resources.

The QUERY MEMBER TYPE(ATTRIB) command shows an attribute of `DIRECTORY` if the IMS system is enabled with IMS management of ACBs. An attribute of `DIRECTORY` is shown if the IMS system is enabled with the IMS catalog and IMS management of ACBs.

There are two commands named QUERY MEMBER. QUERY MEMBER TYPE(IMS) is a type-2 command that gets information about IMS members from TSO SPOC or the OM API. QUERY MEMBER TYPE(IMSCON) is a z/OS modify command that gets information about IMS Connect members. See IMS Connect QUERY MEMBER command (Commands) for more information.

QUERY MEMBER attributes

The following table shows the possible IMS attributes. The table contains information about attributes such as the attribute that is returned, the scope of the attribute, and the meaning of the attribute. Global attributes are returned with the ATTR short label. Local attributes are returned with the LATTR short label.

Table 109. Attributes for QUERY MEMBER command

Attribute	Scope	Meaning
DIRECTORY	LCL	The ACBs for this IMS system are managed by IMS in the directory data sets of the IMS catalog. ACBMGMT=CATALOG is specified in the CATALOG section of the DFSDFxxx PROCLIB member.
DYNMODBLKS	LCL	Dynamic resource definition is enabled for MODBLKS resources.
DYNMSC	LCL	Dynamic resource definition is enabled for Multiple Systems Coupling (MSC) resources.
GBLOLC	LCL	Global online change is enabled.
NO-STM	LCL	IMS is not sharing terminal resources because STM=NO was specified in DFSDCxxx PROCLIB member or IMS is registered to an RM that is not using a resource structure.
RMENVNO	LCL	No RM environment is running.
RSRTRK	LCL	Remote Site Recovery tracker.
SHAREDQ	LCL	Shared queues are enabled.

QUERY MEMBER status

The following table shows the possible member status. The table contains information about status such as the status that is returned, the scope of the status, and the meaning of the status. Global status is returned with the STT short label. Local status is returned with the LSTT short label. A scope of LCL means that the status is local to the IMS specified and is returned with the LSTT short label. A scope of GBL means that the status is global to all the IMS systems and is returned with the STT short label.

The output from the QUERY MEMBER command may contain an additional value to reflect that the OPTION(UPDATEPSB) or OPTION(REFRESHPSB) is in progress on a given IMS system in the IMSplex.

Table 110. Status for QUERY MEMBER command

Status	Scope	Meaning
AUTODDL	GBL	The automatic activation of resources in the IMS catalog directory data sets was initiated by the submission of DDL statements that included a COMMIT statement. This status remains until the import from IMS catalog completes successfully or, if there is a failure, is backed out or cleaned up by another DDL definition activation or an IMPORT DEFN SOURCE(CATALOG) command.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
IMPABORT	GBL	IMS catalog import processing failed and take over processing will abort import processing.
		The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended. The updates made to the directory will be aborted. This status remains until abort processing completes successfully or fails.
IMPCMD	GBL	An IMPORT DEFN SOURCE(CATALOG) command to update the IMS catalog directory data sets was initiated.
		This status remains until the import from IMS catalog completes successfully or, if there is a failure, is backed out or cleaned up by another DDL definition activation or an IMPORT DEFN SOURCE(CATALOG) command.
IMPCMT	GBL	IMS catalog import processing failed after commit and the take over process will complete the import.
		The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended. The updates made to the IMS catalog directory data sets will be committed.
IMPPHC0I	LCL	IMS catalog import commit phase 0 processing is in progress.
		IMS catalog import processing has completed for both prepare and quiesce phases and the commit phase is in progress.
IMPPHC0F	GBL	IMS catalog import commit phase 0 failed.
		The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 0. A take over attempt failed, or there was no active IMS to perform take over.
		Reenter the IMPORT DEFN SOURCE(CATALOG) command or resubmit the DDL statements to finish the import from the IMS catalog and the clean up process.
IMPPHC0T	GBL	IMS catalog import processing was taken over during commit phase 0.
		The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 0. The IMS catalog import processing was taken over by another IMS to abort.
		This IMS participates in the take over process to cleanup the import processing in this IMS. If message DFS4833E, DFS4839W, DFS4913E, or DFS5006I was issued during take over, the takeover failed. Reenter the IMPORT command or resubmit the DDL statements. The next import processing will cleanup the previous RM process information and the directory data sets.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
IMPPHC0UPF	GBL	IMS catalog import commit phase 0 failed. The IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) command master abended during commit phase 0. A take over attempt failed, or there was no active IMS to perform take over. Reenter the IMPORT command again to finish the import from IMS catalog and clean up.
IMPPHC1I	GBL	IMS catalog import commit phase 1 processing is in progress.
IMPPHC1F	GBL	IMS catalog import commit phase 1 failed. The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 1. A take over attempt failed, or there was no active IMS to perform take over. Reenter the IMPORT DEFN SOURCE(CATALOG) command or resubmit the DDL definition activation, to finish the import from IMS catalog and clean up. If the IMS directory is shared by multiple IMS systems and there is no active IMS to process the IMPORT command or DDL statements, you must run the IMS Directory Recovery utility (DFS3RU00) to clean up the IMS directory.
IMPPHC1T	GBL	IMS catalog import commit phase 1 was taken over. The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 1. The abort of the IMS catalog import processing was taken over by another IMS. This IMS participates in the take over process to cleanup the import processing in this IMS. If message DFS4833E, DFS4839W, DFS4913E, or DFS5006I was issued during take over, the takeover failed. Reenter the IMPORT command or resubmit the DDL statements. The next import processing will cleanup the previous RM process information and the directory data sets.
IMPPHC1UPF	GBL	IMS catalog import commit phase 1 failed. The IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) command master abended during commit phase 1. A take over attempt failed, or there was no active IMS to perform take over. Reenter the IMPORT command again to finish the import from IMS catalog and clean up.
IMPPHC2I	LCL	IMS catalog import commit phase 2 processing is in progress.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
IMPPHC2F	GBL	IMS catalog import commit phase 2 failed.
		The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 2. A take over attempt failed, or there was no active IMS to perform take over.
		Reenter the IMPORT DEFN SOURCE(CATALOG) command or resubmit the DDL definition activation, to finish the import from IMS catalog and clean up.
		If the IMS directory is shared by multiple IMS systems and there is no active IMS to process the IMPORT command or DDL statements, you must run the IMS Directory Recovery utility (DFS3RU00) to clean up the IMS directory.
IMPPHC2T	GBL	IMS catalog import commit phase 2 was taken over.
		The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 2. The abort of the IMS catalog import processing was taken over by another IMS.
		This IMS participates in the take over process to cleanup the import processing in this IMS.
		If message DFS4833E, DFS4796E, DFS4797E, DFS4839W, DFS4913E, or DFS5006I was issued during take over, the takeover failed. Reenter the IMPORT command or resubmit the DDL statements. The next import processing will cleanup the previous RM process information and the directory data sets.
IMPPHC2UPF	GBL	IMS catalog import commit phase 2 failed.
		The IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) command master abended during commit phase 2. A take over attempt failed, or there was no active IMS to perform take over.
		Reenter the IMPORT command again to finish the import from IMS catalog and clean up.
IMPPHC3I	LCL	IMS catalog import commit phase 3 processing is in progress.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
IMPPHC3F	GBL	IMS catalog import commit phase 3 failed. The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 3. A take over attempt failed, or there was no active IMS to perform take over. Reenter the IMPORT DEFN SOURCE(CATALOG) command or resubmit the DDL definition activation, to finish the import from IMS catalog and clean up. If the IMS directory is shared by multiple IMS systems and there is no active IMS to process the IMPORT command or DDL statements, you must run the IMS Directory Recovery utility (DFS3RU00) to clean up the IMS directory.
IMPPHC3T	GBL	IMS catalog import commit phase 3 was taken over. The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 3. The abort of the IMS catalog import processing was taken over by another IMS. This IMS participates in the take over process to cleanup the import processing in this IMS. If message DFS4833E, DFS4796E, DFS4797E, DFS4839W, DFS4913E, or DFS5006I was issued during take over, the takeover failed. Reenter the IMPORT command or resubmit the DDL statements. The next import processing will cleanup the previous RM process information and the directory data sets.
IMPPHC3UPF	GBL	IMS catalog import commit phase 3 failed. The IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) command master abended during commit phase 3. A take over attempt failed, or there was no active IMS to perform take over. Reenter the IMPORT command again to finish the import from IMS catalog and clean up.
IMPPHC4I	LCL	IMS catalog import commit phase 4 processing is in progress.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
IMPPHC4F	GBL	IMS catalog import commit phase 4 failed.
		The IMPORT DEFN SOURCE(CATALOG) command master or the IMS where the DDL definitions were activated abended during commit phase 4. A take over attempt failed, or there was no active IMS to perform take over.
		If a previous IMPORT DEFN SOURCE(CATALOG) command was in progress, reenter the IMPORT command to cleanup RM process information and the directory data sets.
		If the previous DDL activation was for CREATE statements, then the activation is committed to the IMS catalog database but not to the IMS directory data sets. Submit both DROP DDL statements and CREATE DDL definition and commit statements to cleanup RM process information and the directory data sets.
		If the previous DDL activation was a DROP statement, enter the IMPORT command to cleanup RM process and the directory data sets.
		If the IMS directory is shared by multiple IMS systems and there is no active IMS to process the IMPORT command or DDL statements, you must run the IMS Directory Recovery utility (DFS3RU00) to clean up the IMS directory.
IMPPHC4T	GBL	IMS catalog import commit phase 4 was taken over.
		The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 4. The commit of the IMS catalog import processing was taken over by another IMS.
		This IMS participates in the take over process to cleanup the import processing in this IMS.
		If message DFS4832E, DFS4834E, DFS4839W, or DFS5006I was issued during take over, the takeover failed. Reenter the IMPORT command or resubmit the DDL statements. The next import processing will cleanup the previous RM process information and the directory data sets.
IMPPHC4UPF	GBL	IMS catalog import commit phase 4 failed.
		The IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) command master abended during commit phase 4. A take over attempt failed, or there was no active IMS to perform take over.
		Reenter the IMPORT command again to clean up RM process information.
IMPPHC5I	LCL	IMS catalog import commit phase 5 processing is in progress.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
IMPPHC5F	GBL	IMS catalog import commit phase 5 failed.
		The IMPORT DEFN SOURCE(CATALOG) command master or the IMS where the DDL definitions were activated abended during commit phase 5. A take over attempt failed, or there was no active IMS to perform take over.
		If a previous IMPORT DEFN SOURCE(CATALOG) command was in progress, reenter the IMPORT command to cleanup RM process information and the directory data sets.
		If the previous DDL activation was for CREATE statements, then the activation is committed to the IMS catalog database but not to the IMS directory data sets. Submit both DROP DDL statements and CREATE DDL definition and commit statements to cleanup RM process information and the directory data sets.
		If the previous DDL activation was a DROP statement, enter the IMPORT command to cleanup RM process and the directory data sets.
		If the IMS directory is shared by multiple IMS systems and there is no active IMS to process the IMPORT command or DDL statements, you must run the IMS Directory Recovery utility (DFS3RU00) to clean up the IMS directory.
IMPPHC5T	GBL	IMS catalog import commit phase 5 was taken over.
		The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during commit phase 5. The commit of the IMS catalog import processing was taken over by another IMS.
		This IMS participates in the take over process to cleanup the import processing in this IMS.
		If message DFS4832E, DFS4834E, DFS4839W, or DFS5006I was issued during take over, the takeover failed. Reenter the IMPORT command or resubmit the DDL statements. The next import processing will cleanup the previous RM process information and the directory data sets.
		If the QUERY MEMBER command also shows another IMS with a local status of IMPHC5I, this indicates the import processing is in final cleanup stage. This non master IMS has already completes its cleaning up while other IMS systems are in progress.
IMPPHC5UPF	GBL	IMS catalog import commit phase 5 failed.
		The IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) command master abended during commit phase 5. A take over attempt failed, or there was no active IMS to perform take over.
		Reenter the IMPORT command again to clean up RM process information.
IMPPHPI	LCL	IMS catalog import prepare phase processing is in progress.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
IMPPHPT	GBL	IMS catalog import prepare phase was taken over.
		The IMPORT DEFN SOURCE(CATALOG) command master IMS or the IMS where the DDL definitions were activated abended during the prepare phase. The abort of IMS catalog import processing was taken over by another IMS.
		This IMS participates in the take over process to cleanup the import processing in this IMS.
		Reenter the IMPORT command or resubmit the DDL statements. The next import processing will cleanup the previous RM process information if message CSL2200I CLIENT client INITIATED PROCESS DFSOLC was issued.
LEOPT	LCL	Language Environment options are enabled for this IMS.
OLCABRTC	LCL	Online change abort completed.
		A TERMINATE OLC command or /MODIFY ABORT command is entered. Online change abort phase completed locally for this IMS. The IMS is taken out of the online change state.
OLCABRTI	LCL	Online change abort in progress.
		A TERMINATE OLC command or /MODIFY ABORT command is entered. Online change abort phase is in progress locally for this IMS.
OLCCMT1C	LCL or GBL	Online change commit phase 1 completed.
		An INITIATE OLC PHASE(COMMIT) command or /MODIFY COMMIT command is entered. Online change commit phase 1 completed either locally for the IMS, or globally for all of the IMS systems in the IMSplex. After all of the IMS systems have attempted commit phase 1, the online change master updates the OLCSTAT data set and the online change is considered to be complete.
		Type-1 commands from the system console, an IMS terminal, or the MTO are queued while the IMS is in this state. Queued commands are processed after the online change is committed or aborted. If the type-1 command is entered from the system console, the WTOR does not appear until this IMS is out of the online change state.
OLCCMT1I	LCL or GBL	Online change commit phase 1 in progress.
		An INITIATE OLC PHASE(COMMIT) command or /MODIFY COMMIT command is entered. Online change commit phase 1 is in progress either locally for this IMS or globally for all the IMS systems in the IMSplex.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
OLCCMT2C	LCL or GBL	Online change commit phase 2 completed. An INITIATE OLC PHASE(COMMIT) command or a /MODIFY COMMIT command is entered. Online change commit phase completed either locally for this IMS or globally for all the IMS systems in the IMSplex. Type-1 commands from the system console, an IMS terminal, or the MTO are queued while the IMS is in this state. Queued commands are processed after the online change is committed or aborted. If the type-1 command is entered from the system console, the WTOR does not appear until this IMS is out of the online change state.
OLCCMT2F	LCL	Online change commit phase 2 failed. An INITIATE OLC PHASE(COMMIT) command or a /MODIFY COMMIT command is entered. Online change commit phase 2 failed locally for this IMS. This IMS may be stuck in an online change state, where the TERMINATE OLC command or /MODIFY ABORT command does not work. If that is the case, cancel the IMS and warm start IMS. This IMS can warm start, since it successfully participated in the online change except for commit phase 2.
OLCCMT2I	LCL or GBL	Online change commit phase 2 in progress. An INITIATE OLC PHASE(COMMIT) command or a /MODIFY COMMIT command is entered. Online change commit phase 2 is in progress either locally for this IMS or globally for all the IMS systems in the IMSplex.
OLCCMT3C	GBL	Online change commit phase 3 completed. An INITIATE OLC PHASE(COMMIT) command is entered. Online change commit phase 3 is completed globally on the other IMS systems except for the master. The COMMIT master still needs to perform commit phase 3 locally. The online change is committed, but commit phase 3 is still needed to clean up the online change information on all the IMS systems.
OLCCMT3F	GBL	Online change commit phase 3 failed. An INITIATE OLC PHASE(COMMIT) command is entered. Online change commit phase 3 failed globally on the other IMS systems. The master skips attempting to perform commit phase 3 locally and exits with an error, leaving itself in a global online change state. The other IMS systems may or may not have actually completed commit phase 3. Issue another COMMIT command to the previous COMMIT command master to complete the online change.
OLCCMT3I	LCL or GBL	Online change commit phase 3 is in progress. An INITIATE OLC PHASE(COMMIT) command is entered. Online change commit phase 3 is in progress either locally for this IMS or globally for all the IMS systems in the IMSplex. The online change is committed, but commit phase 3 is needed to clean up online change information on all the IMS systems.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
OLCMACB	GBL	Member online change for ACBLIB in progress.
OLCMSTR	GBL	Online change phase master. An INITIATE OLC PHASE(PREPARE), an INITIATE OLC PHASE(COMMIT), or a TERMINATE OLC command is entered. This IMS is the master of the online change phase currently in progress, either prepare, commit, or terminate. A different IMS may be master of each phase of online change.
OLCPREPC	LCL or GBL	Online change prepare phase completed. An INITIATE OLC PHASE(PREPARE) command or a /MODIFY PREPARE command is entered. Online change prepare phase completed locally for this IMS or globally for all the IMS systems in the IMSplex.
OLCPREPF	LCL	Online change prepare phase failed. An INITIATE OLC PHASE(PREPARE) command is entered. Online change prepare phase failed locally for this IMS. A TERMINATE OLC is required to delete the MWA created for the online change and also to delete the online change process that was initiated with RM.
OLCPREPI	LCL or GBL	Online change prepare phase in progress. An INITIATE OLC PHASE(PREPARE) command or a /MODIFY PREPARE command is entered. Online change prepare phase is in progress locally for this IMS or globally for all the IMS systems in the IMSplex.
OLCTERMC	GBL	Online change terminate completed. A TERMINATE OLC command was entered. Online change termination is completed for the IMS systems in the IMSplex.
OLCTERMF	LCL	TERMINATE FAILED: A TERMINATE OLC command is directed to the IMS that is not in an online change state. An MWA is created to coordinate the TERMINATE OLC command. The TERMINATE OLC command fails because of an RM, SCI, or CQS error and the MWA is set to a 'Terminate Failed' state. A subsequent TERMINATE OLC command is required to delete the MWA in this state.

Table 110. Status for QUERY MEMBER command (continued)

Status	Scope	Meaning
OLCTERMI	GBL	Online change terminate in progress. A TERMINATE OLC command is entered. Online change termination is in progress for the IMS systems in the IMSplex. Online change termination aborts the online change. If all of the IMS systems are in an online change prepare state, TERMINATE OLC aborts the online change and removes all of the IMS systems from the online change state. If an error occurs before the OLCSTAT data set is updated, then TERMINATE OLC aborts the online change. The online change abort phase is performed on the IMS systems where abort is needed. All of the IMS systems are removed from the online change state.
UPDPSB	LCL	Online change to update PSBs on a subset of systems in the IMSplex is in progress. An IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) command was entered and is being processed.
RFSHPSB	LCL	Online change to update PSBs on a subset of systems in the IMSplex is in progress. An IMPORT DEFN SOURCE(CATALOG) OPTION(REFRESHPSB) command was entered and is being processed.
SECCMD	LCL	Eligible IMS commands issued from the master terminal, and their command responses, are sent to the secondary master when the command /SMC MASTER ON is issued.
SECCMDT	LCL	Eligible IMS commands that are issued from terminals other than the master terminal, and their command responses, are sent to the secondary master when the command /SMC TERMINAL ON is issued.
SECMSG	LCL	IMS system messages are sent to the secondary master when the command /SMC MESSAGE ON is issued.
XRFALT	LCL	XRF alternate system.

Output fields

The following table shows information about the QUERY MEMBER TYPE(IMS) output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated.

error appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning

Provides a brief description of the output field.

Table 111. Output fields for QUERY MEMBER command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	N/A	Completion code for the line of output. The completion code is always returned.
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. IMS identifier of IMS that built the output. The IMS identifier is always returned.
LATTR	LclAttr	ATTRIB	LCL	Local IMS attributes. See Table 109 on page 327 for more information.
MODI	ModId	MODID	LCL	Online change modify ID, which is increased by 1 for each online change.
LSTT	LclStat	STATUS	LCL	Local IMS status. See Table 110 on page 327 for more information.
STT	Status	STATUS	GBL	Global IMS status. See Table 110 on page 327 for more information.
TYP	Type	TYPE	LCL	IMSplex member type. The IMSplex member type can be IMS, for the IMS address space.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY MEMBER command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 112. Return and reason codes for QUERY MEMBER command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY MEMBER TYPE(IMS) command completed successfully.
X'00000014'	X'00005004'	The QUERY MEMBER command failed because a DFSOCMD response buffer could not be obtained.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual member.

The following table contains the completion code that can be returned on a QUERY MEMBER command.

Table 113. Completion codes for QUERY MEMBER

Completion code	Meaning
0	The QUERY MEMBER TYPE(IMS) command completed successfully for this IMS.

Examples

The following are examples of the QUERY MEMBER command:

Example 1 for QUERY MEMBER TYPE (IMS) command

TSO SPOC input:

```
QRY MEMBER TYPE(IMS) SHOW(ALL)
```

TSO SPOC output:

```
MbrName    CC Type  Status LclAttr LclStat    ModId
SYS3        0 IMS                                  1
```

OM API input:

```
CMD(QRY MEMBER TYPE(IMS) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 16:32:12.998765</statime>
<stotime>2002.163 16:32:12.999775</stotime>
<staseq>B7C4B78AFD86D562</staseq>
<stoseq>B7C4B78AFDC5FA80</stoseq>
<rqsttkn1>USRT002 10093212</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>MEMBER   </kwd>
<input>QRY MEMBER TYPE(IMS) SHOW(ALL)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"   llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"    llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT" align="right" />
<hdr slbl="TYP"   llbl="Type"    scope="LCL" sort="n" key="0" scroll="yes" len="5"
dtype="CHAR" align="left" />
<hdr slbl="STT"   llbl="Status"  scope="GBL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LATTR" llbl="LclAttr" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LSTT"  llbl="LclStat" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="MODI"  llbl="ModId"   scope="LCL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" /></cmdrsphdr>
```

```
<cmdrspdata>
<rsp>MBR(SYS3   ) CC(  0) TYP(IMS) MODI(       1) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS member SYS3 is active in the IMSplex.

Example 2 for QUERY MEMBER TYPE(IMS) command

TSO SPOC input:
```
QRY MEMBER TYPE(IMS) SHOW(ALL)
```

TSO SPOC output:
```
MbrName    CC Type  Status LclAttr LclStat    ModId
SYS3        0 IMS                   LEOPT          1
```

OM API input:
```
CMD(QRY MEMBER TYPE(IMS) SHOW(ALL))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 16:42:10.557119</statime>
<stotime>2002.163 16:42:10.557503</stotime>
<staseq>B7C4B9C4DDCBF28D</staseq>
<stoseq>B7C4B9C4DDE3F02D</stoseq>
<rqsttkn1>USRT002 10094210</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>QRY </verb>
<kwd>MEMBER </kwd>
<input>QRY MEMBER TYPE(IMS) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"   llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"    llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT" align="right" />
<hdr slbl="TYP"   llbl="Type"    scope="LCL" sort="n" key="0" scroll="yes" len="5"
dtype="CHAR" align="left" />
<hdr slbl="STT"   llbl="Status"  scope="GBL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LATTR" llbl="LclAttr" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LSTT"  llbl="LclStat" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="MODI"  llbl="ModId"   scope="LCL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3   ) CC(  0) TYP(IMS) LSTT(LEOPT) MODI(       1) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS member SYS3 is active in the IMSplex. The local status shows
that LE runtime option overrides are enabled.

Example 3 for QUERY MEMBER TYPE(IMS) command

TSO SPOC input:

```
QRY MEMBER TYPE(IMS) SHOW(ALL)
```

TSO SPOC output:

```
 Response for: QUERY MEMBER TYPE(IMS) SHOW(ALL)
MbrName   CC Type  Status LclAttr      LclStat    ModId
IMS2       0 IMS           SHAREDQ,GBLOLC              1
IMS3       0 IMS           SHAREDQ,GBLOLC              1
SYS3       0 IMS           SHAREDQ,GBLOLC              1
```

OM API input:

```
CMD (QRY MEMBER TYPE(IMS) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 15:13:05.255654</statime>
<stotime>2002.163 15:13:06.479196</stotime>
<staseq>B7C4A5DB308E6544</staseq>
<stoseq>B7C4A5DC5B45C385</stoseq>
<rqsttkn1>USRT011 10081304</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS3 </master>
<userid>USRT011</userid>
<verb>QRY</verb>
<kwd>MEMBER</kwd>
<input>QUERY MEMBER TYPE(IMS) SHOW(ALL)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"   llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"    llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT"  align="right" />
<hdr slbl="TYP"   llbl="Type"    scope="LCL" sort="n" key="0" scroll="yes" len="5"
dtype="CHAR" align="left" />
<hdr slbl="STT"   llbl="Status"  scope="GBL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LATTR" llbl="LclAttr" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" />
<hdr slbl="LSTT"  llbl="LclStat" scope="LCL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left"/>
<hdr slbl="MODI"  llbl="ModId"   scope="LCL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right"/>
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS3    ) CC(   0) TYP(IMS) LATTR(SHAREDQ,GBLOLC) MODI(      1) </rsp>
<rsp>MBR(IMS2    ) CC(   0) TYP(IMS) LATTR(SHAREDQ,GBLOLC) MODI(      1) </rsp>
<rsp>MBR(SYS3    ) CC(   0) TYP(IMS) LATTR(SHAREDQ,GBLOLC) MODI(      1) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS members SYS3, IMS2, and IMS3 are active in the IMSplex. All three IMS systems have shared queues enabled and global online change enabled.

Example 4 for QUERY MEMBER TYPE(IMS) SHOW(STATUS) command

TSO SPOC input:

```
QUERY MEMBER TYPE(IMS) SHOW(STATUS)
```

TSO SPOC output:

```
MbrName   CC  Status                                    LclStat
IMS1       0                                            OLCCMT1C,OLCCMT2I
IMS2       0                                            OLCCMT1C,OLCCMT2I
IMS2       0  OLCMSTR,OLCMACB,OLCCMT1C,OLCCMT2I
IMS3       0                                            OLCCMT1C,OLCCMT2I
```

OM API input:

```
CMD (QUERY MEMBER TYPE(IMS) SHOW(STATUS))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2006.275 18:28:38.671513</statime>
<stotime>2006.275 18:28:38.683275</stotime>
<staseq>BF7E87571A0999E4</staseq>
<stoseq>BF7E87571CE8B96E</stoseq>
<rqsttkn1>USRT001 10112838</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>MEMBER          </kwd>
<input>QUERY MEMBER TYPE(IMS) SHOW(STATUS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="GBL" sort="n" key="0" scroll="yes"
 len="*" dtype="CHAR" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS2    ) CC(    0) STT(OLCMACB,OLCPREPC) </rsp>
<rsp>MBR(IMS2    ) CC(    0) LSTT(OLCPREPC,SECCMD,SECMSG) </rsp>
<rsp>MBR(IMS1    ) CC(    0) LSTT(OLCPREPC,SECCMD,SECMSG) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS2 is the command master for the ACB member online change process. All IMS systems finished online change commit phase 1. The member OLC is currently in commit phase 2. The online change process cannot be terminated using a TERM OLC command. All updated members will be committed to the active ACBLIB.

Example 5 for QUERY MEMBER TYPE(IMS) SHOW(ATTRIB) command

In the following example, the QUERY MEMBER TYPE(IMS) SHOW(ATTRIB) command is issued to determine if dynamic definition is enabled for MSC in the IMS system IMS1. In the IMS system, dynamic definition is enabled for both MSC and MODBLKS resources. The value of NO_STM indicates that sysplex terminal management is not enabled in the IMS system.

TSO SPOC input:

```
QUERY MEMBER TYPE(IMS) SHOW(ATTRIB)
```

TSO SPOC output:

```
MbrName    CC LclAttr
IMS1        0  NO-STM,DYNMODBLKS,DYNMSC
```

OM API input:

```
CMD (QUERY MEMBER TYPE(IMS) SHOW(ATTRIB))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2014.132 20:55:40.440662</statime>
<stotime>2014.132 20:55:40.440903</stotime>
<staseq>CD24A74424456F95</staseq>
<stoseq>CD24A74424547C95</stoseq>
<rqsttkn1>USRT011 10135540</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MEMBER          </kwd>
<input>QUERY MEMBER TYPE(IMS) SHOW(ATTRIB) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LATTR" llbl="LclAttr" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS1    ) CC(   0) LATTR(NO-STM,DYNMODBLKS,DYNMSC) </rsp>
</cmdrspdata>
</imsout>
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY MSLINK command

Use the QUERY MSLINK command to query information about the definition and status of specified logical links.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 346
- "Usage notes" on page 353
- "Equivalent IMS type-1 commands" on page 353
- "Output fields" on page 354
- "Return, reason, and completion codes" on page 365
- Examples

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY MSLINK command and keywords can be issued.

Table 114. Valid environments for the QUERY MSLINK command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
QUERY MSLINK	X		X
NAME	X		X
BANDWIDTH	X		X
SHOW	X		X
STATUS	X		X

Syntax

```
►►──┬─QUERY─┬──MSLINK──┬──────────NAME(*)────────────────────────►
    └─QRY───┘          │                    ,                    │
                       │              ┌──────────────┐           │
                       └─NAME──(──────┬─linkname─────┬────)──────┘
                                      └─linkname*────┘
```

```
                                                                              ►◄
►──┬─ BANDWIDTH(─┬─ON──┬─)──────────────────────────────────────┬──
   │             └─OFF─┘                                         │
   │               ┌──────────,──────────┐                      │
   ├─ SHOW(─▼─┬─AFFIN──────┬─)─┤ A ├─┬──────────────────────────┤
   │          ├─ALL────────┤       └─ BANDWIDTH(─┬─ON──┬─)─┘     │
   │          ├─ASR────────┤                     └─OFF─┘         │
   │          ├─BACKUP──────┤                                    │
   │          ├─BANDWIDTH───┤                                    │
   │          ├─BUFSIZE─────┤                                    │
   │          ├─CID─────────┤                                    │
   │          ├─COUNT───────┤                                    │
   │          ├─DEFN────────┤                                    │
   │          ├─DEFNTYPE────┤                                    │
   │          ├─GLOBAL,DEFN─┤                                    │
   │          ├─IMSID───────┤                                    │
   │          ├─LOCAL,DEFN──┤                                    │
   │          ├─MODETBL─────┤                                    │
   │          ├─MSPLINK─────┤                                    │
   │          ├─PARTNER─────┤                                    │
   │          ├─STATUS──────┤                                    │
   │          ├─SYNCOPT─────┤                                    │
   │          └─TIMESTAMP───┘                                    │
   ├─ SHOW(EXPORTNEEDED) ────────────────────────────────────────┤
   ├─ SHOW(MSNAME) ──────────────────────────────────────────────┤
   ├─ SHOW(STATISTICS)─┤ A ├─┬────────────────────────────┬──────┤
   │                         └─ BANDWIDTH(─┬─ON──┬─)─┘            │
   │                                        └─OFF─┘              │
   └────┤ A ├───────────────────────────────────────────────────┘
```

A:

```
├──┬──────────────────────────────────┬──┤
   │            ┌──────────,──────┐    │
   └─ STATUS(─▼─┬─ACTIVE───┬─)─┘
               ├─ASR───────┤
               ├─COLDSESS──┤
               ├─ERE───────┤
               ├─FORCSESS──┤
               ├─IDLE──────┤
               ├─INOP──────┤
               ├─LOST──────┤
               ├─NOTIDLE───┤
               ├─NOTOPEN───┤
               ├─NRE───────┤
               ├─PRI───────┤
               ├─RESYNC────┤
               ├─SEC───────┤
               ├─STOCOMM───┤
               ├─SYNCSESS──┤
               ├─TKOTRA────┤
               └─TRACE─────┘
```

Keywords

The following keywords are valid for the QUERY MSLINK command:

BANDWIDTH()
This filter selects logical links that are either using, or not using, bandwidth mode.

When you specify the BANDWIDTH keyword, the output returned includes the bandwidth mode, even if you do not specify SHOW(BANDWIDTH).

ON Selects only logical links that are using bandwidth mode.

OFF
Selects only logical links that are not using bandwidth mode.

NAME()
Specifies the 1- to 8-character names of the logical links displayed. NAME(*) indicates that the command is to be applied to ALL the links in the system. Wildcards (*) can be specified in the name. The default is NAME(*) which returns all MSLINK resources. The logical link number, assigned during system generation, cannot be specified. Instead, the logical link name must be specified. The logical link number is returned as command output.

SHOW()
Specifies the MSLINK output fields to be returned. If you do not specify the SHOW() parameter, only the logical link names and associated logical link numbers are returned.

The MSLINK logical link name is always returned along with the name of the IMS that created the output for the link and the completion code. The filters supported with the SHOW parameter are:

AFFIN
In an IMS system that uses either TCP/IP or VTAM generic resources, AFFIN identifies the specific IMS system in the generic resources group with which the specified logical link has an affinity. For TCP/IP generic resources AFFIN displays the IMS ID of the IMS system. For VTAM generic resources, AFFIN displays the APPLID name of the IMS system.

ALL
Includes all of the information in the other SHOW parameters, except for MSNAME.

This filter displays the values of the ASR and SYNCOPT attributes as statuses, not as attributes

ASR
Specifies the Automatic Session Restart (ASR) when the Session Outage Notification (SON) facility value is returned.

ASR applies only to logical links that are assigned to VTAM physical links when both sides of the link are using the ASR option.

If SHOW(DEFN,ASR) is specified, the local and repository ASR values are returned. If SHOW(STATUS) is specified, the ASR value is shown as a status.

ASR must be specified with DEFN, otherwise the command is rejected.

BACKUP
For XRF-capable IMS systems, displays how the automatic restart of TCP/IP and VTAM links after an XRF takeover are controlled.

The specification of BACKUP in the definition of a logical link overrides any switching options that are specified in the definition of the associated physical link. Use only if HSB=YES is specified on the IMSCTRL macro.

BACKUP is an MSC physical link attribute that you can set by using the CREATE MSPLINK, CREATE MSLINK, or UPDATE MSPLINK command. You can display the physical link BACKUP value by using the QUERY MSPLINK and QUERY MSLINK commands.

Important: Even though BACKUP is a physical link attribute, the CREATE MSLINK NAME(*linkname*) SET(MSPLINK(*msplinkname*)) command can also change the BACKUP value of the specified physical link. You can use the QUERY MSPLINK command to display the updated BACKUP value. You can also use the QUERY MSLINK command and specify the MSC logical links, including the newly created logical link, that are assigned to the physical link to display the updated BACKUP value.

The displayed value can be a numeric integer from 1 to 7, inclusive, which is the priority for reestablishing the session that has been set. When either the BACKUP keyword is omitted or no value is specified for BACKUP in the definition of either the physical or logical link, the default is 4. NO means that the session recovery of the physical link at takeover has been suppressed.

Although BACKUP displays the priority of the order in which IMS restarts TCP/IP and VTAM links, the active requests might be completed in any order because of variables such as internal VTAM conflicts and pacing, TCP/IP network traffic, and so on.

BANDWIDTH
Displays whether the link is using bandwidth mode.

BUFSIZE
Displays the current input and output buffer size for the logical links.

CID
Displays the VTAM Communication Identifications (CIDs) for the logical links.

COUNT
Displays the number of messages sent to and received from the specified logical links.

DEFN
Specifies that the resource definitions are to be returned.

The MSC logical link attributes that can be returned are: ASR, BACKUP, BANDWIDTH, BUFSIZE, MODETBL, MSPLINK, PARTNER, SYNCOPT, the repository create and update time stamps, and the IMS runtime create and update time stamps.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition that is obtained from the repository. This response line displays the attributes of the generic resource definition. When

SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with AFFIN, CID, COUNT, DEFNTYPE, MSNAME, STATISTICS or STATUS. The LAffin, LCID, LDefnType, LRecdCnt, LSentCnt, LSIDL, LSIDR and LLclStat, which are returned on the QRY MSLINK SHOW(ALL) command, are not returned with SHOW(DEFN).
- SHOW(DEFN) cannot be specified with the filter BANDWIDTH or STATUS.
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).

Resource definitions that are stored in the repository are used to determine the response lines with the repository information, and the runtime resource definitions are used to determine the response lines with the IMS runtime resource information.

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

DEFNTYPE

Displays the definition type that the resource was defined with.

EXPORTNEEDED

Specifies that only the MSC logical link resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command. SHOW(EXPORTNEEDED) behaves like a filter by only displaying resources that need to be exported, as if QUERY MSLINK EXPORTNEEDED(Y) were supported.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the MSC logical links that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each MSC logical link name specified on the command. If the specified MSC logical link does not need to be exported to the repository, an N is returned in the ExportNeeded column.

Restrictions:
- SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords.
- SHOW(EXPORTNEEDED) is valid only when IMS is enabled with dynamic resource definition for MSC resources (MSCRSCS=DYN) and the IMSRSC repository.
- The QUERY MSLINK SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources that have not been exported to the IMSRSC repository, you can issue the /CHECKPOINT command to automatically write the resources to the repository so that they are available for the next IMS cold start.

GLOBAL

For output fields that have both local and global values, this option returns only the global values. GLOBAL must be specified with DEFN, otherwise the command is rejected. If used with another SHOW keyword to request a specific output field, this option requests that only the global value of the specified output field is returned.

SHOW(DEFN,GLOBAL,LOCAL) is equivalent to SHOW(DEFN).

IMSID

Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only when the command master IMS is enabled to use the repository.

IMSID cannot be specified with any other SHOW keywords other than DEFN or GLOBAL.

When SHOW(DEFN,IMSID) is specified, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN keyword, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

LOCAL

For output fields that have both local and global values, this option returns only the local values. LOCAL must be specified with DEFN, otherwise the command is rejected. If used with another SHOW keyword to request a specific output field, this option requests that only the local value of the specified output field is returned. Local output is returned by each IMS that processes the command.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

SHOW(DEFN,GLOBAL,LOCAL) is equivalent to SHOW(DEFN).

MODETBL

Displays the mode table names that are associated with the specified links. The output includes DEF MODETBL and ACT MODETBL. DEF MODETBL is the default mode table name that is set by system definition or by issuing an UPDATE command. The name can be overridden with the UPDATE command or, for non-IMS session initiations, the LOGON exit. ACT MODETBL is the mode table name used to initiate the session. This name is displayed only while the session is active. The field is blank at normal session termination.

MSNAME

Displays the MSNAMEs assigned to the specified logical link, and their remote and local SYSIDs. When SHOW(MSNAME) is specified, STATUS() is invalid. MSNAME is mutually exclusive of all other SHOW filters.

MSPLINK

Displays the physical link to which this logical link is assigned.

PARTNER

Displays the current partner ID for this logical link.

STATISTICS

Displays the statistics for the logical link.

MSC link statistics are displayed by type-2 command QUERY MSLINK NAME(linkname) SHOW(STATISTICS). The SHOW(STATISTICS) keyword also displays the statistics reset mode, RESET,CHKPT or NORESET,CHKPT. Link statistics are not displayed with the SHOW(ALL) keyword.

I/O statistics are recorded only in bandwidth mode.

Additional I/O statistics are recorded for TCP/IP links, including the high, low, and total send I/O times for both the local and remote SCI, IMS Connect, and TCP/IP components.

Important: To be accurate, the I/O statistics for CTC, TCP/IP, and VTAM links require synchronized clocks on the local and remote IMS installations.

If the clocks are not synchronized, elapsed I/O times can be calculated as negative values on one of the send I/O paths. The QUERY MSLINK command does not display negative values for elapsed I/O times.

IMS cannot determine if clocks are synchronized. Unless both partner IMS systems are on the same processor or are within the same sysplex, clocks are typically synchronized using an external method, such as an External Time Reference (ETR) hardware device.

MTM link partners are always on the same processor, so the timers are always synchronized. Partner IMS systems within the same SYSPLEX also have synchronized timers.

STATUS

Displays the logical link status.

SYNCOPT

For logical links that are assigned to TCP/IP or VTAM physical links only, displays the session resynchronization options for the logical link.

SYNCSESS

Indicates that session initiation is to be completed only if session resynchronization is successful. Successful session resynchronization occurs when the message sequence numbers of the two logical units in session agree, or when the sequence number of the sender is not less than the sequence number of the receiver.

FORCSESS

Forces the session initiation to be completed regardless of whether session resynchronization is successful.

If SHOW(DEFN,SYNCOPT) is specified, the local and repository synchronization option values are returned. If SHOW(STATUS) is specified, the synchronization option value is shown as a status.

SYNCOPT must be specified with DEFN, otherwise the command is rejected.

SYNCOPT only applies to the global SYNCOPT definition in the IMSRSC repository. Use the STATUS(SYNCSESS) filter to display the local SYNCOPT definition.

TIMESTAMP

Displays the creation time (TimeCreate), last update time (TimeUpdate), last access time (TimeAccess), and last import time (TimeImport) time stamps are returned. The time is returned in local time in the format `YYYY.JJJ HH:MM:SS:TH`:

YYYY

The year

JJJ

The Julian day (001 - 365)

HH The hour (01 - 24)

MM The minute (00 - 59)

SS The seconds (00 - 59)

TH The tenths and hundredths of a second (00 - 99)

STATUS()

Displays logical links that possess at least one of the specified status. When you specify the STATUS keyword, the output returned includes the logical link status even if you do not specify SHOW(STATUS). The STATUS keyword cannot be specified with SHOW(MSNAME).

ACTIVE

Indicates, for the logical link, that link startup processing is complete and the line is available for message transfer.

ASR

Indicates the current automatic session restart designation as part of the Session Outage Notification.

COLDSESS

Indicates that for the logical link, startup processing is not complete.

ERE

Indicates that for the logical link, startup processing is not complete.

However, when started, emergency restart synchronization is performed, because the previous link shutdown was either not normal or an IMS emergency restart was performed.

FORCSESS
Indicates that message resynchronization is attempted when the logical link is started. Even if resynchronization fails, the link is still started.

IDLE
Indicates that no activity is in progress for the logical link.

INOP
Indicates an inoperable link.

LOST
Indicates that the VTAM LOSTERM EXIT has been scheduled for this link.

NOTIDLE
Indicates that the link is waiting for the completion of a synchronous event. The status can be NOTIDLE-A, NOTIDLE-B, or NOTIDLE-C*xx*, where *xx* represents the value of the access method operation code.

> **NOTIDLE-A**
> Indicates a status of NOTIDLE-POST, which means that an event has completed but the link has not been dispatched to process it.
>
> **NOTIDLE-B**
> Indicates a status of NOTIDLE-IWAIT, which means the link is waiting for completion of internal I/O.
>
> **NOTIDLE-C*xx***
> Indicates a status of NOTIDLE-TP WAIT, which means the link is waiting for completion of a TP access method request. The two characters *xx* indicate the value of the access method operation code. Blanks appear if the link is VTAM.

NOTOPEN
Indicates that the link is not in open status.

NRE
Indicates that startup processing is not complete for the logical link. Normal restart synchronization is performed when the logical link starts, because the previous link shutdown or IMS restart was normal.

PRI
Indicates that the link is the primary partner of the MSC VTAM session.

RESYNC
Indicates that the positive acknowledgment for an IMS recoverable output message was not received when the MSC session was terminated.

SEC
Indicates that the link is the primary partner of the MSC VTAM session.

STOCOMM
Indicates that the link was stopped for communications.

SYNCSESS
Indicates that message resynchronization is attempted when the logical link is started, and that the link is started only if resynchronization is successful.

TKOTRA

Indicates that the logical link in an XRF session is traced only during takeover, to help diagnose XRF link switch problems.

TRACE

Indicates that the logical link is traced.

Usage notes

You can issue this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

The CREATE MSLINK NAME(*mslinkname*) command specified with SET(MSPLINK(*msplinkname*)) is rejected if at least one logical link attribute specified in the command (ASR, BACKUP, MODETBL, or SYNCOPT) conflicts with the physical link type. However, after the logical link is created, it can be reassigned to any type of physical link with an UPDATE MSLINK command or an /MSASSIGN command. If a logical link defined with attributes applicable to specific physical link types is reassigned to a different physical link type that does not support those attributes, they are ignored when the link is started.

If you want to display information about resource definitions, specify SHOW(DEFN). If you want to know which IMS systems have the resource defined and also know the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource defined, specify SHOW(IMSID).

If QUERY MSLINK is specified with SHOW(DEFN,BACKUP) and the local BACKUP value is NO, but the repository value is a number from 1 to 7, this is because the local IMS is not XRF capable. If QUERY MSLINK is specified with SHOW(DEFN,BACKUP) and the local BACKUP value is a number from 1 to 7, but the repository value is NO, this is because the local IMS was not XRF capable when it last exported the MSLINK, but it is XRF capable now.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY MSLINK command and the type-1 IMS commands that perform similar functions.

Table 115. Type-1 equivalents for the QUERY MSLINK command

QUERY MSLINK command	Similar IMS type-1 command
QUERY MSLINK NAME(linkname ǀ linkname*) SHOW(ALL)	/DIS LINK link# ǀ ALL
QUERY MSLINK NAME(linkname ǀ linkname*) SHOW(MODETBL)	/DIS LINK link# ǀ ALL MODE
QUERY MSLINK NAME(linkname ǀ linkname*) SHOW(MSPLINKǀMSNAME)	/DIS ASSIGNMENT LINK link# ǀ ALL
QUERY MSLINK NAME(linkname ǀ linkname*) SHOW(AFFIN)	/DIS AFFIN LINK *link#* ǀ ALL /DIS AFFIN NODE *nodename*

Output fields

The query output will display a series of headers and values, including the MSLINK name and number, IMSID that processed the command, and the command condition code (CC) of the command.

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Following are the statistics headers on the QUERY MSLINK command, and their meanings. All count fields with times and rates are in seconds. For example, 1.3 is equal to 1.3 seconds. .000003 is equal to 3 microseconds. The other fields such as byte or message counts are in decimal numbers. Rates are in units per second (for example, CHKW_RATE means check write calls per second).

Table 116. Output fields for the QUERY MSLINK command

Short label	Long label	Keyword	Scope	Meaning
AFFIN	LAffin	AFFIN	N/A	In an IMS system that uses either TCP/IP or VTAM generic resources, this field identifies the IMS system in the generic resource group with which the link has affinity, if any. For TCP/IP generic resources AFFIN displays the IMS ID of the IMS system. For VTAM generic resources, AFFIN displays the APPLID name of the IMS system.
AMRS	LAvg_Msg_Rec_SZ	STATISTICS	N/A	Average message size received (type 01/03 message record).
AMSS	LAvg_Msg_Send_SZ	STATISTICS	N/A	Average message size sent (type 01/03 message record).
AMTB	LActMdtbl	MODETBL	N/A	Active VTAM logon mode table entry for the logical link.
APT	LAvg_Proc_Time	STATISTICS	N/A	Average link processing time per dispatch calculated by dividing TPT by TDN.
AQGT	LAvg_Qget_Time	STATISTICS	N/A	The average time for QMGR calls (GU or DEQ) to process a send message.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
AQPT	LAvg_Qput_Time	STATISTICS	N/A	The average QMGR call (ISRT or ENQ) to process a received message.
ASR	LASR	ASR	LCL	Indicates the current automatic session restart designation as part of the Session Outage Notification. The output is returned from the local IMS.
BANDW	LBandwidth	BANDW	N/A	The current usage of bandwidth mode.
BKUP	LBackUp	BACKUP	N/A	Automatic restart of TCP/IP and VTAM links after an XRF takeover.
BSR	LSend_MsgByte_Rate	STATISTICS	N/A	Message bytes sent per second from MSC link. The send time per each message sent is the time from Get Unique (GU) to get the message to send, to dequeue the message when the response is received. The link idle time in between sending messages is ignored.
BUFSZ	LBufSize	BUFSIZE	N/A	Input and output buffer size for the logical link.
CID	LCID	CID	N/A	VTAM Communication Identification.
CNTR	LRecdCnt	COUNT	N/A	Number of messages received from the specified link.
CNTS	LSentCnt	COUNT	N/A	Number of messages sent on the specified link.
CWION	LChkwIO_CT	STATISTICS	N/A	Number of the logger check writes that resulted in a write (I/O) of the get unique (GU) or insert (ISRT) call to the logger write ahead data set (WADS).
CWN	LChkw_CT	STATISTICS	N/A	Number of logger check writes. Check writes are requests to the logger to insure the message is logged (for example, Message get uniques are logged on the send side and inserts are logged on the receive side).
CWR	LChkw_Rate	STATISTICS	N/A	Logger check write rate calculated by dividing the CWR by the statistics recording time.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
DFNT	LDefnType	DEFNTYPE	N/A	Definition type, which can be one of the following: **CREATE** Defined by CREATE LTERM command. **MSCGEN** Defined by system definition in the DFSCLL3x member of the IMS RESLIB. **UPDATE** Defined by system definition in the DFSCLL3x member of the IMS RESLIB, but modified in online memory by a /CHANGE LINK command, an /MSASSIGN LINK command, or an UPDATE command that changed at least one attribute. At the next IMS cold start, the definition reverts back to the definition in the DFSCLL3x member.
DMTB	LDefMdtbl	MODETBL	N/A	Default VTAM logon mode table entry (set by system definition or modified by the UPDATE command).
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the MSC logical link has been exported to the IMSRSC repository. The value can be Y or N.
HLISIOT	LHiLocIconSendIOTime	STATISTICS	N/A	The longest interval of time that the local IMS Connect instance required to process a message from SCI and send it to TCP/IP.
HLSSIOT	LHiLocSciSendIOTime	STATISTICS	N/A	The longest interval of time that the local SCI instance required to process a message from IMS and send it to the local IMS Connect.
HMRS	LHi_Msg_Rec_SZ	STATISTICS	N/A	Largest message size received (type 01/03 message record).
HMSS	LHi_Msg_Send_SZ	STATISTICS	N/A	Largest message size sent (type 01/03 message record).
HPT	LHi_Proc_Time	STATISTICS	N/A	The highest (longest) time the link was dispatched to process.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
HQGT	LHi_Qget_Time	STATISTICS	N/A	The highest (longest) QMGR call (GU or DEQ) to process a send message.
HQPT	LHi_Qput_Time	STATISTICS	N/A	The highest (longest) QMGR call (ISRT or ENQ) to process a received message.
HRIOT	LHi_RecIO_Time	STATISTICS	N/A	The highest (longest) I/O time to receive a message.
HRISIOT	LHiRmtIconSendIOTime	STATISTICS	N/A	The longest interval of time that the remote IMS Connect instance required to process a message from TCP/IP and send it to the remote SCI.
HRSSIOT	LHiRmtSciSendIOTime	STATISTICS	N/A	The longest interval of time that the remote SCI instance required to process a message from the remote IMS Connect instance and send it to the remote IMS system.
HSIOT	LHi_SentIO_Time	STATISTICS	N/A	The highest (longest) I/O time to send a message.
HTCSIOT	LHiTcpipSendIOTime	STATISTICS	N/A	The longest interval of time that a message required to travel from the local IMS Connect instance to the remote IMS Connect instance on the TCP/IP network.
IMSID	IMSid	IMSID	GBL	Returns from the repository the IMSIDs that have the resource defined.
LINKN	LMSLink#	No keywords, or DEFN, or DEFN,LOCAL	LCL	Logical link number.
LLISIOT	LLowLocIconSendIOTime	STATISTICS	N/A	The shortest interval of time that the local IMS Connect instance required to process a message from SCI and send it to TCP/IP.
LLSSIOT	LLowLocSciSendIOTime	STATISTICS	N/A	The shortest interval of time that the local SCI instance required to process a message from IMS and send it to the local IMS Connect.
LMRS	LLow_Msg_Rec_SZ	STATISTICS	N/A	Smallest message size received (type 01/03 message record).
LMSS	LLow_Msg_Send_SZ	STATISTICS	N/A	Smallest message size sent (type 01/03 message record).
LPT	LLow_Proc_Time	STATISTICS	N/A	Lowest (shortest) time the link was dispatched to process.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
LQGT	LLow_Qget_Time	STATISTICS	N/A	The lowest (shortest) QMGR call (GU or DEQ) to process a send message.
LQPT	LLow_Qput_Time	STATISTICS	N/A	The lowest (shortest) QMGR call (ISRT or ENQ) to process a received message.
LRIOT	LLow_RecIO_Time	STATISTICS	N/A	The lowest (shortest) I/O time to receive a message.
LRISIOT	LLowRmtIconSendIOTime	STATISTICS	N/A	The shortest interval of time that the remote IMS Connect instance required to process a message from TCP/IP and send it to the remote SCI.
LRSSIOT	LLowRmtSciSendIOTime	STATISTICS	N/A	The shortest interval of time that the remote SCI instance required to process a message from the remote IMS Connect instance and send it to the remote IMS system.
LSIOT	LLow_SendIO_Time	STATISTICS	N/A	The lowest (shortest) I/O time to send a message.
LSTT	LLclStat	STATUS	N/A	The current status of the logical link. For the possible status values returned, see the description of the STATUS keyword.
LTCSIOT	LLowTcpipSendIOTime	STATISTICS	N/A	The shortest interval of time that a message required to travel from the local IMS Connect instance to the remote IMS Connect instance on the TCP/IP network.
MSN	LMSName	MSNAME	N/A	MSNAMEs associated with the logical link.
MSP	LMSPLink	MSPLINK	N/A	Physical link to which the logical link is assigned.
MSR	LSend_MsgCT_Rate	STATISTICS	N/A	Messages sent per second by MSC link. The send time per each message sent is the time from Get Unique (GU) to get the message to send, to dequeue the message when the response is received. The link idle time in between sending messages is ignored.
PID	LPID	PARTNER	N/A	Partner ID for the link.
RASR	ASR	ASR and DEFN	GBL	Indicates the current automatic session restart designation as part of the Session Outage Notification. The output is returned from the repository.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
RBANDW	Bandwidth	BANDWIDTH and DEFN	GBL	The current usage of bandwidth mode. The output is returned from the repository.
RBKUP	BackUp	BACKUP and DEFN	GBL	Automatic restart of TCP/IP and VTAM links after an XRF takeover. The output is returned from the repository.
RBR	LRecIO_Byte_Rate	STATISTICS	N/A	Rate in bytes per second being received. This is calculated by dividing the total bytes received (TBRC) by the total send time (TRIOT).
RBUFSZ	BufSize	BUFSIZE and DEFN	GBL	Input and output buffer sizes for logical link. The output is returned from the repository.
RDMTB	DefMdtbl	DEFN and MODETBL	GBL	VTAM logon mode table entry (logon mode name) containing the SNA bind parameters to be used when a session is established for this terminal. The output is returned from the repository.
REPO	Repo	DEFN	GBL	Indicates whether the line displays the stored resource definitions. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
RIOR	LRecIO_Req_Rate	STATISTICS	N/A	The receive I/O requests per second. This is calculated by dividing the total receives (TRN) by the total send I/O time (TRIOT).
RMSP	MSPLink	DEFN and MSPLINK	GBL	Physical link to which the logical link is assigned. The output is returned from the repository.
RMT	LRec_Msg_Time	STATISTICS	N/A	Interval of time between the first and last message received. Used for benchmark testing of a block of messages.
RPID	PID	DEFN and PARTNER	GBL	Partner ID for the link. The output is returned from the repository.
RSYNC	SyncOpt	DEFN and SYNCOPT	GBL	Indicates the session resynchronization options for the logical link. The output is returned from the repository.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
RTMCR	TimeCreate	DEFN	GBL	Create time from the repository. This is the time the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN	GBL	Update time from the repository. This is the time the resource was last updated in the repository.
RSYNC	SyncOpt	DEFN and SYNCOPT	GBL	Indicates the session resynchronization options for the logical link. The output is returned from the repository.
SBR	LSendIO_Byte_Rate	STATISTICS	N/A	Rate in bytes per second being sent. This is calculated by dividing the total bytes sent (TBSC) by the total send time (TSIOT).
SIDL	LSIDL	MSNAME	N/A	Local system identification of the associated MSNAME.
SIDR	LSIDR	MSNAME	N/A	Remote system identification of the associated MSNAME.
SIOR	LSendIO_Req_Rate	STATISTICS	N/A	The send I/O requests per second. This is calculated by dividing the total sends (TSN) by the total send I/O time (TSIOT).
SMT	LSend_Msg_Time	STATISTICS	N/A	Interval of time between the first and last message sent. Used for benchmark testing of a block of messages.
SOPT	LOption	STATISTICS	N/A	Statistics reset option = RESET,CHKPT = statistics are reset at each IMS checkpoint or NORESET,CHKPT = statistics are not reset at IMS checkpoints.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
STIM	LStart_Time	STATISTICS	N/A	The start date and local time for the statistics. For option RESET,CHKPT, this will be the last IMS checkpoint time. For option NORESET,CHKPT, this will be when the last UPDATE MSLINK START STATISTICS OPTION (RESET) command was issued. The format is yyyy.ddd hh.mm.ss.tt, where:
				yyyy Four digit year
				ddd Julian day of the year
				hh hour of the day, 00 - 23
				mm minute of the hour, 00 - 59
				ss second of the minute, 00 - 59
				tt tenths of the second, 0 - 9
				hh hundredths of the second, 0 - 9
TBRC	LTot_Byte_Rec_CT	STATISTICS	N/A	Total bytes of data received including message data, response data, and internal message control blocks used for bandwidth mode.
TBSC	LTot_Byte_Send_CT	STATISTICS	N/A	Total bytes of data sent including message data, response data, and internal message control blocks used for bandwidth mode.
TDN	LTot_Disp_CT	STATISTICS	N/A	Number of times the logical link ITASK was dispatched. A typical send or receive message operation will require two dispatches, one to send and one to process the acknowledgment, or one to receive and one to send the acknowledgment.
TLISIOT	LTotLocIconSendIOTime	STATISTICS	N/A	The total amount of time that the local IMS Connect instance required to process all messages from SCI and send them to TCP/IP.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
TLSSIOT	LTotLocSciSendIOTime	STATISTICS	N/A	The total amount of time that the local SCI instance required to process all messages from IMS and send them to the local IMS Connect.
TMAC	LTimeAccess	TIMESTAMP	N/A	The time that the resource was last accessed. This is not yet supported.
TMBRC	LTot_MsgByte_Rec_CT	STATISTICS	N/A	Total count of message bytes of data received. This includes the message prefixes and user data segments (for example, all the data in the type 01/03 log record).
TMBSC	LTot_MsgByte_Send_CT	STATISTICS	N/A	Total count of message bytes of data sent. This includes the message prefixes and user data segments (for example, all the data in the type 01/03 log record).
TMCR	LTimeCreate	TIMESTAMP	N/A	The time that the resource was created with a CREATE MSLINK command, an IMPORT command that creates the MSC logical link, or an IMS cold start. The create time is retained across IMS warm start. The output is returned from the local IMS.
TMIM	LTimeImport	TIMESTAMP	N/A	The time that the resource was last imported, if applicable. The local import timestamp is set to be the IMS cold start time, if the resource was automatically imported from the IMSRSC repository during IMS cold start. The import time is retained across IMS warm start and emergency restart. The output is returned from the local IMS.
TMRN	LTot_Msg_Rec_CT	STATISTICS	N/A	Total number of messages received.
TMSN	LTot_Msg_Send_CT	STATISTICS	N/A	Total count of messages sent.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
TMUP	LTimeUpdate	TIMESTAMP	N/A	The last time the attributes of the runtime resource definition were updated as a result of the UPDATE MSLINK command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
TPT	LTot_Proc_Time	STATISTICS	N/A	Total amount of time the logical link ITASK was dispatched to be processed. This includes wait time while the ITASK was dispatched. This time is in seconds with a resolution to micro seconds.
TQGN	LTot_Qget_CT	STATISTICS	N/A	Total count of QMGR calls issued to get messages off the queue (local or shared queues) for processing. This includes get unique (GU) and dequeue (DEQ) calls.
TQGT	LTot_Qget_Time	STATISTICS	N/A	Total processing time for QMGR calls issued to get messages off the queue (local or shared queues) for processing. This includes get unique (GU) and dequeue (DEQ) calls. Times are calculated from the time the call was issued by the MSC ITASK, to the time the call returned from QMGR.
TQPN	LTot_Qput_CT	STATISTICS	N/A	Total count of QMGR calls issued to put messages to the queue (local or shared queues) for processing. This includes insert (ISRT) and enqueue (ENQ) calls.
TQPT	LTot_Qput_Time	STATISTICS	N/A	Total processing time for QMGR calls issued to put messages to the queue (local or shared queues) for processing. This includes insert (ISRT) and enqueue (ENQ) calls. Times are calculated from the time the call was issued by the MSC ITASK, to the time the call returned from QMGR.

Table 116. Output fields for the QUERY MSLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
TRIOT	LTot_RecIO_Time	STATISTICS	N/A	Total amount of I/O time to receive messages. This is time from which the access method was called to process the receive, to the time the partner IMS access method was called to send the message. This is the time between the access method calls on each side, for the data to cross the media (for example, CTC link, memory, or network).
TRISIOT	LTotRmtIconSendIOTime	STATISTICS	N/A	The total amount of time that the remote IMS Connect instance required to process all messages from TCP/IP and send them to the remote SCI.
TRN	LTot_Req_CT	STATISTICS	N/A	Total count of receive I/O requests received from the access method.
TRSSIOT	LTotRmtSciSendIOTime	STATISTICS	N/A	The total amount of time that the remote SCI instance required to process all messages from the remote IMS Connect instance and send them to the remote IMS system.
TSIOT	LTot_SendIO_Time	STATISTICS	N/A	Total amount of I/O time to send messages. This is the time from which the access method was called to process the send, to the time the partner IMS ITASK was dispatched upon receipt of the message. This is the time between the access method calls on each side, for the data to cross the media (for example, CTC link, memory, or network).
TSN	LTot_Send_CT	STATISTICS	N/A	Total count of send I/O requests issued to the access method.
TTCSIOT	LTotTcpipSendIOTime	STATISTICS	N/A	The sum total of the amount of time that all messages required to travel from the local IMS Connect instance to the remote IMS Connect instance on the TCP/IP network.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY MSLINK command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 117. Return and reason codes for the QUERY MSLINK command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY MSLINK command completed successfully.
X'00000004'	X'00001010'	No resources were found to be returned because of one of the following reasons: • The resource names specified might be invalid. • There were no resources that match the filter specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002040'	More than one filter value is specified on the QUERY MSLINK command.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. For details on completion codes, see Table 118 on page 367.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. For details on completion codes, see Table 118 on page 367.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.

Table 117. Return and reason codes for the QUERY MSLINK command (continued)

Return code	Reason code	Meaning
X'00000010'	X'0000402D'	Command was not successful because dynamic resource definition for MSC resources is not enabled; that is, MSCRSCS=DYN is not defined in the DFSDFxxx member.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004500'	IMS is not enabled to use the IMSRSC repository.
X'00000010'	X'00004501'	RM is not enabled with the IMSRSC repository.
X'00000010'	X'00004502'	The IMSRSC repository is not available.
X'00000010'	X'00004504'	IMSRSC repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	IMSRSC repository access is denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server (RS) is being shut down.
X'00000010'	X'0000450B'	Repository Server (RS) is not available.
X'00000010'	X'0000450C'	Repository Server (RS) is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields that are related to the IMSRSC repository.
X'00000014'	X'00005004'	The QUERY MSLINK command processing terminated as a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005100'	RM request error.
X'00000014'	X'00005104'	CQS error.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.

Table 117. Return and reason codes for the QUERY MSLINK command (continued)

Return code	Reason code	Meaning
X'00000014'	X'0000551C'	The QUERY MSLINK command is rejected because the resource list is unusable. Contact IBM Software Support with the following information: • The output from the command that caused the error • A dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex • A copy of the IMSRSC repository RID and RMD data sets

Errors that are unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 118. Completion codes for the QUERY MSLINK command

Completion code	Completion code text	Meaning
0		The QUERY MSLINK command completed successfully for the resource.
10	NO RESOURCES FOUND	MSLINK name is invalid, or the specified wildcard parameter does not match any resource names.

Example 1 for QUERY MSLINK command

TSO SPOC input:

```
QUERY MSLINK NAME(LNK12C01,LNK12V01,LNK12M01,LNK12T01) SHOW(ALL)
```

TSO SPOC output:

(screen 1)

```
MSLink    LMSLink# MbrName    CC LMSPLink LCID     LPID LRecdCnt LSentCnt
LNK12C01         2 IMS1        0 PLNK12C           AC          0        0
LNK12M01         3 IMS1        0 PLNK12M           AD          0        0
LNK12T01        22 IMS1        0 PLNK12TA          TA          0        0
LNK12V01         1 IMS1        0 PLNK12V  00000000 AB          0        0
```

(scrolled to the right, screen 2)

```
MSLink    LMSLink# MbrName LDefMdtbl LBufSize LBandwidth LAffin
LNK12C01         2 IMS1                  1024 OFF
LNK12M01         3 IMS1                  1024 OFF
LNK12T01        22 IMS1                 65536 ON
LNK12V01         1 IMS1                  1024 OFF
```

(scrolled to the right, screen 3)

MSLink	LMSLink#	MbrName	LclStat	LBackUp
LNK12C01	2	IMS1	STOCOMM,IDLE,NOTOPEN	
LNK12M01	3	IMS1	STOCOMM,IDLE,COLDSESS	
LNK12T01	22	IMS1	STOCOMM,IDLE,COLDSESS,SYNCSESS	NO
LNK12V01	1	IMS1	STOCOMM,IDLE,COLDSESS,SYNCSESS	NO

(scrolled to the right, screen 4)

MSLink	LMSLink#	MbrName	LTimeCreate
LNK12C01	2	IMS1	2014.282 15:59:39.74
LNK12M01	3	IMS1	2014.282 15:59:39.74
LNK12T01	22	IMS1	2014.282 15:59:39.74
LNK12V01	1	IMS1	2014.282 15:59:39.74

(scrolled to the right, screen 5)

MSLink	LMSLink#	MbrName	LTimeUpdate	LTimeAccess	LDefnType
LNK12C01	2	IMS1			MSCGEN
LNK12M01	3	IMS1			MSCGEN
LNK12T01	22	IMS1			MSCGEN
LNK12V01	1	IMS1			MSCGEN

OM API input:

CMD(QRY MSLINK NAME(LNK12C01,LNK12V01,LNK12M01,LNK12T01) SHOW(ALL))

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2014.283 16:40:17.403520</statime>
<stotime>2014.283 16:40:17.403851</stotime>
<staseq>CDE247CF8E680574</staseq>
<stoseq>CDE247CF8E7CB0F4</stoseq>
<rqsttkn1>USRT011 10094017</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSLINK          </kwd>
<input>QUERY MSLINK NAME(LNK12C01,LNK12V01,LNK12M01,LNK12T01) SHOW(ALL)
 </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSL" llbl="MSLink" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LINKN" llbl="LMSLink#" scope="LCL" sort="a" key="2"
 scroll="no" len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MSP" llbl="LMSPLink" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CID" llbl="LCID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="INT" align="left" skipb="yes" />
<hdr slbl="PID" llbl="LPID" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CNTR" llbl="LRecdCnt" scope="LCL" sort="n" key="0"
```

```
scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CNTS" llbl="LSentCnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="DMTB" llbl="LDefMdtbl" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="AMTB" llbl="LActMdtbl" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="BUFSZ" llbl="LBufSize" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="BANDW" llbl="LBandwidth" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="AFFIN" llbl="LAffin" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="BKUP" llbl="LBackUp" scope="LCL" sort="n" key="0"
 scroll="yes" len="2" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSL(LNK12C01) LINKN(2) MBR(IMS1) CC(   0) MSP(PLNK12C ) PID(AC)
 CNTR(0) CNTS(0) BUFSZ(1024) BANDW(OFF ) LSTT(STOCOMM,IDLE,NOTOPEN)
 TMCR(2014.282 15:59:39.74) DFNT(MSCGEN) </rsp>
<rsp>MSL(LNK12V01) LINKN(1) MBR(IMS1) CC(   0) MSP(PLNK12V )
 CID(00000000) PID(AB) CNTR(0) CNTS(0) BUFSZ(1024) BANDW(OFF )
 LSTT(STOCOMM,IDLE,COLDSESS,SYNCSESS) BKUP(NO) TMCR(2014.282
 15:59:39.74) DFNT(MSCGEN) </rsp>
<rsp>MSL(LNK12M01) LINKN(3) MBR(IMS1) CC(   0) MSP(PLNK12M ) PID(AD)
 CNTR(0) CNTS(0) BUFSZ(1024) BANDW(OFF ) LSTT(STOCOMM,IDLE,COLDSESS)
 TMCR(2014.282 15:59:39.74) DFNT(MSCGEN) </rsp>
<rsp>MSL(LNK12T01) LINKN(22) MBR(IMS1) CC(   0) MSP(PLNK12TA) PID(TA)
 DMTB(        ) CNTR(0) CNTS(0) BUFSZ(65536) BANDW(ON  )
 LSTT(STOCOMM,IDLE,COLDSESS,SYNCSESS) BKUP(NO) TMCR(2014.282
 15:59:39.74) DFNT(MSCGEN) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY MSLINK command is issued for one MSC logical link of each link type. Many of the possible headers are displayed.

Example 2 for QUERY MSLINK command

TSO SPOC input:

```
QUERY MSLINK NAME(LNK12*02) SHOW(MSNAME)
```

TSO SPOC output:

```
MSLink    LMSLink# MbrName   CC LMSName  LSIDR LSIDL
LNK12C02       18 IMS1       0 LINK12C2    92    72
LNK12M02        9 IMS1       0 ELINK126    54    46
LNK12T02       23 IMS1       0 LINK12T2    85    75
LNK12T02       23 IMS1       0 LINK12T3    97    75
LNK12V02       10 IMS1       0 LINK12Y     90   500
LNK12V02       10 IMS1       0 LINK13V    266  2036
LNK12V02       10 IMS1       0 LINK13X     80   300
```

```
|                    LNK12V02        10 IMS1        0 LINK12V1   20    10
|                    LNK12V02        10 IMS1        0 LINK13V1   36    16
|                    LNK12V02        10 IMS1        0 LINK12V   256  1012
```

| OM API input:

| CMD(QRY MSLINK NAME(LNK12*02) SHOW(MSNAME))

| OM API output:


```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2014.283 16:56:50.714836</statime>
<stotime>2014.283 16:56:50.715191</stotime>
<staseq>CDE24B82DA0D46F4</staseq>
<stoseq>CDE24B82DA2370F4</stoseq>
<rqsttkn1>USRT011 10095650</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSLINK          </kwd>
<input>QRY MSLINK NAME(LNK12*02) SHOW(MSNAME) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSL" llbl="MSLink" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LINKN" llbl="LMSLink#" scope="LCL" sort="a" key="2"
 scroll="no" len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MSN" llbl="LMSName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SIDR" llbl="LSIDR" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="SIDL" llbl="LSIDL" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSL(LNK12002) LINKN(5) MBR(IMS1) CC(   0) MSN(ELINK120) SIDR(26)
 SIDL(16) </rsp>
<rsp>MSL(LNK12002) LINKN(5) MBR(IMS1) CC(   0) MSN(ELINK122) SIDR(29)
 SIDL(19) </rsp>
<rsp>MSL(LNK12M02) LINKN(9) MBR(IMS1) CC(   0) MSN(ELINK126) SIDR(54)
 SIDL(46) </rsp>
<rsp>MSL(LNK12V02) LINKN(10) MBR(IMS1) CC(   0) MSN(LINK12V1) SIDR(20)
 SIDL(10) </rsp>
<rsp>MSL(LNK12V02) LINKN(10) MBR(IMS1) CC(   0) MSN(LINK13V1) SIDR(36)
 SIDL(16) </rsp>
<rsp>MSL(LNK12V02) LINKN(10) MBR(IMS1) CC(   0) MSN(LINK12V ) SIDR(256)
 SIDL(1012) </rsp>
<rsp>MSL(LNK12V02) LINKN(10) MBR(IMS1) CC(   0) MSN(LINK12Y ) SIDR(90)
 SIDL(500) </rsp>
<rsp>MSL(LNK12V02) LINKN(10) MBR(IMS1) CC(   0) MSN(LINK13V ) SIDR(266)
 SIDL(2036) </rsp>
<rsp>MSL(LNK12V02) LINKN(10) MBR(IMS1) CC(   0) MSN(LINK13X ) SIDR(80)
 SIDL(300) </rsp>
<rsp>MSL(LNK12C02) LINKN(18) MBR(IMS1) CC(   0) MSN(LINK12C2) SIDR(92)
 SIDL(72) </rsp>
```

```
<rsp>MSL(LNK12T02) LINKN(23) MBR(IMS1) CC(   0) MSN(LINK12T2) SIDR(85)
 SIDL(75) </rsp>
<rsp>MSL(LNK12T02) LINKN(23) MBR(IMS1) CC(   0) MSN(LINK12T3) SIDR(97)
 SIDL(75) </rsp>
<rsp>MSL(LNK12X02) LINKN(26) MBR(IMS1) CC(   0) MSN(LINK12X2) SIDR(88)
 SIDL(78) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY MSLINK command is issued with a wildcard parameter to display all of the MSC logical links that match the wildcard name, and displays the MSC logical link path (MSNAME) names, local SYSID, and remote SYSIDs.

Example 3 for QUERY MSLINK command

TSO SPOC input:
```
QUERY MSLINK NAME(LNK31V02,LNK32M01,LNK32C01,LNK32T01,LNKV001) SHOW(DEFN)
```

TSO SPOC output:

(screen 1)
```
MSLink    LMSLink# MbrName   CC Repo IMSid MSPLink  LMSPLink PID LPID
LNKV001            IMS1       0 Y          PLNKV001          ZA
LNKV001        33 IMS1        0      IMS1           PLNKV001      ZA
LNK31V02           IMS1       0 Y          PLNKSON1          SA
LNK31V02        8 IMS1        0      IMS1           PLNKSON1      SA
LNK32C01           IMS1       0 Y          PLNK23CA          BD
LNK32C01       12 IMS1        0      IMS1           PLNK23CA      BD
LNK32M01           IMS1       0 Y          PLNK23M           BE
LNK32M01        3 IMS1        0      IMS1           PLNK23M       BE
LNK32T01           IMS1       0 Y          PLNK32TA          TD
LNK32T01       20 IMS1        0      IMS1           PLNK32TA      TD
```

(screen 2)
```
MSLink    LMSLink# MbrName  Repo DefMdtbl  LDefMdtbl  BufSize LBufSize
LNKV001            IMS1      Y                          4096
LNKV001        33 IMS1                                          4096
LNK31V02           IMS1      Y    MTMSCVAA              1024
LNK31V02        8 IMS1                      MTMSCVAA             1024
LNK32C01           IMS1      Y                          1024
LNK32C01       12 IMS1                                          1024
LNK32M01           IMS1      Y                          1024
LNK32M01        3 IMS1                                          1024
LNK32T01           IMS1      Y                         65536
LNK32T01       20 IMS1                                         65536
```

(screen 3)
```
MSLink    LMSLink# MbrName  Repo Bandwidth LBandwidth ASR LASR BackUp LBackUp
LNKV001            IMS1      Y   OFF                  OFF          NO
LNKV001        33 IMS1                     OFF            OFF            NO
LNK31V02           IMS1      Y   OFF                  OFF          NO
LNK31V02        8 IMS1                     OFF            OFF            NO
LNK32C01           IMS1      Y   OFF
LNK32C01       12 IMS1                     OFF
LNK32M01           IMS1      Y   OFF
LNK32M01        3 IMS1                     OFF
LNK32T01           IMS1      Y   ON                                NO
LNK32T01       20 IMS1                     ON                             NO
```

(screen 4)

```
               MSLink    LMSLink# MbrName  Repo SyncOpt   LSyncOpt
               LNKV001            IMS1     Y    SYNCSESS
               LNKV001      33 IMS1                       SYNCSESS
               LNK31V02           IMS1     Y    FORCSESS
               LNK31V02       8 IMS1                       FORCSESS
               LNK32C01           IMS1     Y
               LNK32C01      12 IMS1
               LNK32M01           IMS1     Y
               LNK32M01       3 IMS1
               LNK32T01           IMS1     Y    SYNCSESS
               LNK32T01      20 IMS1                       SYNCSESS
```

(screen 5)

```
               MSLink    LMSLink# MbrName  Repo TimeCreate           LTimeCreate
               LNKV001            IMS1     Y    2015.307 11:10:46.05
               LNKV001      33 IMS1                                  2015.307 11:10:46.05
               LNK31V02           IMS1     Y    2015.307 11:09:46.97
               LNK31V02       8 IMS1                                  2015.307 11:09:46.97
               LNK32C01           IMS1     Y    2015.307 11:09:46.97
               LNK32C01      12 IMS1                                  2015.307 11:09:46.97
               LNK32M01           IMS1     Y    2015.307 11:09:46.97
               LNK32M01       3 IMS1                                  2015.307 11:09:46.97
               LNK32T01           IMS1     Y    2015.307 11:09:46.97
               LNK32T01      20 IMS1                                  2015.307 11:09:46.97
```

(screen 6)

```
               MSLink    LMSLink# MbrName  Repo TimeUpdate           LTimeUpdate
               LNKV001            IMS1     Y
               LNKV001      33 IMS1
               LNK31V02           IMS1     Y
               LNK31V02       8 IMS1
               LNK32C01           IMS1     Y
               LNK32C01      12 IMS1
               LNK32M01           IMS1     Y
               LNK32M01       3 IMS1
               LNK32T01           IMS1     Y
               LNK32T01      20 IMS1
```

(screen 7)

```
               MSLink    LMSLink# MbrName  Repo TimeAccess
               LNKV001            IMS1     Y
               LNKV001      33 IMS1
               LNK31V02           IMS1     Y
               LNK31V02       8 IMS1
               LNK32C01           IMS1     Y
               LNK32C01      12 IMS1
               LNK32M01           IMS1     Y
               LNK32M01       3 IMS1
               LNK32T01           IMS1     Y
               LNK32T01      20 IMS1
```

OM API input:

```
CMD(QRY MSLINK NAME(LNK31V02,LNK32M01,LNK32C01,LNK32T01,LNKV001) SHOW(DEFN)
```

OM API output:

```
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSLINK          </kwd>
<input>QRY MSLINK NAME(LNK31V02,LNK32M01,LNK32C01,LNK32T01,LNKV001)
 SHOW(DEFN) </input>
</cmd>
```

```
<cmdrsphdr>
<hdr slbl="MSL" llbl="MSLink" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LINKN" llbl="LMSLink#" scope="LCL" sort="a" key="2"
 scroll="no" len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RMSP" llbl="MSPLink" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSP" llbl="LMSPLink" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RPID" llbl="PID" scope="GBL" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PID" llbl="LPID" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RDMTB" llbl="DefMdtbl" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DMTB" llbl="LDefMdtbl" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="AMTB" llbl="LActMdtbl" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RBUFSZ" llbl="BufSize" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="BUFSZ" llbl="LBufSize" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="RBANDW" llbl="Bandwidth" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="BANDW" llbl="LBandwidth" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RASR" llbl="ASR" scope="GBL" sort="n" key="0" scroll="yes"
 len="3" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ASR" llbl="LASR" scope="LCL" sort="n" key="0" scroll="yes"
 len="3" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RBKUP" llbl="BackUp" scope="GBL" sort="n" key="0"
 scroll="yes" len="2" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="BKUP" llbl="LBackUp" scope="LCL" sort="n" key="0"
 scroll="yes" len="2" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="RSYNC" llbl="SyncOpt" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="SYNC" llbl="LSyncOpt" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RTMCR" llbl="TimeCreate" scope="GBL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RTMUP" llbl="TimeUpdate" scope="GBL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="EXPN" llbl="ExportNeeded" scope="LCL" sort="y" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSL(LNK31V02) LINKN(8) MBR(IMS1) CC(    0) MSP(PLNKSON1)
 CID(00000000) PID(SA) DMTB(MTMSCVAA) CNTR(0) CNTS(0) ASR(OFF )
```

```
|                    BUFSZ(1024) BANDW(OFF ) LSTT(STOCOMM,IDLE,COLDSESS) BKUP(NO)
|                    SYNC(FORCSESS) TMCR(2015.307 11:09:46.97) DFNT(MSCGEN) IMSID(IMS1)
|                    </rsp>
|                    <rsp>MSL(LNK32M01) LINKN(3) MBR(IMS1) CC(   0) MSP(PLNK23M ) PID(BE)
|                    CNTR(0) CNTS(0) BUFSZ(1024) BANDW(OFF ) LSTT(STOCOMM,IDLE,COLDSESS)
|                    TMCR(2015.307 11:09:46.97) DFNT(MSCGEN) IMSID(IMS1) </rsp>
|                    <rsp>MSL(LNK32C01) LINKN(12) MBR(IMS1) CC(   0) MSP(PLNK23CA) PID(BD)
|                    CNTR(0) CNTS(0) BUFSZ(1024) BANDW(OFF ) LSTT(STOCOMM,IDLE,NOTOPEN)
|                    TMCR(2015.307 11:09:46.97) DFNT(MSCGEN) IMSID(IMS1) </rsp>
|                    <rsp>MSL(LNK32T01) LINKN(20) MBR(IMS1) CC(   0) MSP(PLNK32TA) PID(TD)
|                    CNTR(0) CNTS(0) BUFSZ(65536) BANDW(ON ) LSTT(STOCOMM,IDLE,COLDSESS)
|                    BKUP(NO) SYNC(SYNCSESS) TMCR(2015.307 11:09:46.97) DFNT(MSCGEN)
|                    IMSID(IMS1) </rsp>
|                    <rsp>MSL(LNKV001 ) LINKN(33) MBR(IMS1) CC(   0) MSP(PLNKV001)
|                    CID(00000000) PID(ZA) CNTR(0) CNTS(0) ASR(OFF ) BUFSZ(4096) BANDW(OFF
|                    ) LSTT(STOCOMM,IDLE,COLDSESS) BKUP(NO) SYNC(SYNCSESS) TMCR(2015.307
|                    11:10:46.05) DFNT(CREATE) IMSID(IMS1) </rsp>
|                    <rsp>MSL(LNK31V02) MBR(IMS1   ) CC(   0) REPO(Y) IMSID() RASR(OFF)
|                    RBKUP(NO) RBANDW(OFF) RBUFSZ(1024) RDMTB(MTMSCVAA) RMSP(PLNKSON1)
|                    RPID(SA) RSYNC(FORCSESS) RTMCR(2015.307 11:09:46.97) </rsp>
|                    <rsp>MSL(LNK32M01) MBR(IMS1   ) CC(   0) REPO(Y) IMSID() RASR()
|                    RBANDW(OFF) RBUFSZ(1024) RMSP(PLNK23M ) RPID(BE) RSYNC()
|                    RTMCR(2015.307 11:09:46.97) </rsp>
|                    <rsp>MSL(LNK32C01) MBR(IMS1   ) CC(   0) REPO(Y) IMSID() RASR()
|                    RBANDW(OFF) RBUFSZ(1024) RMSP(PLNK23CA) RPID(BD) RSYNC()
|                    RTMCR(2015.307 11:09:46.97) </rsp>
|                    <rsp>MSL(LNK32T01) MBR(IMS1   ) CC(   0) REPO(Y) IMSID() RASR()
|                    RBKUP(NO) RBANDW(ON ) RBUFSZ(65536) RMSP(PLNK32TA) RPID(TD)
|                    RSYNC(SYNCSESS) RTMCR(2015.307 11:09:46.97) </rsp>
|                    <rsp>MSL(LNKV001 ) MBR(IMS1   ) CC(   0) REPO(Y) IMSID() RASR(OFF)
|                    RBKUP(NO) RBANDW(OFF) RBUFSZ(4096) RMSP(PLNKV001) RPID(ZA)
|                    RSYNC(SYNCSESS) RTMCR(2015.307 11:10:46.05) </rsp>
|                    </cmdrspdata>
```

| **Explanation:** The QUERY command displays all four types of MSC logical links,
| including one that is dynamically created and one that has a mode table defined,
| so that all possible local and repository definition headers are displayed for all
| eight MSLINK attributes.

| ## Example 4 for QUERY MSLINK command

| TSO SPOC input:
| ```
| QUERY MSLINK NAME(LNK12C01,LNK12M01) SHOW(EXPORTNEEDED)
| ```

| TSO SPOC output:
| ```
| MSLink   LMSLink# MbrName   CC ExportNeeded
| LNK12C01   2 IMS1   0 N
| LNK12M01   3 IMS1   0 N
| ```

| OM API input:
| ```
| CMD(QRY MSLINK NAME(LNK12V01,LNK12M01) SHOW(EXPORTNEEDED)
| ```

| OM API output:
| ```
| <imsout>
| <ctl>
| <omname>OM1OM   </omname>
| <omvsn>1.7.0</omvsn>
| <xmlvsn>20 </xmlvsn>
| <statime>2015.322 22:08:27.436635</statime>
| <stotime>2015.322 22:08:27.436888</stotime>
| <staseq>CFDE82A7A765B1CA</staseq>
| <stoseq>CFDE82A7A77584CA</stoseq>
| <rqsttkn1>USRT011 10140827</rqsttkn1>
| ```

```
<rc>02000004</rc>
<rsn>00001014</rsn>
<rsnmsg>CSLN055I</rsnmsg>
<rsntxt>At least one request completed with warning/s.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS    </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001010</rsn>
<rsntxt>No resources or matches found</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSLINK        </kwd>
<input>QRY MSLINK NAME(LNK12C01,LNK12M01) SHOW(EXPORTNEEDED) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSL" llbl="MSLink" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LINKN" llbl="LMSLink#" scope="LCL" sort="a" key="2"
 scroll="no" len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="EXPN" llbl="ExportNeeded" scope="LCL" sort="y" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSL(LNK12C01) LINKN(2) MBR(IMS1) CC(   0) EXPN(N) </rsp>
<rsp>MSL(LNK12M01) LINKN(3) MBR(IMS1) CC(   0) EXPN(N) </rsp>
</cmdrspdata>
```

Explanation: This QUERY command displays whether export is needed for two
MSC logical links that do not need to be exported. The resulting two lines of
output have a completion code of zero, but a non-zero reason code of 1010 and
reason text of No resources or matches found are returned.

Example 5: Querying statistics for a TCP/IP link

In this example, the QUERY MSLINK command is issued twice against the TCP/IP
link LNK12T01: first with SHOW(ALL) specified to view the buffer size and
second to view the statistics.

For this example, the Batch SPOC utility (CSLUSPOC) was used and configured to
print the output in a column format by specifying F=BYCOL in the JCL.

TSO SPOC input to display the buffer size for the link:
```
QUERY MSLINK NAME(LNK12T01) SHOW(ALL)
```

TSO SPOC output that shows the link buffer size. For TCP/IP links, the bandwidth
mode is always on.

(screen 1)

```
MSLink    LMSLink# MbrName    CC LMSPLink LPID LRecdCnt LSentCnt LDefMdtbl
LNK12T01      22 IMS1       0 PLNK12TA TA        0        0
```

(scrolled to the right, screen 2)

```
MSLink    LMSLink# MbrName   LBufSize LBandwidth LAffin
LNK12T01      22 IMS1       65536 ON
```

(scrolled to the right, screen 3)

```
MSLink    LMSLink# MbrName  LclStat                     LBackUp
LNK12T01      22 IMS1       STOCOMM,IDLE,COLDSESS,SYNCSESS    NO
```

(scrolled to the right, screen 4)

```
MSLink    LMSLink# MbrName  LTimeCreate
LNK12T01      22 IMS1       2014.282 15:59:39.74
```

(scrolled to the right, screen 5)

```
MSLink    LMSLink# MbrName  LTimeUpdate          LTimeAccess         LDefnType
LNK12T01      22 IMS1                                                MSCGEN
```

OM API input to display the buffer size for the link:

```
CMD(QRY MSLINK NAME(LNK12T01)  SHOW(ALL))
```

OM API output that shows the link buffer size:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2014.283 17:01:21.646214</statime>
<stotime>2014.283 17:01:21.646548</stotime>
<staseq>CDE24C853B68666A</staseq>
<stoseq>CDE24C853B7D416A</stoseq>
<rqsttkn1>USRT011 10100121</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1     </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSLINK          </kwd>
<input>QUERY MSLINK NAME(LNK12T01) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSL" llbl="MSLink" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LINKN" llbl="LMSLink#" scope="LCL" sort="a" key="2"
 scroll="no" len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MSP" llbl="LMSPLink" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CID" llbl="LCID" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="INT" align="left" skipb="yes" />
<hdr slbl="PID" llbl="LPID" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CNTR" llbl="LRecdCnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
<hdr slbl="CNTS" llbl="LSentCnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
```

```
|                        <hdr slbl="DMTB" llbl="LDefMdtbl" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                        <hdr slbl="AMTB" llbl="LActMdtbl" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                        <hdr slbl="BUFSZ" llbl="LBufSize" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="8" dtype="INT" align="right" skipb="yes" />
|                        <hdr slbl="BANDW" llbl="LBandwidth" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
|                        <hdr slbl="AFFIN" llbl="LAffin" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
|                        <hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
|                        <hdr slbl="BKUP" llbl="LBackUp" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="2" dtype="CHAR" align="right" skipb="yes" />
|                        <hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
|                        <hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
|                        <hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
|                        <hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
|                        <hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
|                         scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
|                      </cmdrsphdr>
|                      <cmdrspdata>
|                      <rsp>MSL(LNK12T01) LINKN(22) MBR(IMS1) CC(   0) MSP(PLNK12TA) PID(TA)
|                       DMTB(        ) CNTR(0) CNTS(0) BUFSZ(65536) BANDW(ON )
|                       LSTT(STOCOMM,IDLE,COLDSESS,SYNCSESS) BKUP(NO) TMCR(2014.282
|                       15:59:39.74) DFNT(MSCGEN) </rsp>
|                      </cmdrspdata>
|                      </imsout>
```

| TSO SPOC input to display the link statistics:

QUERY MSLINK NAME(LNK12T01) SHOW(STATISTICS)

The TSO SPOC output of the link statistics for a TCP/IP link:

```
Response for: QUERY MSLINK NAME(LNK12T01) SHOW(STATISTICS)
MSLink   MSLink# MbrName    CC Option         Start_Time           Tot_Disp_CT Tot_Proc_Time Hi_Proc_Time
-------- ------- --------   ---- ---------------  --------------------  ----------- ------------- ------------
LNK12T01     22 IMS1        0 RESET,CHKPT  2010.209 11:52:59.39          16    0.020195      0.004190
MSLink   MSLink# MbrName    Low_Proc_Time Avg_Proc_Time Chkw_CT ChkwIO_CT Chkw_Rate Tot_Send_CT Tot_Msg_Send_CT
-------- ------- --------   ------------- ------------- -------- --------- --------- ----------- ---------------
LNK12T01     22 IMS1          0.000235      0.001262       5        2 0.000000           5               2
MSLink   MSLink# MbrName    Tot_MsgByte_Send_CT Tot_Byte_Send_CT Hi_Msg_Send_SZ Low_Msg_Send_SZ Avg_Msg_Send_SZ
-------- ------- --------   ------------------- ---------------- -------------- --------------- ---------------
LNK12T01     22 IMS1                1,040            2,816            538            502             520
MSLink   MSLink# MbrName    Send_Msg_Time Send_MsgCT_Rate Send_MsgByte_Rate Tot_Qget_CT Tot_Qget_Time Hi_Qget_Time
-------- ------- --------   ------------- --------------- ---------------- ----------- ------------- -------------
LNK12T01     22 IMS1         47.917786      30.472011       15,845.445957          6    0.000129      0.000043
MSLink   MSLink# MbrName    Low_Qget_Time Avg_Qget_Time Tot_SendIO_Time Hi_SendIO_Time Low_SendIO_Time SendIO_Req_Rate
-------- ------- --------   ------------- ------------- --------------- -------------- --------------- ---------------
LNK12T01     22 IMS1          0.000004      0.000021      0.052853        0.016744       0.011833      94.602009
MSLink   MSLink# MbrName    SendIO_Byte_Rate Tot_Loc_SCI_SendIO_Time Hi_Loc_SCI_SendIO_Time Low_Loc_SCI_SendIO_Time
-------- ------- --------   ---------------- ----------------------- ---------------------- -----------------------
LNK12T01     22 IMS1        53,279.851664          0.001503                0.000988                0.000103
MSLink   MSLink# MbrName    Tot_Loc_ICON_SendIO_Time Hi_Loc_ICON_SendIO_Time Low_Loc_ICON_SendIO_Time
-------- ------- --------   ------------------------ ----------------------- ------------------------
LNK12T01     22 IMS1                0.023818                0.008290                0.004861
MSLink   MSLink# MbrName    Tot_TCPIP_SendIO_Time Hi_TCPIP_SendIO_Time Low_TCPIP_SendIO_Time Tot_Rmt_ICON_SendIO_Time
-------- ------- --------   --------------------- -------------------- --------------------- ------------------------
LNK12T01     22 IMS1                0.003279              0.001210              0.000506              0.019013
MSLink   MSLink# MbrName    Hi_Rmt_ICON_SendIO_Time Low_Rmt_ICON_SendIO_Time Tot_Rmt_SCI_SendIO_Time
-------- ------- --------   ----------------------- ------------------------ -----------------------
LNK12T01     22 IMS1                0.005403                0.004312                0.005240
MSLink   MSLink# MbrName    Hi_Rmt_SCI_SendIO_Time Low_Rmt_SCI_SendIO_Time Tot_Rec_CT Tot_Msg_Rec_CT
-------- ------- --------   ---------------------- ----------------------- ---------- --------------
LNK12T01     22 IMS1                0.005403                0.004312          5             3
MSLink   MSLink# MbrName    Tot_MsgByte_Rec_CT Tot_Byte_Rec_CT Hi_Msg_Rec_SZ Low_Msg_Rec_SZ Avg_Msg_Rec_SZ
-------- ------- --------   ------------------ --------------- ------------- -------------- --------------
```

```
LNK12T01     22 IMS1                1,595         3,419          549           502           531
MSLink   MSLink# MbrName   Rec_Msg_Time Tot_Qput_CT Tot_Qput_Time Hi_Qput_Time Low_Qput_Time Avg_Qput_Time
-------- ------- -------- -------------- ------------ -------------- ------------- -------------- --------------
LNK12T01     22 IMS1          235.933123           0      0.000000      0.000000      0.000000      0.000000
MSLink   MSLink# MbrName   Tot_RecIO_Time Hi_RecIO_Time Low_RecIO_Time RecIO_Req_Rate RecIO_Byte_Rate
-------- ------- -------- -------------- ------------- -------------- -------------- ---------------
LNK12T01     22 IMS1            0.053281      0.014511      0.012323     93.842082  64,169.216043
```

Example 6: Querying statistics for VTAM links

Explanation: In this example, the QUERY MSLINK command is issued twice: once in the IMS systems at each end of the VTAM link. The commands were issued after each side sent 10 remote transactions that processed and sent a response message back (10 transactions + 10 responses = 20 messages sent each direction).

TSO SPOC input:

```
QUERY MSLINK NAME(LNK12V02) SHOW(STATISTICS)
```

TSO SPOC output:

(screen 1)

```
MSLink      LMSLink#  MbrName  CC   LOption        LStart_Time
LNK12V02        10    IMS1     0    NORESET,CHKPT  2006.261 19:03:58.77
```

(scrolled to the right, screen 2)

```
MSLink      LMSLink#  MbrName   LTot_Disp_CT  LTot_Proc_Time  LHi_Proc_Time  LLow_Proc_Time  LAvg_Proc_Time
LNK12V02        10    IMS1                98        0.176661       0.059530        0.000003        0.000853
```

(scrolled to the right, screen 3)

```
MSLink      LMSLink#  MbrName    LChkw_CT   LChkwIO_CT  LChkw_Rate  LTot_Send_CT  LTot_Msg_Send_CT
LNK12V02        10    IMS1             33           33    0.165016            26                20
```

(scrolled to the right, screen 4)

```
MSLink      LMSLink#  MbrName   LTot_MsgByte_Send_CT  LTot_Byte_Send_CT  LHi_Msg_Send_SZ  LLow_Msg_Send_SZ
LNK12V02        10    IMS1                    10,461             22,403              578               502
```

(scrolled to the right, screen 5)

```
MSLink      LMSLink#  MbrName   LAvg_Msg_Send_SZ  LSend_Msg_Time  LTot_Qget_CT  LTot_Qget_Time  LHi_Qget_Time
LNK12V02        10    IMS1                   523        2.408548            51        0.002054       0.001070
```

(scrolled to the right, screen 6)

```
MSLink      LMSLink#  MbrName   LLow_Qget_Time  LAvg_Qget_Time  LTot_SendIO_Time  LHi_SendIO_Time
LNK12V02        10    IMS1            0.000004        0.000040          0.505401         0.193435
```

(scrolled to the right, screen 7)

```
MSLink      LMSLink#  MbrName    LLow_SendIO_Time  LSendIO_Req_Rate  LSendIO_Byte_Rate  LTot_Rec_CT
LNK12V02        10    IMS1               0.000708         51.444298             44,327           26
```

(scrolled to the right, screen 8)

```
MSLink      LMSLink#  MbrName    LTot_Msg_Rec_CT  LTot_MsgByte_Rec_CT  LTot_Byte_Rec_CT  LHi_Msg_Rec_SZ
LNK12V02        10    IMS1                    20               10,503            22,445             584
```

(scrolled to the right, screen 9)

```
MSLink      LMSLink#  MbrName    LLow_Msg_Rec_SZ  LAvg_Msg_Rec_SZ  LRec_Msg_Time  LTot_Qput_CT  LTot_Qput_Time
LNK12V02        10    IMS1                   502              525       2.320062            40        0.037326
```

(scrolled to the right, screen 10)

```
MSLink      LMSLink#  MbrName    LHi_Qput_Time  LLow_Qput_Time  LAvg_Qput_Time  LTot_RecIO_Time  LHi_RecIO_Time
LNK12V02        10    IMS1            0.015957        0.000008        0.000933         0.070947        0.025376
```

(scrolled to the right, screen 11)

```
MSLink     LMSLink#   MbrName     LLow_RecIO_Time   LRecIO_Req_Rate   LRecIO_Byte_Rate
LNK12V02        10     IMS1             0.000155        366.470750         148,040.09
```

TSO SPOC input:

```
QUERY MSLINK NAME(LNK21V02) SHOW(STATISTICS)
```

TSO SPOC output:

(scrolled to the right, screen 1)

```
MSLink     LMSLink#  LMbrName   CC    LOption          LStart_Time
LNK21V02        13    IMS2        0    NORESET,CHKPT    2006.261 19:03:08.42
```

(scrolled to the right, screen 2)

```
MSLink     LMSLink#  MbrName    LTot_Disp_CT   LTot_Proc_Time   LHi_Proc_Time   LLow_Proc_Time   LAvg_Proc_Time
LNK21V02        13    IMS2                97        0.150468         0.060122         0.000003         0.000696
```

(scrolled to the right, screen 3)

```
MSLink     LMSLink#  MbrName    LChkw_CT    LChkwIO_CT   LChkw_Rate   LTot_Send_CT   LTot_Msg_Send_CT
LNK21V02        13    IMS2            33           33     0.544554             26                 20
```

(scrolled to the right, screen 4)

```
MSLink     LMSLink#  MbrName    LTot_MsgByte_Send_CT   LTot_Byte_Send_CT   LHi_Msg_Send_SZ   LLow_Msg_Send_SZ
LNK21V02        13    IMS2                     10,503              22,445               584                502
```

(scrolled to the right, screen 5)

```
MSLink     LMSLink#  MbrName    LAvg_Msg_Send_SZ   LSend_Msg_Time   LTot_Qget_CT   LTot_Qget_Time   LHi_Qget_Time
LNK21V02        13    IMS2                   525         2.395508             51         0.003884        0.001169
```

(scrolled to the right, screen 6)

```
MSLink     LMSLink#  MbrName    LLow_Qget_Time   LAvg_Qget_Time   LTot_SendIO_Time   LHi_SendIO_Time
LNK21V02        13    IMS2            0.000004         0.000076           1.304330          0.217838
```

(scrolled to the right, screen 7)

```
MSLink     LMSLink#  LMbrName    LLow_SendIO_Time   LSendIO_Req_Rate   LSendIO_Byte_Rate   LTot_Rec_CT
LNK21V02        13    IMS2               0.000821          19.933605              17,208            26
```

(scrolled to the right, screen 8)

```
MSLink     LMSLink#  MbrName    LTot_Msg_Rec_CT   LTot_MsgByte_Rec_CT   LTot_Byte_Rec_CT   LHi_Msg_Rec_SZ
LNK21V02        13    IMS2                   20                10,461             22,403              578
```

(scrolled to the right, screen 9)

```
MSLink     LMSLink#  MbrName    LLow_Msg_Rec_SZ   LAvg_Msg_Rec_SZ   LRec_Msg_Time   LTot_Qput_CT   LTot_Qput_Time
LNK21V02        13    IMS2                  502               523        2.344878             40         0.054777
```

(scrolled to the right, screen 10)

```
MSLink     LMSLink#  MbrName    LHi_Qput_Time   LLow_Qput_Time   LAvg_Qput_Time   LTot_RecIO_Time   LHi_RecIO_Time
LNK21V02        13    IMS2           0.023559         0.000007         0.001369          0.210252         0.036846
```

(scrolled to the right, screen 11)

```
MSLink     LMSLink#  MbrName    LLow_RecIO_Time   LRecIO_Req_Rate   LRecIO_Byte_Rate
LNK21V02        13    IMS2             0.000155        123.661130           106,553
```

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

➡ List of commands with similar functions for multiple resources (Operations and Automation)

QUERY MSNAME command

Use the QUERY MSNAME command to query definition and status information about the specified logical link path.

Subsections:

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY MSNAME command and keywords can be issued.

Table 119. Valid environments for the QUERY MSNAME command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
QUERY MSNAME	X		X
NAME	X		X
QCNT	X		X
STATUS	X		X
SHOW	X		X

Syntax

```
|                                                                          ►◄
    ►─┬─QCNT(─┬─LT─┬──,number)─────────────────────────────────────
      │       ├─LE─┤    └─SHOW(QCNT)─┘
      │       ├─GT─┤
      │       ├─GE─┤
      │       ├─EQ─┤
      │       └─NE─┘
      │                    ┌─,──────────────┐
      ├─SHOW(──▼─┬─ALL────────┬─)─┬──────────────────────────────────
      │          ├─DEFN───────┤   │        ┌─,────────┐
      │          ├─DEFNTYPE───┤   └─STATUS(─▼─┬─DYN─────┬─)─┘
      │          ├─GLOBAL,DEFN┤              ├─QERR────┤
      │          ├─IMSID──────┤              ├─STOQ────┤
      │          ├─LOCAL,DEFN─┤              └─STOSEND─┘
      │          ├─MSLINK─────┤
      │          ├─MSPLINK────┤
      │          ├─QCNT───────┤
      │          ├─STATUS─────┤
      │          ├─SYSID──────┤
      │          └─TIMESTAMP──┘
      ├─SHOW(EXPORTNEEDED)────────────────────────────────────────
      │          ┌─,────────┐
      └─STATUS(──▼─┬─DYN─────┬─)─────────────────────────────────
                  ├─QERR────┤
                  ├─STOQ────┤
                  └─STOSEND─┘
```
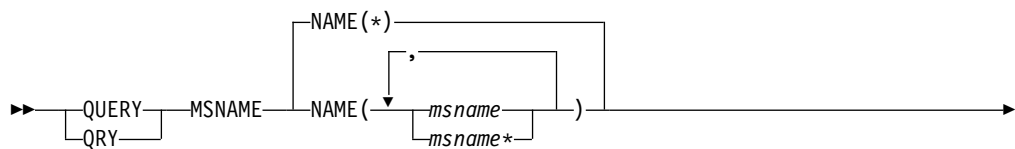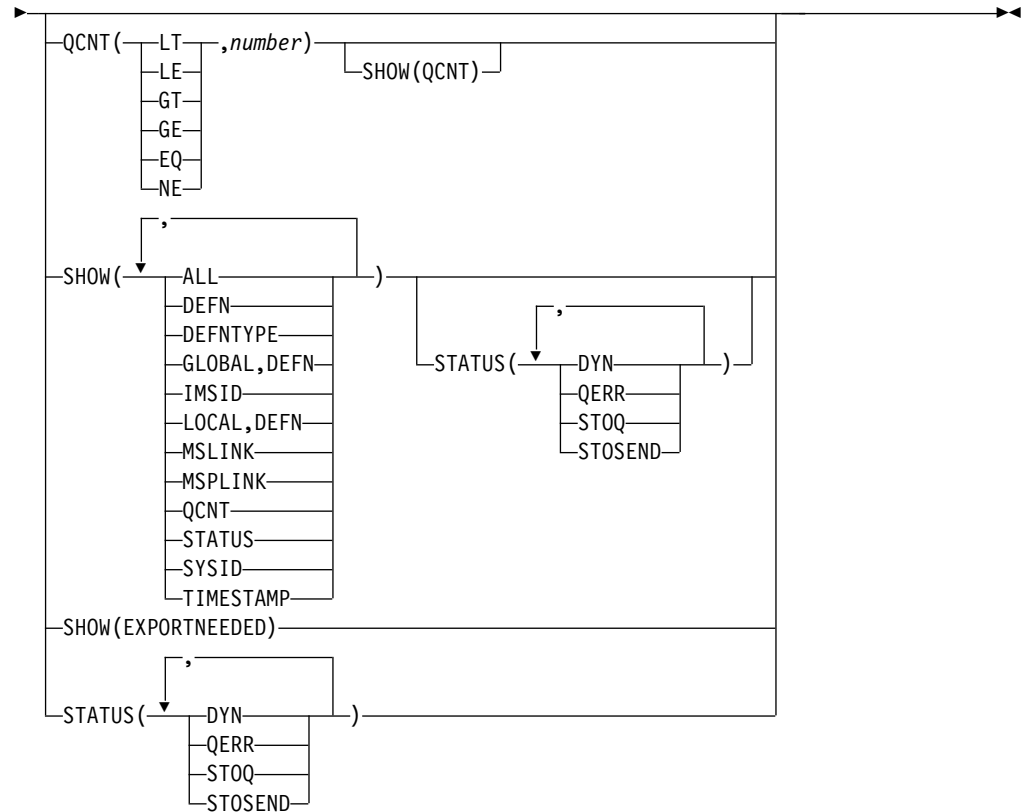
Keywords

NAME()

Specifies the 1- to 8-character names of the MSC logical link paths that are to be displayed or indicates that the command is to be applied to all the links in the system. Wildcards (*) can be specified in the name. The default is NAME(*), which returns all MSNAME resources.

QCNT()

Selects logical link paths that have a queue count less than (LT), less than or equal to (LE), greater than (GT), greater than or equal to (GE), equal to (EQ), or not equal to (NE) the specified numbers. The specified number cannot be a 1 when LT is specified. This allows additional filtering by QCNT value.

Logical link paths with a queue count of 0 are not returned when you specify the QCNT parameter. When you specify the QCNT(LT,n) parameter, *msnames* with a queue count greater than 0 or less than 'n' are returned. If you do not specify the STATUS or QCNT parameters, all of the logical link paths that match the MSNAME names are returned.

The QCNT filter is valid in both a shared-queues environment and a non-shared-queues environment.

In a shared-queues environment, if QCNT is specified, the shared queues are slower than local queues. In this environment, the QRY MSNAME command is processed only by the master IMS because the queues are global. The command master returns all the logical link paths on the shared queues that match the specified queue count parameter. If QCNT is specified with a

wildcard logical link path name, all of the shared queues logical link path messages on the coupling facility must be read.

In a non-shared-queues environment, the local queue count values are used to determine the *msnames* to be displayed. In this environment, the QRY MSNAME QCNT command is processed by each IMS that the command is routed to because the queues are local. Each IMS returns all the *msnames* found locally that match the specified queue count parameter.

When you specify the QCNT parameter, the output returned includes the queue count, even if the SHOW(QCNT) option is not specified.

The QCNT filter is mutually exclusive with the STATUS filter. The QCNT filter is mutually exclusive with SHOW(), except for SHOW(QCNT).

SHOW()
Specifies the MSNAME output fields to be returned. If SHOW() is not specified, only the logical link path names and associated remote system identification and local system identification are returned. The MSNAME name is always returned, along with the name of the IMS that created the output for the link path and the completion code. Only SHOW(QCNT) is allowed with the QCNT() parameter. All other SHOW filters are mutually exclusive with the QCNT() parameter. The parameters supported with the SHOW keyword are:

ALL
Includes all of the information in the other SHOW parameters.

DEFN
Specifies that the resource definitions are to be returned.

The MSC logical link path (MSNAME) attributes that can be returned are: MSLINK, MSPLINK, SIDL, SIDR, the repository create and update time stamps, and the IMS runtime create and update time stamps. Only the local MSPLINK can be displayed with the QRY MSNAME command.

Link number is a local, run-time value that is not stored in the repository.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition that is obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE, QCNT or STATUS. The LclStat and LDefnType columns, which are returned on the QRY MSNAME SHOW(ALL) command, are not returned with SHOW(DEFN).
- SHOW(DEFN) cannot be specified with filter QCNT or STATUS.
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).

Resource definitions that are stored in the repository are used to determine the response lines with the repository information, and the runtime resource definitions are used to determine the response lines with the IMS runtime resource information.

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

DEFNTYPE
Displays the definition type that the resource was defined with.

EXPORTNEEDED
Specifies that only the MSC logical link path resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command. SHOW(EXPORTNEEDED) behaves like a filter by only displaying resources that need to be exported, as if QUERY MSNAME EXPORTNEEDED(Y) were supported.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the MSC logical link paths that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each MSC logical link path specified on the command. If the specified MSC logical link path does not need to be exported to the repository, an N is returned in the ExportNeeded column.

Restrictions:
- SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords.
- SHOW(EXPORTNEEDED) is valid only when IMS is enabled with dynamic resource definition for MSC resources (MSCRSCS=DYN) and the IMSRSC repository.
- The QUERY MSNAME SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources that have not been exported to the IMSRSC repository, you can issue the /CHECKPOINT command to automatically write the resources to the repository so that they are available for the next IMS cold start.

GLOBAL

For output fields that have both local and global values, this option returns only the global values. GLOBAL must be specified with DEFN, otherwise the command is rejected. If used with another SHOW keyword to request a specific output field, this option requests that only the global value of the specified output field is returned.

SHOW(DEFN,GLOBAL,LOCAL) is equivalent to SHOW(DEFN).

IMSID

Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only when the command master IMS is enabled to use the repository.

IMSID cannot be specified with any other SHOW keywords other than DEFN or GLOBAL.

When SHOW(DEFN,IMSID) is specified, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN keyword, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

LOCAL

For output fields that have both local and global values, this option returns only the local values. LOCAL must be specified with DEFN, otherwise the command is rejected. If used with another SHOW keyword to request a specific output field, this option requests that only the local value of the specified output field is returned. Local output is returned by each IMS that processes the command.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

SHOW(DEFN,GLOBAL,LOCAL) is equivalent to SHOW(DEFN).

MSLINK

Displays the logical link assigned to the specified logical link path.

MSPLINK

Displays the physical link that is assigned to the specified logical link path.

QCNT

Specifies that queue count information is returned. Both local and global queue counts are returned.

The local queue counts value returned on this command represents the messages being processed by the IMS system where this command is issued. In a shared-queues environment, do not use the local queue counts for queue evaluation purposes. The global queue counts, which are returned on this command, represent the current message queue counts on the shared queues structure.

In a shared-queues environment, the command master returns both the global and its local queue count information for all the MSNAMEs

specified. The MSNAME does not have to be defined at the command master IMS. All other IMS systems return their local queue count information.

In a non-shared-queues environment, all IMS systems return their local queue count information.

STATUS
Displays the logical link path status.

SYSID
Displays the remote system identification and the local system identification for this logical link path.

TIMESTAMP
Displays the creation time (TimeCreate), last update time (TimeUpdate), last access time (TimeAccess), and last import time (TimeImport) time stamps are returned. The time is returned in local time in the format `YYYY.JJJ HH:MM:SS:TH`:

YYYY
The year

JJJ
The Julian day (001 - 365)

HH The hour (01 - 24)

MM The minute (00 - 59)

SS The seconds (00 - 59)

TH The tenths and hundredths of a second (00 - 99)

STATUS()
Displays logical link paths that display at least one of the specified status. The STATUS filter is mutually exclusive with the QCNT filter. When you specify the STATUS keyword, the output returned includes the logical link path status, even if you do not specify SHOW(STATUS).

DYN
Displays MSNAMEs that have been dynamically created in a shared queue environment.

QERR
Displays MSNAMEs with the queue error status.

STOQ
Displays MSNAMEs with the stopped queuing status.

STOSEND
Displays MSNAMEs with the stopped sending status.

Usage notes

You can issue this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

You can use the following filters to see information about your resource definitions:

| • Specify SHOW(DEFN) to display information about resource definitions.
| • Specify SHOW(DEFN,IMSID) to see which IMS systems have the resource
| defined and the attributes or resource definitions at each IMS system.
| • Specify SHOW(IMSID) to see which IMS systems have the resource defined.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY MSNAME and the type-1 IMS
commands that perform similar functions.

Table 120. Type-1 equivalents for the QUERY MSNAME command

QUERY MSNAME command	Similar IMS type-1 command
QUERY MSNAME NAME(msname ǀ msname*) SHOW(ALL)	/DIS MSNAME msname ǀ msname*
QUERY MSNAME NAME(msname ǀ msname*) SHOW(QCNT)	/DIS MSNAME msname QCNT
QUERY MSNAME NAME(*) SHOW(ALL)	/DIS MSNAME ALL
QUERY MSNAME NAME(msname ǀ *) SHOW(MSPLINK ǀ MSLINK)	/DIS ASSIGNMENT MSNAME msname ǀ msname* ǀ ALL

Output fields

Short label
    Contains the short label generated in the XML output.

Long label
    Contains the long label generated in the XML output.

Keyword
    Identifies keyword on the command that caused the field to be generated.
    error appears for output fields that can appear for a nonzero completion
    code. N/A (not applicable) appears for output fields that are always
    returned.

Meaning
    Provides a brief description of the output field.

Table 121. Output field descriptions for the QUERY MSNAME command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	LCL	Completion code.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the nonzero completion code.

Short label	Long label	Keyword	Scope	Meaning
DFNT	LDefnType	DEFNTYPE	LCL	Definition type, which can be one of the following: **CREATE** Defined by the CREATE MSNAME command. **DFSDFxxx** Definition PROCLIB member, where xxx is the suffix defined to this IMS. This MSNAME was defined by IMS to represent each local SYSID defined with the SYSID= keyword in the DFSDFxxx member MSC section. **MSCGEN** Defined by system definition in the DFSCLCxy member of the IMS RESLIB. **SHAREDQ** Defined as an MSNAME that is defined remotely in another IMS system in a shared-queues group. **UPDATE** Defined by system definition in the DFSCLCxy member of the IMS RESLIB, but modified in online memory by a /MSASSIGN MSNAME command or an UPDATE MSNAME command that changed at least one attribute. At the next IMS cold start, the definition reverts back to the definition in the DFSCLCxy member.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the MSC logical link path has been exported to the IMSRSC repository. The value can be Y or N.
IMSID	IMSid	IMSID	GBL	Returns from the repository the IMSIDs that have the resource defined.
LINKN	LMSLink#	MSLINK	LCL	Link number of logical link associated with the link path.
LQ	LQCnt	QCNT	LCL	Local queue count.
LSTT	LclStat	STATUS	LCL	The current status of the logical link path.
MBR	MbrName	N/A	LCL	IMSplex member that built the output line.
MSL	LMSLink	MSLINK	LCL	Logical link associated with the logical link path.
MSN	MSName	N/A	LCL	Logical link path name.
MSP	LMSPLink	MSPLINK	LCL	Physical link associated with the logical link path.
QCNT	QCnt	QCNT	GBL	Global queue count.

Short label	Long label	Keyword	Scope	Meaning
REPO	Repo	DEFN	GBL	Indicates whether the line shows the stored resource definitions. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
RMSL	MSLink	DEFN, MSLINK	GBL	Displays the logical link assigned to the specified logical link path. The output is returned from the repository.
RSIDL	SIDL	DEFN, SYSID	GBL	Indicates the local system ID. The output is returned from the repository.
RSIDR	SIDR	DEFN, SYSID	GBL	Indicates the remote system ID. The output is returned from the repository.
RTMCR	TimeCreate	DEFN	GBL	Indicates the time that the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN	GBL	Indicates the time that the resource was last updated in the repository.
SIDL	LSIDL	SYSID	LCL	Local system identification for the logical link path.
SIDR	LSIDR	SYSID	LCL	Remote system identification for the logical link path.
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the resource was last accessed. This is not yet supported for MSC resources.
TMCR	LTimeCreate	TIMESTAMP	LCL	The time that the resource was created with a CREATE MSNAME command, an IMPORT command that creates the MSC MSNAME, or an IMS cold start. The create time is retained across IMS warm start, emergency restart, export, and import. The output is returned from the local IMS.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the resource was last imported, if applicable. The local import timestamp is set to be the IMS cold start time, if the resource was automatically imported from the IMSRSC repository during IMS cold start. The import time is retained across IMS warm start and emergency restart. The output is returned from the local IMS.
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated as a result of an UPDATE MSNAME command or an /MSASSIGN MSNAME command. The update time is retained across warm start and emergency restarts. The output value is returned from the local IMS.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY MSNAME command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 122. Return and reason codes for the QUERY MSNAME command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY MSNAME command completed successfully.
X'00000004'	X'00001010'	No resources were found to be returned for one of the following reasons: • The resource names specified were invalid. • There were no resources that match the filter specified. • If SHOW(EXPORTNEEDED) was specified, no resources were required to be exported to the IMSRSC repository.
X'00000008'	X'00002040'	More than one filter value is specified on the QUERY MSNAME command.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'00004004'	No CQS address space.
X'00000010'	X'00004005'	IMS attempted to obtain the message queue count from CQS, but CQS was not connected to the queue structure.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.

Table 122. Return and reason codes for the QUERY MSNAME command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'0000402D'	Command flavor is not allowed because dynamic resource definition for MSC resources is not enabled; that is, DFSDFxxx is not defined with MSCRSCS=DYN.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Repository access is denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server (RS) is being shut down.
X'00000010'	X'0000450B'	Repository Server (RS) is not available.
X'00000010'	X'0000450C'	Repository Server (RS) is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields related to the IMSRSC repository.
X'00000014'	X'00005004'	The QUERY MSNAME command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	RM request error.

Table 122. Return and reason codes for the QUERY MSNAME command (continued)

Return code	Reason code	Meaning
X'00000014'	X'00005104'	IMS attempted to obtain the message queue count from CQS, but CQS returned with an unexpected return code.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY MSNAME command is rejected because the resource list is unusable. Contact IBM Software Support. Send the output from the command in error, and send a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the RID and RMD data sets of the IMSRSC repository .
X'00000014'	X'00005FFF'	IMS attempted to obtain the message queue count from CQS, but failed because of an internal IMS error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 123. Completion codes for the QUERY MSNAME command

Completion code	Completion code text	Meaning
0		The QUERY MSNAME command completed successfully for the resource.
10	NO RESOURCES FOUND	MSNAME name is invalid, or the specified wildcard parameter does not match any resource names.
98	CQS REQUEST ERROR	IMS was unable to obtain the message queue count from CQS.

Example 1 for QUERY MSNAME command

In this example, the specified logical link paths are returned with all output fields.

TSO SPOC input:

QRY MSNAME NAME(MSNV001) SHOW(ALL)

TSO SPOC output:

(screen 1)

```
MSName    MbrName    CC LMSPLink LMSLink  LMSLink# LSIDR LSIDL    LQCnt
MSNV001   IMS1        0 PLNK12V LNK12V02       10    55   705         0
```

(scrolled to the right, screen 2)

```
MSName    MbrName  LclStat LTimeCreate          LTimeUpdate          LTimeAccess>
MSNV001   IMS1              2015.061 15:55:44.49
```

(scrolled to the right, screen 3)

```
LDefnType
CREATE
```

OM API input:

```
CMD(QRY MSNAME NAME(MSNV001) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2015.062 00:48:04.588000</statime>
<stotime>2015.062 00:48:04.588367</stotime>
<staseq>CE967F8BD51E0A8C</staseq>
<stoseq>CE967F8BD534F90C</stoseq>
<rqsttkn1>USRT011 10164804</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1     </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSNAME          </kwd>
<input>QRY MSNAME NAME(MSNV001) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSN" llbl="MSName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MSP" llbl="LMSPLink" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSL" llbl="LMSLink" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LINKN" llbl="LMSLink#" scope="LCL" sort="a" key="2"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="SIDR" llbl="LSIDR" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="SIDL" llbl="LSIDL" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="LQ" llbl="LQCnt" scope="LCL" sort="d" key="3" scroll="yes"
 len="8" dtype="INT" align="right" skipb="no" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMCR" llbl="LTimeCreate"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMUP" llbl="LTimeUpdate"scope="LCL" sort="n" key="0"
```

```
scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMAC" llbl="LTimeAccess"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMIM" llbl="LTimeImport"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DFNT" llbl="LDefnType"scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSN(MSNV001 ) MBR(IMS1    ) CC(   0) MSP(PLNK12V ) MSL(LNK12V02)
 LINKN(  10) SIDR(  55) SIDL( 705) LQ(   0) DFNT(CREATE  )
 TMCR(2015.061 15:55:44.49) </rsp>
</cmdrspdata>
</imsout>
```

Example 2 for QUERY MSNAME command

TSO SPOC input:

QUERY MSNAME NAME(LINK32B1,LINK32M1,LINK32C1,LINK32T1,MSNV001) SHOW(DEFN)

TSO SPOC output:

(screen 1)

MSName	MbrName	CC	Repo	IMSid	LMSPLink	MSLink	LMSLink	LMSLink#	SIDR	LSIDR
LINK32B1	IMS1	0	Y			LNK32V01			21	
LINK32B1	IMS1	0		IMS1	PLNK23V		LNK32V01	1		21
LINK32C1	IMS1	0	Y			LNK32C01			23	
LINK32C1	IMS1	0		IMS1	PLNK23CA		LNK32C01	12		23
LINK32M1	IMS1	0	Y			LNK32M01			24	
LINK32M1	IMS1	0		IMS1	PLNK23M		LNK32M01	3		24
LINK32T1	IMS1	0	Y			LNK32T01			86	
LINK32T1	IMS1	0		IMS1	PLNK32TA		LNK32T01	20		86
MSNV001	IMS1	0	Y			LNKV001			56	
MSNV001	IMS1	0		IMS1	PLNKV001		LNKV001	33		56

(screen 2)

MSName	MbrName	Repo	SIDL	LSIDL	TimeCreate	LTimeCreate
LINK32B1	IMS1	Y	31		2015.310 14:57:33.98	
LINK32B1	IMS1			31		2015.310 14:57:33.98
LINK32C1	IMS1	Y	33		2015.310 14:57:33.98	
LINK32C1	IMS1			33		2015.310 14:57:33.98
LINK32M1	IMS1	Y	34		2015.310 14:57:33.98	
LINK32M1	IMS1			34		2015.310 14:57:33.98
LINK32T1	IMS1	Y	117		2015.310 14:57:33.98	
LINK32T1	IMS1			117		2015.310 14:57:33.98
MSNV001	IMS1	Y	706		2015.310 14:58:40.37	
MSNV001	IMS1			706		2015.310 14:58:40.37

(screen 3)

MSName	MbrName	Repo	TimeUpdate	LTimeUpdate	LTimeAccess
LINK32B1	IMS1	Y			
LINK32B1	IMS1				
LINK32C1	IMS1	Y			
LINK32C1	IMS1				
LINK32M1	IMS1	Y			
LINK32M1	IMS1				
LINK32T1	IMS1	Y			
LINK32T1	IMS1				
MSNV001	IMS1	Y			
MSNV001	IMS1				

OM API input:

CMD(QRY MSNAME NAME(LINK32B1,LINK32M1,LINK32C1,LINK32T1,MSNV001) SHOW(DEFN)

OM API output:

```
<cmdrsphdr>
<hdr slbl="MSN" llbl="MSName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSP" llbl="LMSPLink" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RMSL" llbl="MSLink" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MSL" llbl="LMSLink" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LINKN" llbl="LMSLink#" scope="LCL" sort="a" key="2"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="RSIDR" llbl="SIDR" scope="GBL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="SIDR" llbl="LSIDR" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="RSIDL" llbl="SIDL" scope="GBL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="SIDL" llbl="LSIDL" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="EXPN" llbl="ExportNeeded" scope="LCL" sort="y" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RTMCR" llbl="TimeCreate"scope="GBL" sort="n" key="0"
scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMCR" llbl="LTimeCreate"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RTMUP" llbl="TimeUpdate"scope="GBL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMUP" llbl="LTimeUpdate"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMAC" llbl="LTimeAccess"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMIM" llbl="LTimeImport"scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSN(LINK32B1) MBR(IMS1   ) CC(   0) MSP(PLNK23V ) MSL(LNK32V01)
 LINKN(  1) SIDR(  21) SIDL(  31) IMSID(IMS1) LQ(   0) DFNT(MSCGEN  )
 TMCR(2015.310 14:57:33.98) </rsp>
<rsp>MSN(LINK32M1) MBR(IMS1   ) CC(   0) MSP(PLNK23M ) MSL(LNK32M01)
 LINKN(  3) SIDR(  24) SIDL(  34) IMSID(IMS1) LQ(   0) DFNT(MSCGEN  )
 TMCR(2015.310 14:57:33.98) </rsp>
<rsp>MSN(LINK32C1) MBR(IMS1   ) CC(   0) MSP(PLNK23CA) MSL(LNK32C01)
 LINKN( 12) SIDR(  23) SIDL(  33) IMSID(IMS1) LQ(   0) DFNT(MSCGEN  )
 TMCR(2015.310 14:57:33.98) </rsp>
<rsp>MSN(LINK32T1) MBR(IMS1   ) CC(   0) MSP(PLNK32TA) MSL(LNK32T01)
LINKN( 20) SIDR(  86) SIDL( 117) IMSID(IMS1) LQ(   0) DFNT(MSCGEN  )
 TMCR(2015.310 14:57:33.98) </rsp>
<rsp>MSN(MSNV001 ) MBR(IMS1   ) CC(   0) MSP(PLNKV001) MSL(LNKV001 )
 LINKN( 33) SIDR(  56) SIDL( 706) IMSID(IMS1) LQ(   0) DFNT(CREATE  )
 TMCR(2015.310 14:58:40.37) </rsp>
<rsp>MSN(LINK32B1) MBR(IMS1   ) CC(   0) REPO(Y) RMSL(LNK32V01) RSIDL(
 31) RSIDR(  21) RTMCR(2015.310 14:57:33.98) </rsp>
<rsp>MSN(LINK32M1) MBR(IMS1   ) CC(   0) REPO(Y) RMSL(LNK32M01) RSIDL(
```

```
|        34) RSIDR(  24) RTMCR(2015.310 14:57:33.98) </rsp>
|    <rsp>MSN(LINK32C1) MBR(IMS1   ) CC(   0) REPO(Y) RMSL(LNK32C01) RSIDL(
|        33) RSIDR(  23) RTMCR(2015.310 14:57:33.98) </rsp>
|    <rsp>MSN(LINK32T1) MBR(IMS1   ) CC(   0) REPO(Y) RMSL(LNK32T01) RSIDL(
|       117) RSIDR(  86) RTMCR(2015.310 14:57:33.98) </rsp>
|    <rsp>MSN(MSNV001 ) MBR(IMS1   ) CC(   0) REPO(Y) RMSL(LNKV001 ) RSIDL(
|       706) RSIDR(  56) RTMCR(2015.310 14:58:40.37) </rsp>
|    </cmdrspdata>
```

| Explanation: This QUERY command displays msnames for all four types of MSC
| logical links, including one that is dynamically created.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

➡ List of commands with similar functions for multiple resources (Operations
and Automation)

QUERY MSPLINK command

Use the QUERY MSPLINK command to query definition and status information
about the specified physical links.

Subsections:

- "Environment"
- "Syntax" on page 396
- "Keywords" on page 396
- "Usage notes" on page 402
- "Equivalent IMS type-1 commands" on page 402
- "Output fields" on page 402
- "Return, reason, and completion codes" on page 409
- Examples

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from
which the QUERY MSPLINK command and keywords can be issued.

Table 124. Valid environments for the QUERY MSPLINK command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
QUERY MSPLINK	X		X
NAME	X		X
SHOW	X		X
STATUS	X		X
TYPE	X		X

Syntax

```
►►──┬─QUERY─┬──MSPLINK──┬──────NAME(*)──────────────────────────┬───────────►
    └─QRY───┘           │           ┌─────,──────┐              │
                        │           ▼            │              │
                        └─NAME(──┬──msplinkname──┴──┬──)────────┘
                                 └─msplinkname*──────┘

►──┬─────────────────────────────────┬──┬──────────────────────────────────┬─►
   │         ┌──────,────────┐        │  │              ┌────,─────┐         │
   │         ▼               │        │  │              ▼          │         │
   │ ┌─SHOW(──┬─ALL──────────┴──)─┐   │  └─STATUS(──┬─NOTOPEN───┬──┴──)──────┘
   │ │        ├─ADDR────────────┤ │   │             ├─STOGENLGN─┤
   │ │        ├─ASR─────────────┤ │   │             └─STOLGN────┘
   │ │        ├─BACKUP──────────┤ │   │
   │ │        ├─BUFSIZE─────────┤ │   │
   │ │        ├─DEFN────────────┤ │   │
   │ │        ├─DEFNTYPE────────┤ │   │
   │ │        ├─GLOBAL,DEFN─────┤ │   │
   │ │        ├─ICONPLKID───────┤ │   │
   │ │        ├─IMSCON──────────┤ │   │
   │ │        ├─IMSID───────────┤ │   │
   │ │        ├─LOCAL,DEFN──────┤ │   │
   │ │        ├─MAXSESS─────────┤ │   │
   │ │        ├─MODETBL─────────┤ │   │
   │ │        ├─MSLINK──────────┤ │   │
   │ │        ├─MSNAME──────────┤ │   │
   │ │        ├─NODE────────────┤ │   │
   │ │        ├─RMTIMS──────────┤ │   │
   │ │        ├─STATUS──────────┤ │   │
   │ │        ├─TIMESTAMP───────┤ │   │
   │ │        └─TYPE────────────┘ │   │
   │ ├─SHOW(EXPORTNEEDED)─────────┤   │
   │ ├─SHOW(MSLINK)───────────────┤   │
   │ └─SHOW(MSNAME)───────────────┘   │

►──┬───────────────────────────────┬─────────────────────────────────────►◄
   │            ┌───,────┐          │
   │            ▼        │          │
   └─TYPE(──┬─CTC──────┬─┴──)───────┘
            ├─MTM──────┤
            ├─TCPIP────┤
            └─VTAM─────┘
```
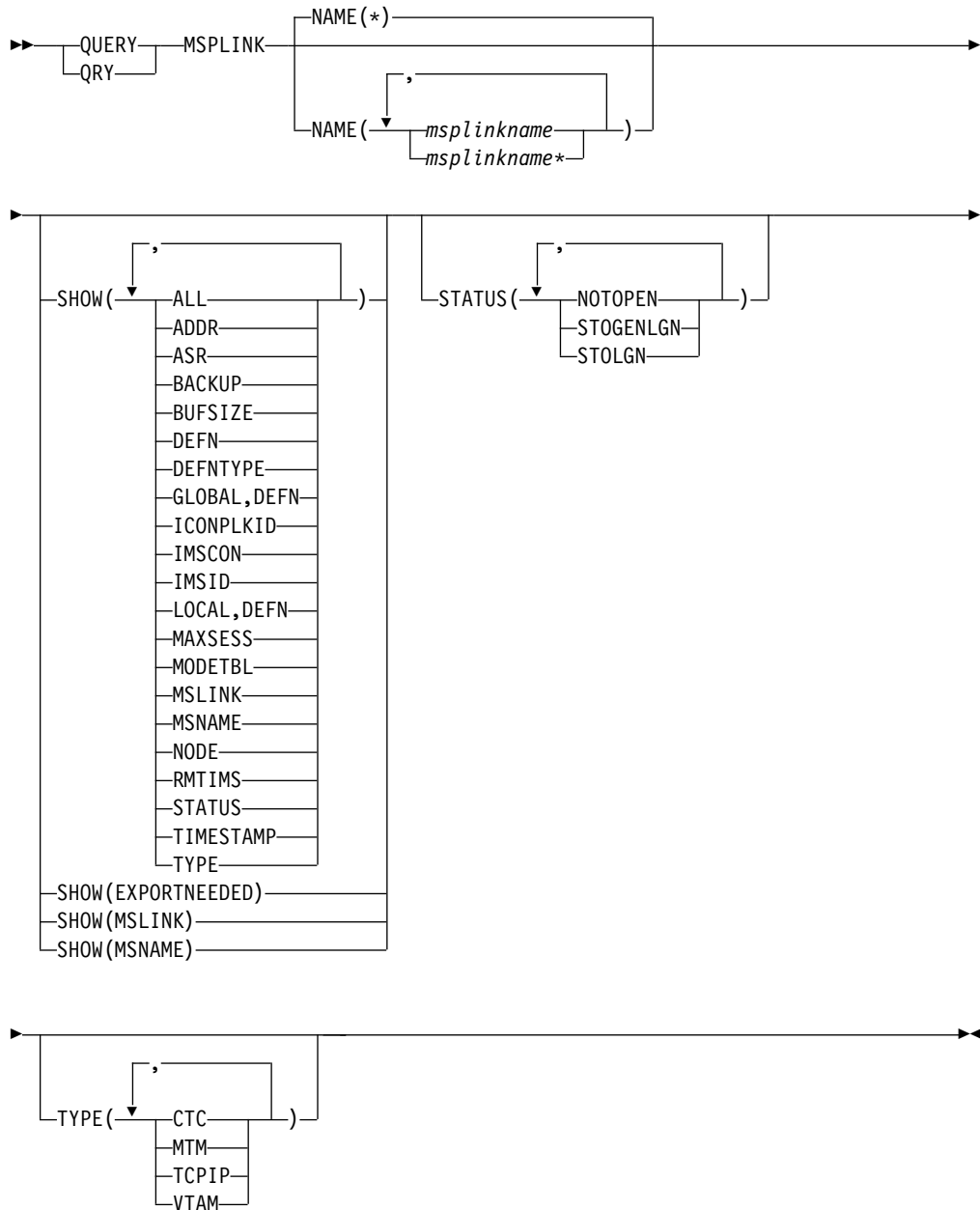
Keywords

The following keywords are valid for the QUERY MSPLINK command:

NAME()
Specifies the 1- to 8-character names of the physical links for which physical link information is to be returned. Wildcards (*) can be specified in the name. The default is NAME(*), which returns all MSPLINK resources.

SHOW()
Specifies the MSPLINK output fields to be returned. If SHOW() is not specified, only the physical names and associated link types are returned.

The MSPLINK name is always returned along with the name of the IMS that created the output for the link and the completion code. The parameters supported with the SHOW keyword are:

ALL
Includes all of the information in the other SHOW parameters (except MSLINK and MSNAME).

ADDR
Displays the address of the channel-to-channel adapter for CTC physical links.

ASR
Displays the Automatic Session Restart value for VTAM links only.

ON For VTAM links that are supported by the VTAM session outage notification (SON) facility, shows that Automatic Session Restart is on.

OFF
Shows that Automatic Session Restart is off, which is the default.

ASR is on only if both sides of the link are using the ASR option.

BACKUP
For XRF-capable IMS systems, displays how the automatic restart of TCP/IP and VTAM links after an XRF takeover are controlled.

BACKUP is an MSC physical link attribute that you can set by using the CREATE MSPLINK, CREATE MSLINK, or UPDATE MSPLINK command. You can display the physical link BACKUP value by using the QUERY MSPLINK and QUERY MSLINK commands.

Important: Even though BACKUP is a physical link attribute, the CREATE MSLINK NAME(*linkname*) SET(MSPLINK(*msplinkname*)) command can also change the BACKUP value of the specified physical link. You can use the QUERY MSPLINK command to display the updated BACKUP value. You can also use the QUERY MSLINK command and specify the MSC logical links, including the newly created logical link, that are assigned to the physical link to display the updated BACKUP value.

The displayed value can be a numeric integer from 1 to 7, inclusive, which is the priority for reestablishing the session that has been set. NO means that the session recovery of the physical link at takeover has been suppressed.

Although BACKUP displays the priority of the order in which IMS restarts TCP/IP and VTAM links, the active requests might be completed in any order because of variables, such as internal VTAM conflicts and pacing, and TCP/IP network traffic.

BUFSIZE
Displays the current input and output buffer size for each logical link that is assigned to this physical link.

DEFN
Specifies that the resource definitions are to be returned.

The MSC physical link attributes that can be returned are: ADDR, ASR, BACKUP, BUFSIZE, ICONPLKID, IMSCON, MAXSESS, MODETBL, NODE, RMTIMS, TYPE, the repository create and update time stamps, and the IMS runtime create and update time stamps.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition that is obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE or STATUS. The LclStat and LDefnType columns, which are returned on the QRY MSPLINK SHOW(ALL) command, are not returned with SHOW(DEFN).
- SHOW(DEFN) cannot be specified with filters STATUS or TYPE.
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).

Resource definitions that are stored in the repository are used to determine the response lines with the repository information, and the runtime resource definitions are used to determine the response lines with the IMS runtime resource information.

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

DEFNTYPE
Displays the definition type that the resource was defined with.

EXPORTNEEDED
Specifies that only the MSC physical link resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command. SHOW(EXPORTNEEDED) behaves like a filter by only

displaying resources that need to be exported, as if QUERY MSPLINK EXPORTNEEDED(Y) were supported.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the MSC physical links that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each MSC physical link name specified on the command. If the specified MSC physical link does not need to be exported to the repository, an N is returned in the ExportNeeded column.

Restrictions:
- SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords.
- SHOW(EXPORTNEEDED) is valid only when IMS is enabled with dynamic resource definition for MSC resources (MSCRSCS=DYN) and the IMSRSC repository.
- The QUERY MSPLINK SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources that have not been exported to the IMSRSC repository, you can issue the /CHECKPOINT command to automatically write the resources to the repository so that they are available for the next IMS cold start.

GLOBAL
For output fields that have both local and global values, this option returns only the global values. GLOBAL must be specified with DEFN, otherwise the command is rejected. If used with another SHOW keyword to request a specific output field, this option requests that only the global value of the specified output field is returned.

SHOW(DEFN,GLOBAL,LOCAL) is equivalent to SHOW(DEFN).

ICONPLKID
For TCP/IP physical links, displays the IMS Connect physical link ID as defined by the LCLPLKID keyword on the MSPLINK macro. The IMS Connect physical link ID is defined to IMS Connect on the LCLPLKID in the IMS Connect MSC configuration statement.

IMSCON
For TCP/IP physical links, displays the IMSplex name of the local IMS Connect instance that manages the TCP/IP connections for the physical link. The IMS Connect IMSplex name is specified on the LCLICON keyword of the MSPLINK macro.

IMSID
Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only when the command master IMS is enabled to use the repository.

IMSID cannot be specified with any other SHOW keywords other than DEFN or GLOBAL.

When SHOW(DEFN,IMSID) is specified, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN keyword, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

LOCAL

For output fields that have both local and global values, this option returns only the local values. LOCAL must be specified with DEFN, otherwise the command is rejected. If used with another SHOW keyword to request a specific output field, this option requests that only the local value of the specified output field is returned. Local output is returned by each IMS that processes the command.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

SHOW(DEFN,GLOBAL,LOCAL) is equivalent to SHOW(DEFN).

MAXSESS

Displays the maximum number of sessions allowed for this physical link.

MODETBL

Displays the name of the VTAM logon mode table entry (logon mode name) containing the SNA bind parameters to be used when a session is established for this terminal.

MSLINK

Displays the logical links that are associated with the specified physical link. When the physical link is VTAM, the maximum number of allowed logical sessions for the physical link is also displayed. When SHOW(MSLINK) is specified, STATUS() is invalid. SHOW(MSNAME) and SHOW(MSLINK) are mutually exclusive with all other SHOW parameters.

MSNAME

Displays the logical link paths that are associated with the specified physical link. The MSNAME names, and the remote and local system identifications are displayed. When SHOW(MSNAME) is specified, STATUS() is invalid. SHOW(MSNAME) and SHOW(MSLINK) are mutually exclusive with all other SHOW parameters.

NODE

Displays the VTAM node name (APPLID) of the remote systems at the other end of the physical links for VTAM physical links.

RMTIMS

For TCP/IP physical links, displays the IMS ID of the remote IMS system, as defined on the NAME keyword of the MSPLINK macro in the remote IMS system.

STATUS

Displays the status of the physical link.

TIMESTAMP

Displays the following timestamps in local time:
- Creation time (TimeCreate)
- Last update time (TimeUpdate)
- Last access time (TimeAccess)
- Last import time (TimeImport)

The timestamps are returned in the format YYYY.JJJ HH:MM:SS:TH:

YYYY
> The year

JJJ
> The Julian day (001 - 365)

HH The hour (01 - 24)

MM The minute (00 - 59)

SS The seconds (00 - 59)

TH The tenths and hundredths of a second (00 - 99)

TYPE
> Displays the type of physical link.

STATUS()
> Displays physical links that display at least one of the specified statuses. You cannot specify the STATUS() parameter with the SHOW(MSLINK) parameter or the SHOW(MSNAME) parameter. When you specify the STATUS keyword, the output returned includes the physical link status, even if you do not specify SHOW(STATUS).

NOTOPEN
> VTAM ACB has not been opened.

STOGENLGN
> Displays all TCP/IP-type physical links with a local status of STOGENLGN. Only physical links that are used in a TCP/IP generic resource group can have a status of STOGENGLN.
>
> STOGENLGN indicates that this physical link is stopped in this IMS system. While the physical link is stopped, no logical links can be started on the physical link in this IMS system and the IMS system cannot accept logical link requests for the TCP/IP generic resource group from a partner IMS system.
>
> The STOGENLGN status of the physical link in this IMS system does not prevent other IMS systems in the TCP/IP generic resource group from starting or accepting logical links on the physical link.
>
> To restart the physical link on this IMS system, route an UPDATE MSPLINK NAME(*linkname*) START(GENLOGON) command to the IMS system.

STOLGN
> Displays TCP/IP and VTAM physical links that have a status of STOLGN. Physical links that are used in a TCP/IP generic resource group cannot have a status of STOLGN.
>
> STOLGN indicates that this physical link is stopped in this IMS system. While the physical link is stopped, no logical links can be started on the physical link in this IMS system and the IMS system cannot accept logical link requests on the physical link from a partner IMS system.

TYPE()
> Displays physical links that possess at least one of the specified types of physical links. When you specify the TYPE keyword, the output returned includes the type of physical link, even if you do not specify SHOW(TYPE).

CTC
> Channel-to-Channel adapter.

MTM
> Memory-to-Memory.

TCPIP
> TCP/IP.

VTAM
> Virtual Telecommunication Access Method.

Usage notes

You can specify this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

If you want to display information about resource definitions, specify SHOW(DEFN). If you want to know which IMS systems have the resource defined and also know the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource defined, specify SHOW(IMSID).

Equivalent IMS type-1 commands

The following table shows variations of the QUERY MSPLINK command and the type-1 IMS commands that perform similar functions.

Table 125. Type-1 equivalents for the QUERY MSPLINK command

QUERY MSPLINK command	Similar IMS type-1 command
QUERY MSPLINK NAME(msplinkname │ *) SHOW(ALL)	No similar type-1 IMS command exists.
QUERY MSPLINK NAME(msplinkname │ *) SHOW(MSLINK │ MSNAME)	/DIS ASSIGNMENT MSPLINK msplinkname │ ALL

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the long label that is generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 126. Output field descriptions for the QUERY MSPLINK command

Short label	Long label	Keyword	Scope	Meaning
ADDR	LCTCaddr	ADDR		Address of the communication line or CTC adapter.

Table 126. Output field descriptions for the QUERY MSPLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
ASR	LASR	ASR		Automatic Session Restart value, which can be ON or OFF.
BKUP	LBackUp	BACKUP		Automatic restart of TCP/IP and VTAM links after an XRF takeover.
BUFSZ	LBufSize	BUFSIZE		Input and output buffer sizes for each logical link that is assigned to this physical link.
CC	CC	N/A		Completion code.
CCTXT	CCText	*error*		Completion code text that briefly explains the meaning of the non-zero completion code.
DFNT	LDefnType	DEFNTYPE		Definition type, which can be one of the following: **CREATE** Defined by CREATE MSPLINK command. **MSCGEN** Defined by system definition in the DFSCLL3x member of the IMS RESLIB. **UPDATE** Defined by system definition in the DFSCLL3x member of the IMS RESLIB, but modified in online memory by an UPDATE command that changed at least one attribute. At the next IMS cold start, the definition reverts back to the definition in the DFSCLL3x member.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the MSC physical link is exported to the IMSRSC repository. The value can be Y or N.

Short label	Long label	Keyword	Scope	Meaning
IMSID	IMSid	IMSID	GBL	Returns from the repository the IMSIDs that have the resource defined.
LIC	LLclImsCon	IMSCON		Displayed for TCP/IP links only. Identifies the local IMS Connect instance within the IMSplex that the physical link connects to by way of SCI. The value shown is defined by the LCLICON keyword on the MSPLINK macro. The LCLICON value matches the value specified on the MEMBER parameter of the IMS Connect MSC configuration statement.
LINKN	LMSLink#	MSLINK		Link number of logical link associated with the physical link.
LPLK	LLclPlkID	ICONPLKID		Displayed for TCP/IP links only. Identifies the ID of the physical link statement within the local IMS Connect instance, as defined by the LCLPLKID keyword on the MSPLINK macro. The LCLPLKID value is also specified on the LCLPLKID keyword in the IMS Connect MSC configuration statement.
LSTT	LclStat	STATUS		Status of the physical link. The status of a physical link can be: STOLGN, STOGENLGN, and NOTOPEN.
MAXS	LMaxSess	MSLINK		Maximum number of sessions allowed.
MDTB	LDefMdtbl	MODETBL		VTAM logon mode table entry.
MSL	LMSLink	MSLINK		Logical links associated with the physical link.
MSN	LMSName	MSNAME		MSNAMEs associated with the physical link.
MSP	MSPLink	N/A		Physical link name.

Table 126. Output field descriptions for the QUERY MSPLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
NODE	LNodeName	NODE		VTAM node name of the remote system at the other end of the physical link.
RADDR	CTCaddr	ADDR, DEFN	GBL	Address of the communication line or CTC adaptor. The output is returned from the repository.
RASR	ASR	ASR, DEFN	GBL	Automatic Session Restart value for VTAM links only. The output is returned from the repository.
RBKUP	BackUp	BACKUP, DEFN	GBL	Automatic restart of TCP/IP and VTAM links after an XRF takeover. The output is returned from the repository.
RBUFSZ	BufSize	BUFSIZE, DEFN	GBL	Input and output buffer sizes for each logical link that is assigned to this physical link. The output is returned from the repository.
REPO	Repo	DEFN	GBL	Indicates whether the line shows the stored resource definitions.
				Y Indicates repository definitions.
				(blank) Indicates local definitions.

Table 126. Output field descriptions for the QUERY MSPLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
RIC	LclImsCon	DEFN, IMSCON	GBL	Displayed for TCP/IP links only. Identifies the local IMS Connect instance within the IMSplex that the physical link connects to by way of SCI. The value shown is defined by the LCLICON keyword on the MSPLINK macro, or the MSPLINK keyword on the CREATE MSPLINK command. The local IMS Connect value matches the value specified on the member parameter of IMS Connect MSC configuration statement. The output is returned from the repository.
RIMS	LRmtIms	RMTIMS		Displayed for TCP/IP links only. Identifies the IMS ID of the remote IMS system that the physical link connects to, as defined by the NAME keyword on the MSPLINK macro.
RMAXS	MaxSess	DEFN, MAXSESS	GBL	Maximum number of sessions allowed. The output is returned from the repository.
RMDTB	DefMdtbl	DEFN, MODETBL	GBL	VTAM logon mode table entry (logon mode name) containing the SNA bind parameters to be used when a session is established for this terminal. The output is returned from the repository.
RMSL	MSLink	DEFN, MSLINK	GBL	Logical links associated with the physical link. The output is returned from the repository.
RMSN	MSName	DEFN, MSNAME		MSNAMEs associated with the physical link. The output is returned from the repository.

Table 126. Output field descriptions for the QUERY MSPLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
RNODE	NodeName	DEFN, NODE	GBL	VTAM node name of the remote system at the other end of the physical link. The output is returned from the repository.
RPLK	LclPlkID	DEFN, ICONPLKID	GBL	Defined for TCP/IP links only. Identifies the ID of the physical link statement within the local IMS Connect instance, as defined by the LCLPLKID keyword on the MSPLINK macro or the MSPLINK keyword on the CREAT MSPLINK command. The LCLPLKID value is also specified on the LCLPLKID keyword in the IMS Connect MSC configuration statement. The output is returned from the repository.
RRIMS	RmtIms	DEFN, RMTIMS	GBL	Displayed for TCP/IP links only. Identifies the IMS ID of the remote IMS system that the physical link connects to, as defined by the NAME keyword on the MSPLINK macro, or the MSPLINK keyword on the CREATE MSPLINK command. The output is returned from the repository.
RSIDL	SIDL	DEFN, MSNAME	GBL	Local system identification of the associated MSNAME. The output is returned from the repository.
RSIDR	SIDR	DEFN, MSNAME	GBL	Remote system identification of the associated MSNAME. The output is returned from the repository.
RTMCR	TimeCreate	DEFN, TIMESTAMP	GBL	Create time from the repository. This is the time the resource was first created in the repository.

Table 126. Output field descriptions for the QUERY MSPLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
RTMUP	TimeUpdate	DEFN, TIMESTAMP	GBL	Update time from the repository. This is the time the resource was last updated in the repository.
RTYPE	Type	DEFN, TYPE	GBL	Type of the physical link: CTC, MTM, TCP/IP, or VTAM.
SIDL	LSIDL	MSNAME		Local system identification of the associated MSNAME.
SIDR	LSIDR	MSNAME		Remote system identification of the associated MSNAME.
TMAC	LTimeAccess	TIMESTAMP		The time that the resource was last accessed. This is not yet supported.
TMCR	LTimeCreate	TIMESTAMP		The time that the resource was created with a CREATE MSPLINK command, an IMPORT command that creates the MSC physical link, or IMS initialization. The create time is retained across IMS warm start, emergency restart, export, and import. The output is returned from the local IMS.
TMIM	LTimeImport	TIMESTAMP		The time that the resource was last imported, if applicable. The local import timestamp is set to be the IMS cold start time, if the resource was automatically imported from the IMSRSC repository during IMS cold start. The import time is retained across IMS warm start and emergency restart. The output is returned from the local IMS.

Table 126. Output field descriptions for the QUERY MSPLINK command (continued)

Short label	Long label	Keyword	Scope	Meaning
TMUP	LTimeUpdate	TIMESTAMP		The last time the attributes of the runtime resource definition were updated as a result of the UPDATE MSPLINK command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
TYPE	LType	TYPE		Type of the physical link: CTC, MTM, TCP/IP, or VTAM.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY MSPLINK command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 127. Return and reason codes for the QUERY MSPLINK command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY MSPLINK command completed successfully.
X'00000004'	X'00001010'	No resources were found to be returned for one of the following reasons: • The resource names specified might be invalid. • There were no resources that match the filter specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002040'	More than 1 filter value is specified on the QUERY MSPLINK command.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.

Table 127. Return and reason codes for the QUERY MSPLINK command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'00004004'	No CQS address space.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'0000402D'	Command flavor is not allowed because dynamic resource definition for MSC resources is not enabled; that is, MSCRSCS=DYN is not defined in the DFSDFxxx member of the IMS PROCLIB data set.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Repository access is denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server (RS) is being shut down.
X'00000010'	X'0000450B'	Repository Server (RS) is not available.
X'00000010'	X'0000450C'	Repository Server (RS) is busy.

Table 127. Return and reason codes for the QUERY MSPLINK command (continued)

Return code	Reason code	Meaning
X'00000010'	X'0000450D'	RM failed to define some of the internal fields related to the IMSRSC repository.
X'00000014'	X'00005004'	The QUERY MSPLINK command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005100'	RM request error.
X'00000014'	X'00005104'	CQS error.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY MSPLINK command is rejected because the resource list is unusable. Contact IBM Software Support and send them the following information: • The output from the command in error • A dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex • A copy of the IMSRSC repository RID and RMD data sets

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 128. Completion codes for the QUERY MSPLINK command

Completion code	Completion code text	Meaning
0		The QUERY MSPLINK command completed successfully for the resource.
10	NO RESOURCES FOUND	MSPLINK name is invalid, or the specified wildcard parameter does not match any resource names.

Example 1 for QUERY MSPLINK command

In this example, four MSC physical links of each link type are displayed, to show most of the possible headers. Some headers do not appear if no physical links of

that type are specified. For example, if no TCP/IP links are specified, no
TCP/IP-only headers, such as LLclPlkID or LLclImsCon, are displayed.

TSO SPOC input:

```
QUERY MSPLINK NAME(PLNKV001,PLNK23C,PLNK21TA,PLNK23M) SHOW(ALL)
```

TSO SPOC output:

(screen 1)
```
MSPLink  MbrName   CC LType     LNodeName LCTCaddr LRmtIms  LLclImsCon
PLNKV001 IMS2       0 VTAM      L6APPL3
PLNK21TA IMS2       0 TCPIP                        IMS1     HWS2
PLNK23C  IMS2       0 CTC
PLNK23M  IMS2       0 MTM
```

(scrolled right to screen 2)
```
MSPLink  MbrName   LLclPlkID LclStat LASR LBackUp LBufSize LMaxSess
PLNKV001 IMS2                STOLGN  OFF       NO     4096        1
PLNK21TA IMS2      MSC21                        4    65536        2
PLNK23C  IMS2                NOTOPEN                  1024
PLNK23M  IMS2                                         1024
```

(scrolled right to screen 3)
```
MSPLink  MbrName   LTimeCreate
PLNKV001 IMS2      2014.283 08:22:18.06
PLNK21TA IMS2      2014.282 16:00:47.10
PLNK23C  IMS2      2014.282 16:00:47.10
PLNK23M  IMS2      2014.282 16:00:47.10
```

(scrolled right to screen 4)
```
MSPLink  MbrName   LTimeUpdate        LTimeAccess        LDefnType
PLNKV001 IMS2                                            CREATE
PLNK21TA IMS2                                            MSCGEN
PLNK23C  IMS2                                            MSCGEN
PLNK23M  IMS2                                            MSCGEN
```

OM API input:

```
CMD(QRY MSPLINK NAME(PLNKV001,PLNK21TA,PLNK23C,PLNK23M) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2014.283 15:29:29.975114</statime>
<stotime>2014.283 15:29:29.996047</stotime>
<staseq>CDE237FCE494A5EC</staseq>
<stoseq>CDE237FCE9B0FE6D</stoseq>
<rqsttkn1>USRT011 10082929</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSPLINK         </kwd>
<input>QRY MSPLINK NAME(PLNKV001,PLNK23C,PLNK21TA,PLNK23M) SHOW(ALL)
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSP" llbl="MSPLink" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
```

```
               len="8" dtype="CHAR" align="left" skipb="no" />
             <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
              len="4" dtype="INT" align="right" skipb="no" />
             <hdr slbl="TYPE" llbl="LType" scope="LCL" sort="n" key="0" scroll="yes"
              len="8" dtype="CHAR" align="left" skipb="no" />
             <hdr slbl="NODE" llbl="LNodeName" scope="LCL" sort="n" key="0"
              scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="ADDR" llbl="LCTCaddr" scope="LCL" sort="n" key="0"
              scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="RIMS" llbl="LRmtIms" scope="LCL" sort="n" key="0"
              scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="LIC" llbl="LLclImsCon" scope="LCL" sort="n" key="0"
              scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="LPLK" llbl="LLclPlkID" scope="LCL" sort="n" key="0"
              scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="XCFG" llbl="LXCFgroup" scope="LCL" sort="n" key="0"
              scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="XCFM" llbl="LXCFmem" scope="LCL" sort="n" key="0"
              scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
              scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
             <hdr slbl="ASR" llbl="LASR" scope="LCL" sort="n" key="0" scroll="yes"
              len="3" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="BKUP" llbl="LBackUp" scope="LCL" sort="n" key="0"
              scroll="yes" len="2" dtype="CHAR" align="right" skipb="yes" />
             <hdr slbl="BUFSZ" llbl="LBufSize" scope="LCL" sort="n" key="0"
              scroll="yes" len="5" dtype="INT" align="right" skipb="no" />
             <hdr slbl="MAXS" llbl="LMaxSess" scope="LCL" sort="n" key="0"
              scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
             <hdr slbl="MDTB" llbl="LDefMdtbl" scope="LCL" sort="n" key="0"
              scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
              scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
             <hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
              scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
             <hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
              scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
             <hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
              scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
             <hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
              scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
           </cmdrsphdr>
           <cmdrspdata>
           <rsp>MSP(PLNKV001) MBR(IMS2   ) CC(   0) TYPE(VTAM ) NODE(L6APPL3 )
            LSTT(STOLGN) MAXS(   1) ASR(OFF) BKUP(NO) BUFSZ( 4096) TMCR(2014.283
            08:22:18.06) DFNT(CREATE) </rsp>
           <rsp>MSP(PLNK23C ) MBR(IMS2   ) CC(   0) TYPE(CTC  ) ADDR(    )
            LSTT(NOTOPEN) BUFSZ( 1024) TMCR(2014.282 16:00:47.10) DFNT(MSCGEN)
            </rsp>
           <rsp>MSP(PLNK21TA) MBR(IMS2   ) CC(   0) TYPE(TCPIP) RIMS(IMS1    )
            LIC(HWS2    ) LPLK(MSC21  ) MAXS(   2) BKUP( 4) BUFSZ(65536)
            TMCR(2014.282 16:00:47.10) DFNT(MSCGEN) </rsp>
           <rsp>MSP(PLNK23M ) MBR(IMS2   ) CC(   0) TYPE(MTM  ) BUFSZ( 1024)
            TMCR(2014.282 16:00:47.10) DFNT(MSCGEN) </rsp>
           </cmdrspdata>
           </imsout>
```

Example 2 for QUERY MSPLINK command

In this example, four MSC physical links of each link type are displayed, to show
the MSLink and MSName output fields.

TSO SPOC input:
```
QUERY MSPLINK NAME(PLNKV001,PLNK23C,PLNK21TA,PLNK23M) SHOW(MSLINK,MSNAME)
```

(screen 1)

MSPLink	MbrName	CC	LMSLink	LMSLink#	LMaxSess	LMSName	LSIDR	LSIDL
PLNKV001	IMS2	0			1			
PLNK21TA	IMS2	0			2			
PLNK21TA	IMS2	0	LNK21T01	27		LINK21T1	74	84
PLNK21TA	IMS2	0	LNK21T02	28		LINK21T2	75	85
PLNK21TA	IMS2	0	LNK21T02	28		LINK23T2	120	97
PLNK23C	IMS2	0	LNK23C01	5		LINK23C2	33	23
PLNK23M	IMS2	0	LNK23M01	6		LINK23M2	34	24

OM API input:

```
CMD(QRY MSPLINK NAME(PLNKV001,PLNK21TA,PLNK23C,PLNK23M) SHOW(MSLINK,MSNAME))
```

OM API output:

```
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2014.283 15:38:25.282367</statime>
<stotime>2014.283 15:38:25.300469</stotime>
<staseq>CDE239FB66D3F7EB</staseq>
<stoseq>CDE239FB6B3F5CED</stoseq>
<rqsttkn1>USRT011 10083825</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSPLINK         </kwd>
<input>QRY MSPLINK NAME(PLNKV001,PLNK23C,PLNK21TA,PLNK23M)
 SHOW(MSLINK,MSNAME) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSP" llbl="MSPLink" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MSL" llbl="LMSLink" scope="LCL" sort="a" key="2"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LINKN" llbl="LMSLink#" scope="LCL" sort="a" key="3"
 scroll="yes" len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="MAXS" llbl="LMaxSess" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="MSN" llbl="LMSName" scope="LCL" sort="a" key="4"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SIDR" llbl="LSIDR" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="SIDL" llbl="LSIDL" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSP(PLNKV001) MBR(IMS2    ) CC(   0) MAXS(   1) </rsp>
<rsp>MSP(PLNK23C ) MBR(IMS2    ) CC(   0) MSL(LNK23C01) LINKN(    5)
 MSN(LINK23C2) SIDR(  33) SIDL(  23) </rsp>
<rsp>MSP(PLNK21TA) MBR(IMS2    ) CC(   0) MAXS(   2) </rsp>
<rsp>MSP(PLNK21TA) MBR(IMS2    ) CC(   0) MSL(LNK21T01) LINKN(  27)
 MSN(LINK21T1) SIDR(  74) SIDL(  84) </rsp>
<rsp>MSP(PLNK21TA) MBR(IMS2    ) CC(   0) MSL(LNK21T02) LINKN(  28)
 MSN(LINK21T2) SIDR(  75) SIDL(  85) </rsp>
<rsp>MSP(PLNK21TA) MBR(IMS2    ) CC(   0) MSL(LNK21T02) LINKN(  28)
```

```
 MSN(LINK23T2) SIDR( 120) SIDL(  97) </rsp>
<rsp>MSP(PLNK23M ) MBR(IMS2    ) CC(   0) MSL(LNK23M01) LINKN(   6)
 MSN(LINK23M2) SIDR(  34) SIDL(  24) </rsp>
</cmdrspdata>
</imsout>
```

Example 3 for QUERY MSPLINK command

In this example, all physical links that are defined with TYPE=TCPIP are
displayed.

TSO SPOC input:

```
QUERY MSPLINK TYPE(TCPIP) SHOW(ALL)
```

TSO SPOC output:

(screen 1)
```
MSPLink  MbrName    CC LType    LRmtIms  LLclImsCon LLclPlkID LclStat LBackUp
PLNK21TA IMS2        0 TCPIP    IMS1     HWS2       MSC21     STOLGN        4
PLNK23TA IMS2        0 TCPIP    IMS3     HWS2       MSC23                   4
```

(scrolled right to screen 2)
```
MSPLink  MbrName  LBufSize LMaxSess LTimeCreate
PLNK21TA IMS2        65536        2 2014.282 16:00:47.10
PLNK23TA IMS2        65536        2 2014.282 16:00:47.10
```

(scrolled right to screen 3)
```
MSPLink  MbrName LTimeUpdate           LTimeAccess           LDefnType
PLNK21TA IMS2                                                MSCGEN
PLNK23TA IMS2
```

OM API input:

```
CMD(QRY MSPLINK TYPE(TCPIP) SHOW(ALL))
```

OM API output:

```
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2014.283 15:54:36.419187</statime>
<stotime>2014.283 15:54:36.487405</stotime>
<staseq>CDE23D998CC73DF1</staseq>
<stoseq>CDE23D999D6ED2E6</stoseq>
<rqsttkn1>USRT011 10085434</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSPLINK         </kwd>
<input>QUERY MSPLINK TYPE(TCPIP) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSP" llbl="MSPLink" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="TYPE" llbl="LType" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="NODE" llbl="LNodeName" scope="LCL" sort="n" key="0"
```

```
                   scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
               <hdr slbl="ADDR" llbl="LCTCaddr" scope="LCL" sort="n" key="0"
                scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
               <hdr slbl="RIMS" llbl="LRmtIms" scope="LCL" sort="n" key="0"
                scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
               <hdr slbl="LIC" llbl="LLclImsCon" scope="LCL" sort="n" key="0"
                scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
               <hdr slbl="LPLK" llbl="LLclPlkID" scope="LCL" sort="n" key="0"
                scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
               <hdr slbl="XCFG" llbl="LXCFgroup" scope="LCL" sort="n" key="0"
                scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
               <hdr slbl="XCFM" llbl="LXCFmem" scope="LCL" sort="n" key="0"
                scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
               <hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
                scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
               <hdr slbl="ASR" llbl="LASR" scope="LCL" sort="n" key="0" scroll="yes"
                len="3" dtype="CHAR" align="left" skipb="yes" />
               <hdr slbl="BKUP" llbl="LBackUp" scope="LCL" sort="n" key="0"
                scroll="yes" len="2" dtype="CHAR" align="right" skipb="yes" />
               <hdr slbl="BUFSZ" llbl="LBufSize" scope="LCL" sort="n" key="0"
                scroll="yes" len="5" dtype="INT" align="right" skipb="no" />
               <hdr slbl="MAXS" llbl="LMaxSess" scope="LCL" sort="n" key="0"
                scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
                <hdr slbl="MDTB" llbl="LDefMdtbl" scope="LCL" sort="n" key="0"
                 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
                <hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
                 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
                <hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
                 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
                <hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
                 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
                <hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
                 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
               <hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
                scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
               </cmdrsphdr>
               <cmdrspdata>
               <rsp>MSP(PLNK21TA) MBR(IMS2    ) CC(   0) TYPE(TCPIP) RIMS(IMS1    )
                LIC(HWS2    ) LPLK(MSC21   ) LSTT(STOLGN) MAXS(   2) BKUP( 4)
                BUFSZ(65536) TMCR(2014.282 16:00:47.10) DFNT(MSCGEN) </rsp>
               <rsp>MSP(PLNK23TA) MBR(IMS2    ) CC(   0) TYPE(TCPIP) RIMS(IMS3    )
                LIC(HWS2    ) LPLK(MSC23   ) MAXS(   2) BKUP( 4) BUFSZ(65536)
                TMCR(2014.282 16:00:47.10) DFNT(MSCGEN) </rsp>
               </cmdrspdata>
               </imsout>
```

Example 4 for QUERY MSPLINK command

In this example, all attributes are displayed for all MSC physical links that have a
status of STOLGN (logons stopped).

TSO SPOC input:
```
QUERY MSPLINK STATUS(STOLGN) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
MSPLink  MbrName    CC LType     LNodeName LRmtIms  LLclImsCon LLclPlkID
PLNKV001 IMS2        0 VTAM      L6APPL3
PLNK21TA IMS2        0 TCPIP               IMS1     HWS2       MSC21

(scrolled right to screen 2)
MSPLink  MbrName    LclStat LASR LBackUp LBufSize LMaxSess LTimeCreate
PLNKV001 IMS2       STOLGN  OFF       NO   4096         1 2014.283 08:22:18.06
PLNK21TA IMS2       STOLGN            4   65536         2 2014.282 16:00:47.10
```

```
(scrolled right to screen 3)
MSPLink  MbrName  LTimeUpdate            LTimeAccess            LDefnType
PLNKV001 IMS2                                                   CREATE
PLNK21TA IMS2                                                   MSCGEN
```

OM API input:

```
CMD(QRY MSPLINK STATUS(STOLGN) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2014.283 15:50:05.165574</statime>
<stotime>2014.283 15:50:05.189920</stotime>
<staseq>CDE23C96DCC06FF0</staseq>
<stoseq>CDE23C96E2B208F2</stoseq>
<rqsttkn1>USRT011 10085004</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>MSPLINK         </kwd>
<input>QRY MSPLINK STATUS(STOLGN) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSP" llbl="MSPLink" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="TYPE" llbl="LType" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="NODE" llbl="LNodeName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ADDR" llbl="LCTCaddr" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RIMS" llbl="LRmtIms" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LIC" llbl="LLclImsCon" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LPLK" llbl="LLclPlkID" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="XCFG" llbl="LXCFgroup" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="XCFM" llbl="LXCFmem" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="ASR" llbl="LASR" scope="LCL" sort="n" key="0" scroll="yes"
 len="3" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="BKUP" llbl="LBackUp" scope="LCL" sort="n" key="0"
 scroll="yes" len="2" dtype="CHAR" align="right" skipb="yes" />
<hdr slbl="BUFSZ" llbl="LBufSize" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" skipb="no" />
<hdr slbl="MAXS" llbl="LMaxSess" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
<hdr slbl="MDTB" llbl="LDefMdtbl" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
```

```
<hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSP(PLNKV001) MBR(IMS2   ) CC(   0) TYPE(VTAM ) NODE(L6APPL3 )
 LSTT(STOLGN) MAXS(   1) ASR(OFF) BKUP(NO) BUFSZ( 4096) TMCR(2014.283
 08:22:18.06) DFNT(CREATE) </rsp>
<rsp>MSP(PLNK21TA) MBR(IMS2   ) CC(   0) TYPE(TCPIP) RIMS(IMS1    )
 LIC(HWS2    ) LPLK(MSC21  ) LSTT(STOLGN) MAXS(   2) BKUP( 4)
 BUFSZ(65536) TMCR(2014.282 16:00:47.10) DFNT(MSCGEN) </rsp>
</cmdrspdata>
</imsout>
```

Example 5 for QUERY MSPLINK command

TSO SPOC input:

```
QUERY MSPLINK SHOW(EXPORTNEEDED)
```

TSO SPOC output:

```
MSPLink  MbrName    CC ExportNeeded
PLNKM001 IMS1        0 Y
PLNKM002 IMS1        0 Y
PLNKM003 IMS1        0 Y
```

Example 6 for QUERY MSPLINK command

TSO SPOC input:

```
QUERY MSPLINK NAME(PLNK23M,PLNK23C,PLNK23V,PLNK31TA,PLNKV001) SHOW(DEFN)
```

TSO SPOC output:

```
(screen 1)
MSPLink  MbrName    CC Repo IMSid Type      LType     NodeName LNodeName
PLNKV001 IMS1        0 Y          VTAM                L6APPL1
PLNKV001 IMS1        0    IMS1              VTAM                L6APPL1
PLNK23C  IMS1        0 Y          CTC
PLNK23C  IMS1        0    IMS1              CTC
PLNK23M  IMS1        0 Y          MTM
PLNK23M  IMS1        0    IMS1              MTM
PLNK23V  IMS1        0 Y          VTAM                L6APPL1
PLNK23V  IMS1        0    IMS1              VTAM                L6APPL1
PLNK31TA IMS1        0 Y          TCPIP
PLNK31TA IMS1        0    IMS1              TCPIP

(screen 2)
MSPLink  MbrName    Repo CTCaddr  LCTCaddr RmtIms   LRmtIms  LclImsCon LLclImsCon
PLNKV001 IMS1        Y
PLNKV001 IMS1
PLNK23C  IMS1        Y
PLNK23C  IMS1
PLNK23M  IMS1        Y
PLNK23M  IMS1
PLNK23V  IMS1        Y
PLNK23V  IMS1
PLNK31TA IMS1        Y                      IMS1              HWS3
PLNK31TA IMS1                                        IMS1               HWS3

(screen 3)
```

MSPLink	MbrName	Repo	LclPlkID	LLclPlkID	ASR	LASR	BackUp	LBackUp
PLNKV001	IMS1	Y			OFF		NO	
PLNKV001	IMS1					OFF		NO
PLNK23C	IMS1	Y						
PLNK23C	IMS1							
PLNK23M	IMS1	Y						
PLNK23M	IMS1							
PLNK23V	IMS1	Y			OFF		NO	
PLNK23V	IMS1					OFF		NO
PLNK31TA	IMS1	Y	MSC31				NO	
PLNK31TA	IMS1			MSC31				NO

(screen 4)

MSPLink	MbrName	Repo	BufSize	LBufSize	MaxSess	LMaxSess	DefMdtbl	LDefMdtbl
PLNKV001	IMS1	Y	4096		1		MTMSCVAA	
PLNKV001	IMS1			4096		1		MTMSCVAA
PLNK23C	IMS1	Y	1024					
PLNK23C	IMS1			1024				
PLNK23M	IMS1	Y	1024					
PLNK23M	IMS1			1024				
PLNK23V	IMS1	Y	1024		8			
PLNK23V	IMS1			1024		8		
PLNK31TA	IMS1	Y	65536		2			
PLNK31TA	IMS1			65536		2		

(screen 5)

MSPLink	MbrName	Repo	TimeCreate	LTimeCreate
PLNKV001	IMS1	Y	2015.308 10:43:15.54	
PLNKV001	IMS1			2015.308 10:43:15.54
PLNK23C	IMS1	Y	2015.308 10:42:16.99	
PLNK23C	IMS1			2015.308 10:42:16.99
PLNK23M	IMS1	Y	2015.308 10:42:16.99	
PLNK23M	IMS1			2015.308 10:42:16.99
PLNK23V	IMS1	Y	2015.308 10:42:16.99	
PLNK23V	IMS1			2015.308 10:42:16.99
PLNK31TA	IMS1	Y	2015.308 10:42:16.99	
PLNK31TA	IMS1			2015.308 10:42:16.99

(screen 6)

MSPLink	MbrName	Repo	TimeUpdate	LTimeUpdate	LTimeAccess
PLNKV001	IMS1	Y			
PLNKV001	IMS1				
PLNK23C	IMS1	Y			
PLNK23C	IMS1				
PLNK23M	IMS1	Y			
PLNK23M	IMS1				
PLNK23V	IMS1	Y			
PLNK23V	IMS1				
PLNK31TA	IMS1	Y			
PLNK31TA	IMS1				

OM API input:

```
CMD(QRY MSPLINK NAME(PLNK23M,PLNK23C,PLNK23V,PLNK31TA,PLNKV001) SHOW(DEFN)
```

OM API output:

```
<imsout>
<cmd>
 <master>IMS1     </master>
 <userid>USRT011 </userid>
 <verb>QRY </verb>
 <kwd>MSPLINK          </kwd>
 <input>QRY MSPLINK NAME(PLNK23M,PLNK23C,PLNK23V,PLNK31TA,PLNKV001)
  SHOW(DEFN) </input>
 </cmd>
 <cmdrsphdr>
 <hdr slbl="MSP" llbl="MSPLink" scope="LCL" sort="a" key="1" scroll="no"
```

```
|                                     len="8" dtype="CHAR" align="left" skipb="no" />
|                                    <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
|                                     len="8" dtype="CHAR" align="left" skipb="no" />
|                                    <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
|                                     len="4" dtype="INT" align="right" skipb="no" />
|                                    <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
|                                     len="1" dtype="CHAR" align="left" />
|                                    <hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="RTYPE" llbl="Type" scope="GBL" sort="n" key="0" scroll="yes"
|                                     len="8" dtype="CHAR" align="left" skipb="no" />
|                                    <hdr slbl="TYPE" llbl="LType" scope="LCL" sort="n" key="0" scroll="yes"
|                                     len="8" dtype="CHAR" align="left" skipb="no" />
|                                    <hdr slbl="RNODE" llbl="NodeName" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="NODE" llbl="LNodeName" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="RADDR" llbl="CTCaddr" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="ADDR" llbl="LCTCaddr" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="RRIMS" llbl="RmtIms" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="RIMS" llbl="LRmtIms" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="RIC" llbl="LclImsCon" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="LIC" llbl="LLclImsCon" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="RPLK" llbl="LclPlkID" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="LPLK" llbl="LLclPlkID" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="XCFG" llbl="LXCFgroup" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="XCFM" llbl="LXCFmem" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="RASR" llbl="ASR" scope="GBL" sort="n" key="0" scroll="yes"
|                                     len="3" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="ASR" llbl="LASR" scope="LCL" sort="n" key="0" scroll="yes"
|                                     len="3" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="RBKUP" llbl="BackUp" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="2" dtype="CHAR" align="right" skipb="yes" />
|                                    <hdr slbl="BKUP" llbl="LBackUp" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="2" dtype="CHAR" align="right" skipb="yes" />
|                                    <hdr slbl="RBUFSZ" llbl="BufSize" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="5" dtype="INT" align="right" skipb="no" />
|                                    <hdr slbl="BUFSZ" llbl="LBufSize" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="5" dtype="INT" align="right" skipb="no" />
|                                    <hdr slbl="RMAXS" llbl="MaxSess" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
|                                    <hdr slbl="MAXS" llbl="LMaxSess" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="4" dtype="INT" align="right" skipb="yes" />
|                                    <hdr slbl="RMDTB" llbl="DefMdtbl" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="MDTB" llbl="LDefMdtbl" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
|                                    <hdr slbl="RTMCR" llbl="TimeCreate" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
|                                    <hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
|                                    <hdr slbl="RTMUP" llbl="TimeUpdate" scope="GBL" sort="n" key="0"
|                                     scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
|                                    <hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
|                                     scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
```

```
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="EXPN" llbl="ExportNeeded" scope="LCL" sort="y" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSP(PLNK23M ) MBR(IMS1    ) CC(   0) TYPE(MTM ) BUFSZ( 1024)
TMCR(2015.308 10:42:16.99) DFNT(MSCGEN) IMSID(IMS1) </rsp>
<rsp>MSP(PLNK23C ) MBR(IMS1    ) CC(   0) TYPE(CTC ) ADDR(    )
 LSTT(NOTOPEN) BUFSZ( 1024) TMCR(2015.308 10:42:16.99) DFNT(MSCGEN)
 IMSID(IMS1) </rsp>
<rsp>MSP(PLNK23V ) MBR(IMS1    ) CC(   0) TYPE(VTAM ) NODE(L6APPL1 )
 MAXS(   8) ASR(OFF) BKUP(NO) BUFSZ( 1024) TMCR(2015.308 10:42:16.99)
 DFNT(MSCGEN) IMSID(IMS1) </rsp>
<rsp>MSP(PLNK31TA) MBR(IMS1    ) CC(   0) TYPE(TCPIP) RIMS(IMS1    )
 LIC(HWS3    ) LPLK(MSC31  ) MAXS(   2) BKUP(NO) BUFSZ(65536)
 TMCR(2015.308 10:42:16.99) DFNT(MSCGEN) IMSID(IMS1) </rsp>
<rsp>MSP(PLNKV001) MBR(IMS1    ) CC(   0) TYPE(VTAM ) NODE(L6APPL1 )
 LSTT(STOLGN) MAXS(   1) ASR(OFF) BKUP(NO) BUFSZ( 4096) MDTB(MTMSCVAA)
 TMCR(2015.308 10:43:15.54) DFNT(CREATE) IMSID(IMS1) </rsp>
<rsp>MSP(PLNK23M ) MBR(IMS1) CC(   0) REPO(Y) IMSID() RBUFSZ( 1024)
 RTMCR(2015.308 10:42:16.99) RTYPE(MTM ) </rsp>
<rsp>MSP(PLNK23C ) MBR(IMS1) CC(   0) REPO(Y) IMSID() RADDR(    )
 RBUFSZ( 1024) RTMCR(2015.308 10:42:16.99) RTYPE(CTC ) </rsp>
<rsp>MSP(PLNK23V ) MBR(IMS1) CC(   0) REPO(Y) IMSID() RASR(OFF)
 RBKUP(NO) RBUFSZ( 1024) RMAXS(   8) RMDTB(          ) RNODE(L6APPL1 )
 RTMCR(2015.308 10:42:16.99) RTYPE(VTAM ) </rsp>
<rsp>MSP(PLNK31TA) MBR(IMS1) CC(   0) REPO(Y) IMSID() RBKUP(NO)
 RBUFSZ(65536) RPLK(MSC31  ) RIC(HWS3    ) RMAXS(   2) RRIMS(IMS1    )
 RTMCR(2015.308 10:42:16.99) RTYPE(TCPIP) </rsp>
<rsp>MSP(PLNKV001) MBR(IMS1) CC(   0) REPO(Y) IMSID() RASR(OFF)
 RBKUP(NO) RBUFSZ( 4096) RMAXS(   1) RMDTB(MTMSCVAA) RNODE(L6APPL1 )
 RTMCR(2015.308 10:43:15.54) RTYPE(VTAM ) </rsp>
</cmdrspdata>

<imsout>
```

Explanation: This QUERY command displays all four MSC physical link types, so
that all possible local and repository definition headers are displayed for all eleven
MSPLINK attributes.

Related concepts:

How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

Command keywords and their synonyms (Commands)

List of commands with similar functions for multiple resources (Operations
and Automation)

QUERY NODE command

Use the QUERY NODE command to display information about ISC TCP/IP or
VTAM nodes (terminals) and non-VTAM devices (SPOOL and SYSOUT devices) in
an IMSplex.

For ISC TCP/IP nodes, the command also displays the IMS Connect instance that
provides the TCP/IP support.

This command can be specified only through the OM API and is valid on an XRF alternate.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 424
- "Usage notes" on page 430
- "Equivalent IMS type-1 commands" on page 430
- "Output fields" on page 431
- "QUERY NODE status" on page 436
- "Return, reason, and completion codes" on page 439
- "Examples" on page 441

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 129. Valid environments for the QUERY NODE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY NODE	X		X
NAME	X		X
USER	X		X
SHOW	X		X
STATUS	X		X

Syntax

```
 ►──┬─SHOW(GLOBAL,LOCAL)──────────────────────┬──┬──────────────────────────────────┬──►◄
    │                                          │  │           ┌─,──────────┐          │
    │        ┌─GLOBAL,LOCAL,─┐                 │  └─STATUS(───▼─┬─ACTIVE──────┬──)────┘
    ├─SHOW(──┴───────────────┴──┤ A ├──)───────┤              ├─ASR─────────┤
    │        ┌─,───────────┐                   │              ├─BACKUP──────┤
    └─SHOW(──▼──┬─GLOBAL─┬──┴──┬──────────┬─)──┘              ├─CINOP───────┤
               └─LOCAL──┘     └─,─┤ A ├─┘                    ├─CLSDST──────┤
                                                             ├─COLD────────┤
                                                             ├─CON─────────┤
                                                             ├─CONV────────┤
                                                             ├─DEACT───────┤
                                                             ├─EXCL────────┤
                                                             ├─FORCSESS────┤
                                                             ├─IDLE────────┤
                                                             ├─INOP────────┤
                                                             ├─LOCK────────┤
                                                             ├─LOST────────┤
                                                             ├─MFST────────┤
                                                             ├─OPNDST──────┤
                                                             ├─PAGE────────┤
                                                             ├─PRI─────────┤
                                                             ├─PRST────────┤
                                                             ├─QUI─────────┤
                                                             ├─RELREQ──────┤
                                                             ├─RESP────────┤
                                                             ├─RESPINP─────┤
                                                             ├─RESPINPFP───┤
                                                             ├─RESYNC──────┤
                                                             ├─RM──────────┤
                                                             ├─RMACTIVE────┤
                                                             ├─RMOWNED─────┤
                                                             ├─SEC─────────┤
                                                             ├─SHUT────────┤
                                                             ├─SIGN────────┤
                                                             ├─SIMLOGON────┤
                                                             ├─STATIC──────┤
                                                             ├─STOLGN──────┤
                                                             ├─TEST────────┤
                                                             ├─TKOTRA──────┤
                                                             └─TRACE───────┘
```

A:

```
          ┌──────────,◄──────────┐
          ▼                       │
├──────────┬─ALL───────┬─────────┴────────────────────────────────────────┤
           ├─AFFIN─────┤
           ├─CICSAPPL──┤
           ├─CICSNETID─┤
           ├─CID───────┤
           ├─CONV──────┤
           ├─COUNT─────┤
           ├─EMHQ──────┤
           ├─ID────────┤
           ├─IMSCON────┤
           ├─LOGOND────┤
           ├─LTERM─────┤
           ├─MODETBL───┤
           ├─OWNER─────┤
           ├─PRESET────┤
           ├─QCNT──────┤
           ├─RECOVERY──┤
           ├─STATUS────┤
           ├─TYPE──────┤
           ├─USER──────┤
           ├─USERID────┤
           └─VERSION───┘
```

Keywords

The following keywords are valid for the QUERY NODE command:

NAME()

Specifies the names of one or more VTAM nodes (terminals) or non-VTAM devices that are to be displayed. Valid names are 1-8 characters, and wildcards can be specified. To display all nodes and devices, specify NAME(*). NAME(*) is the default.

Any static or dynamic VTAM terminal name can be specified. Supported terminal types are LU0, LU1, LU2, and LU6.1 (ISC only).

Any non-VTAM device can be specified. Non-VTAM devices are assigned a LINE and PTERM number during the system definition process. Supported device types are CONSOLE, SYSOUT (DISK, PUNCH, PRINTER, READER, TAPE), SPOOL, and TCO:

CONSOLE

This is the system console. IMS automatically assigns LINE 1 PTERM 1 to this device.

DISK Defined by the LINEGRP macro, UNITYPE=DISK.

PUNCH

Defined by the LINEGRP macro, UNITYPE=PUNCH.

PRINTER

Defined by the LINEGRP macro, UNITYPE=PRINTER.

READER

Defined by the LINEGRP macro, UNITYPE=READER.

SPOOL

Defined by the LINEGRP macro, UNITYPE=SPOOL.

TAPE Defined by the LINEGRP macro, UNITYPE=TAPE.

TCO This is the device assigned as the Time Control Option. The system definition process automatically assigns the last LINE number to this device.

To specify a non-VTAM device, specify a name of DFSLN*xxx*, where *xxx* is the LINE number of the device. For example, to display the system console (LINE 1), specify NAME(DFSLN001).

SHOW()
Specifies the node output fields to be returned.

The node name is always returned along with the name of the IMS that created the output and the completion code. If SHOW is not specified, only the node names are returned, provided that the STATUS filter is not specified. This provides a method for a system management application to obtain a list of all node names that are currently known in the IMSplex.

Two parameters, GLOBAL and LOCAL, are used to specify the location (global resources or local resources) where IMS should obtain the information that is to be displayed. The default is both GLOBAL and LOCAL.

The rest of the parameters are used to specify what information is displayed.

The parameters supported with the SHOW keyword, which can be specified in any order, are:

ALL
Returns all of the output fields, except for those fields displayed when the LTERM and CONV parameters are specified. To display LTERM and conversation information with all other output, specify SHOW(ALL,LTERM,CONV).

AFFIN
IMS APPLID to which the node has VTAM generic resource affinity, if applicable. VTAM generic resource affinity is valid only when the GLOBAL parameter is specified on the SHOW keyword. If GLOBAL is not specified, then the AFFIN parameter is ignored.

CICSAPPL
For ISC TCP/IP nodes, displays the APPLID of the remote CICS subsystem.

CICSNETID
For ISC TCP/IP nodes, displays the NETWORK ID of the remote CICS.

CID
VTAM connection identifier.

CONV
The conversation ID, transaction, and conversation status associated with the node. Each conversation is returned on a separate command response display line. Conversation status is not included when SHOW(ALL) is specified. To display conversation information with all other output, specify SHOW(ALL,CONV).

COUNT
Displays the number of messages sent to and received from the specified node.

EMHQ
Displays the node message queue count in the Expedited Message Handler (EMH) queues. The queue count is the sum of the queue counts for each logical terminal associated with the specified node.

EMHQ is valid only when the GLOBAL parameter is specified on the SHOW keyword. If GLOBAL is not specified, then the EMHQ parameter is ignored.

EMHQ is processed by the command master only. It is ignored by all other IMS systems.

EMHQ is valid only if shared EMH is used in a shared queues environment. Otherwise, this parameter is ignored.

If the node resource exists in the resource structure, then the logical terminals associated with the global node resource are used to obtain the queue counts from the EMH queues. Otherwise, the logical terminals associated with the local node resource are used, but if the node does not exist locally on the command master, then the queue count is 0.

GLOBAL
If GLOBAL is specified, the command master displays global information, depending on the other SHOW parameters specified. This includes information from the resource structure.

The GLOBAL parameter is processed by the command master only. All other IMS systems ignore this parameter. If LOCAL is not also specified, then all IMS systems other than the command master ignore the command.

GLOBAL is applicable only when the command master is using shared queues, VTAM generic resources, sysplex terminal management, or any combination of these. GLOBAL is not applicable when the command master is not using shared queues, VTAM generic resources, or sysplex terminal management. In this environment, if LOCAL is also specified, then GLOBAL is ignored. Otherwise, the command master rejects the command.

If shared queues are enabled, and global queue counts are requested, then the command master will make requests to CQS to determine the appropriate queue counts. This includes both MSGQ and EMHQ.

If VTAM generic resources (VGR) is enabled, and VGR affinity information is requested, then the command master will make requests to VTAM to determine any VGR affinity.

If sysplex terminal management is enabled, then the command master will make requests to RM to determine the appropriate global status.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

ID Displays the other half-session qualifier name of the ISC node.

IMSCON
For ISC TCP/IP nodes, shows the IMSplex member name of the IMS Connect instance that provides TCP/IP support for the node.

LOCAL
If LOCAL is specified, then all IMS systems including the command master display local information, depending on the other SHOW parameters specified. This includes information local to the IMS processing the command.

The LOCAL parameter is processed by all IMS systems, including the command master. LOCAL is applicable in any environment, regardless of whether shared queues or sysplex terminal management are used.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

LOGOND

For ISC TCP/IP nodes, displays the name of the ETO logon descriptor that was used to establish the session in use by the node.

LTERM

Displays the logical terminal (LTERM) names, if any, associated with the node. A node may have zero or more logical terminals associated with it.

Each logical terminal associated with the node is returned on a separate command response line.

The LTERM status is not included when SHOW(ALL) is specified. To display logical terminal information with all other output, specify SHOW(ALL,LTERM).

MODETBL

Displays the mode table names that are associated with the specified nodes. The output includes both the default mode table name and the actual mode table name. The default mode table name is the default name that is set by system definition. This name can be modified by the UPDATE NODE command, or by the LOGON exit. The actual mode table name is the actual name used to initiate the session. This name is displayed only while the session is active, and is blank at all other times.

OWNER

Displays the owner of the node resource in the resource structure. This applies only when sysplex terminal management is enabled, and is only processed by the command master. All other IMS systems ignore this parameter.

The owner is the IMSID (or RSENAME for XRF systems) of the IMS system that owns the node. An IMS system owns a node resource if the resource is active (the node is logged on), or an IMS system is maintaining significant status for that resource.

PRESET

Displays the preset destination name for a node. A node is in preset destination mode set by the /SET command. The preset destination name is either a transaction name, or a logical terminal (LTERM) name. All messages entered from this node are sent to the preset destination transaction or LTERM.

QCNT

Displays the node message queue count. The queue count is the sum of the queue counts for each logical terminal associated with the specified node.

The local queue counts value returned on this command represents the messages being processed by the IMS system where this command is issued. In a shared-queues environment, do not use the local queue counts

for queue evaluation purposes. The global queue counts, which are returned on this command, represent the current message queue counts on the shared queues structure.

If the LOCAL parameter is also specified on the SHOW keyword, then all IMS systems that process the command, including the command master, display the local queue count. This is valid whether or not shared message queues are enabled.

If the GLOBAL parameter is also specified on the SHOW keyword, and shared message queues are enabled, then the command master displays the global queue count on the shared message queues (MSGQ). If the node resource exists in the resource structure, then the logical terminals associated with the global node resource are used to obtain the queue counts from shared queues. Otherwise, the logical terminals associated with the local node resource are used, but if the node does not exist locally on the command master, then the queue count is 0.

The local and global queue counts are displayed as separate output fields.

RECOVERY
Displays the status recovery mode (SRM) and level of recovery for the node. End-user significant status can be conversation, Fast Path, full-function response mode, or STSN (set-and-test-sequence-number) status.

SRM determines where end-user significant status, if any exists, is recovered following a session or IMS termination. The output displays the SRM for the node as either GLOBAL (sysplex terminal management recovers it in the resource structure), LOCAL (IMS recovers it locally, which indicates an affinity to a particular IMS), or NONE (status is discarded).

Level of recovery determines what end-user significant status, if any exists, is recovered (if SRM is GLOBAL or LOCAL). The output displays whether conversation status is recovered (RCVYCONV), Fast Path status is recovered (RCVYFP), full-function response mode is recovered (RCVYRESP), and STSN status is recovered (RCVYSTSN).

STATUS
Returns local or global status of the node. See "QUERY NODE status" on page 436 for a list and meaning of possible status that might be returned.

TYPE
VTAM node type. The possible node types are AVM, FIN, LUT6, NTO, SLUP, SLU1, SLU2, 3277, 3286, 3790, CONSOLE, RDR/PTR (for SYSOUT and SPOOL), and TCO.

USER
Displays the dynamic or ISC user associated with the node. For ISC parallel-session nodes, this is the user structure representing the half-session qualifier. For static ISC, this is defined with the SUBPOOL macro. For non-ISC dynamic nodes, this is the user structure allocated to the node during signon of a user ID to the node. The user name might or might not be the same as the user ID, depending on installation-specific user exits or descriptors used. Static non-ISC nodes (and single-session ISC nodes) do not have users (even if a user ID is signed on).

USERID

Displays the user ID (for security, usually RACF), signed on to the node. This is applicable to any device type, static or dynamic, and is distinct from the user name.

VERSION

Displays the version number assigned to RM resources, which is assigned by MVS, and maintained by RM, when the resources are created or updated in the resource structure.

Static single-session nodes (not parallel-session ISC) are represented in RM by two resources: the node resource, and the static-node-user resource. The VERSION parameter displays the version number for each of these resources. Static parallel-session ISC nodes and all dynamic nodes are represented in RM by one resource: the node resource. The VERSION parameter displays the version number of the node resource.

VERSION applies only when sysplex terminal management is enabled. VERSION is ignored when sysplex terminal management is not enabled.

STATUS()

Selects nodes for display that possess at least one of the specified node statuses. The status might exist locally or globally if sysplex terminal management (STM) is enabled.

The STATUS filter is valid in both a sysplex terminal management environment and in a non-sysplex terminal management environment. In a sysplex terminal management environment, the status selected might exist locally, globally, or both. If sysplex terminal management is not enabled, then the status only exists locally.

If SHOW(LOCAL) is specified, then IMS will select only those nodes with the appropriate status in the local system. The command is processed by all IMS systems, including the command master.

If SHOW(GLOBAL) is specified, and sysplex terminal management is enabled, then IMS will select only those nodes with the appropriate status in the resource structure. The command is processed only by the command master.

If SHOW(GLOBAL) is specified, but sysplex terminal management is not enabled, then the command is rejected.

If SHOW(GLOBAL,LOCAL) is specified (the default), then IMS will select those nodes with the appropriate status either locally or in the resource structure (if sysplex terminal management is enabled). The command is processed by all IMS systems. The command master processes both global and local information.

The output returned when the status filter is specified includes the status of the node, even if SHOW(STATUS) is not specified.

See "QUERY NODE status" on page 436 to determine which filters can be used to select nodes with corresponding status.

USER()

For a specific ISC node, specifies the names of one or more ISC user names allocated to the node. Valid names are 1-8 characters, and wildcards can be specified. To display all users allocated to the node, specify USER(*), or omit the USER keyword.

The output returned includes the user name even if SHOW(USER) is not specified.

Usage notes

QUERY NODE can be specified only through the OM API.

QUERY NODE can be issued on an XRF alternate system, but SHOW(GLOBAL) is not supported. Only local information can be displayed.

The processing of the QUERY NODE command is different depending on whether IMS sysplex terminal management is enabled.
- If IMS sysplex terminal management is not enabled, processing is local for each system. The results of type-1 and type-2 commands are similar.
- If IMS sysplex terminal management is enabled, type-1 and type-2 command processing is similar when displaying local information. However, they differ in how global information is displayed.
- For type-1 /DISPLAY commands with IMS sysplex terminal management enabled, the command master displays information from either the resource structure or the local system, but not both. If the resource being displayed is not owned by any system or is owned by the command master, the command master displays the global resource. However, if the resource is owned by a system other than the command master, the command master displays only the local resource, and the owning system is responsible for displaying the global resource.
- For type-2 QUERY commands with IMS sysplex terminal management enabled, the command master is the only system that displays global resource information, regardless of whether the resource is owned. In addition, the command master displays local resource information. All other IMS systems that process the command display local resource information only. This approach allows more flexibility in displaying all information in an IMSplex.

The SHOW keyword determines which IMS systems process the command and what information is displayed.
- If SHOW(GLOBAL) is specified, then the command master displays global information, which includes the global queue count if shared queues are enabled, generic resource affinity information if VTAM generic resource (VGR) is enabled, and status from the resource structure if sysplex terminal management is enabled (STM=YES defined in DFSDCxxx PROCLIB member). This is true whether or not the node is active on any particular IMS system. All other IMS systems to which OM routes the command ignore the GLOBAL parameter with return code X'00000004' and reason code X'00001000'.
- If SHOW(LOCAL) is specified, then each IMS system to which OM routes the command (including the command master) processes the command, and displays information that is local to each system.
- If both GLOBAL and LOCAL are specified (which is the default), then the command master displays both global and local information, and all other IMS systems to which OM routes the command displays local information.

Equivalent IMS type-1 commands

The following table shows variations of the QUERY NODE command and the type-1 IMS commands that perform similar functions.

Table 130. Type-1 equivalents for the QUERY NODE command

QUERY NODE command	Similar IMS type-1 command
QUERY NODE SHOW(AFFIN)	/DISPLAY AFFIN NODE node

Table 130. Type-1 equivalents for the QUERY NODE command (continued)

QUERY NODE command	Similar IMS type-1 command
QUERY NODE SHOW(CID)	/DISPLAY NODE node
QUERY NODE SHOW(CONV)	/DISPLAY CONV NODE node
QUERY NODE SHOW(COUNT)	/DISPLAY NODE node
QUERY NODE SHOW(EMHQ)	/DISPLAY NODE node QCNT EMHQ
QUERY NODE SHOW(LTERM)	/DISPLAY ASMT NODE node
QUERY NODE SHOW(MODETBL)	/DISPLAY NODE node MODE
QUERY NODE SHOW(OWNER)	/DISPLAY NODE node RECOVERY
QUERY NODE SHOW(PRESET)	/DISPLAY NODE node
QUERY NODE SHOW(QCNT)	/DISPLAY NODE node /DISPLAY NODE node QCNT
QUERY NODE SHOW(RECOVERY)	/DISPLAY NODE node RECOVERY
QUERY NODE SHOW(STATUS)	/DISPLAY NODE node
QUERY NODE SHOW(TYPE)	/DISPLAY NODE node
QUERY NODE SHOW(USERID)	/DISPLAY NODE node
QUERY NODE SHOW(USER)	/DISPLAY NODE node /DISPLAY ASMT NODE node
QUERY NODE STATUS(CONV)	/DISPLAY CONV
QUERY NODE STATUS(TRACE)	/DISPLAY TRACE NODE
QUERY NODE STATUS(status)	/DISPLAY STATUS NODE

Output fields

The following table shows the QUERY NODE output fields. The columns in the
table are:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the column heading for the output field in the formatted output.

SHOW parameter
> Identifies the parameter on the SHOW keyword that caused the field to be
> generated. Error appears for output fields that are returned for a non-zero
> completion code. N/A (not applicable) appears for output fields that are
> always returned.

Scope Identifies the scope of the output field. GBL indicates that the field can be
> generated only by the command master when displaying global
> information for SHOW(GLOBAL). LCL indicates that the field can be
> generated by any IMS displaying local information for SHOW(LOCAL).
> N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 131. Output fields for the QUERY NODE command

Short label	Long label	SHOW parameter	Scope	Meaning
AFFIN	Affin	AFFIN	GBL	IMS APPLID to which the node has VTAM generic resource affinity, if applicable.
AMTB	ActMdtbl	MODETBL	LCL	Active mode table. This name is only displayed when the session is active.
CC	CC	N/A	N/A	Completion code. The completion code indicates whether or not IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 439 for more information. The completion code is always returned.
CCTXT	CCText	*Error*	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
CICAP	CICSApplid	CICSAPPL	LCL	The APPLID of the remote CICS subsystem.
CICNT	CICSNetid	CICSNETID	LCL	The NETWORK ID of the remote CICS.
CID	CID	CID	LCL	VTAM connection identifier.
CNTR	RecdCnt	COUNT	LCL	Number of messages received from the specified node.
CNTS	SentCnt	COUNT	LCL	Number of messages sent to the specified node.
CONVID	ConvID	CONV	GBL	The conversation ID for a conversation associated with the node, as it exists in the resource structure. A node might have zero, one, or more conversations. Each conversation will have its own line of output.
CONVSTT	ConvStat	CONV	GBL	The status of a conversation associated with the node, as it exists in the resource structure. A node might have zero, one, or more conversations. The status can be: • CONVHELD - conversation is held • CONVACTV - conversation is active • CONVSCHD - conversation is scheduled
CONVTRN	ConvTran	CONV	GBL	The transaction for a conversation associated with the node, as it exists in the resource structure. A node might have zero, one, or more conversations.
DMTB	DefMdtbl	MODETBL	LCL	Default mode table.
EMHQ	EMHQCnt	EMHQ	GBL	Global logical terminal queue count in the EMH (Expedited Message Handler) queues. EMHQ is displayed only if shared EMH is used.
GBL	Gbl	GLOBAL	GBL	If 'Y', then the output reflects the status found globally in RM. If blank, then the output reflects the status found locally.
ID	ID	ID	GBL	For ISC parallel-session terminals, displays the global half-session qualifier of the other system.

Table 131. Output fields for the QUERY NODE command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
ISCUSER	ISCUser	N/A	N/A	When the node is parallel-session ISC, the user subpool name is returned to differentiate parallel sessions for the same node. This field will appear when an ISC node has been selected for display. When the user name is also displayed, both the ISCUSER and USER fields will contain the same user name. For LOCAL scope, N/A is displayed for a parallel session that is not yet allocated. For GLOBAL scope, N/A is displayed once for each node, which represents global status for the node.
LCONVID	LConvID	CONV	LCL	The conversation ID for a conversation associated with the node, as it exists in the local system. A node might have zero, one, or more conversations. Each conversation will have its own line of output.
LCONVSTT	LConvStat	CONV	LCL	The status of a conversation associated with the node, as it exists in the local system. A node might have zero, one, or more conversations. The status can be: • CONVHELD: Conversation is held • CONVACTV: Conversation is active • CONVSCHD: Conversation is scheduled
LCONVTRN	LConvTran	CONV	LCL	The transaction for a conversation associated with the node, as it exists in the local system. A node might have zero, one, or more conversations.
LGND	LclLogonD	LOGOND	LCL	The dynamic logon descriptor name that was used to establish the current session for the ISC TCP/IP terminal.
LIC	LclImsCon	IMSCON	LCL	For ISC TCP/IP nodes, the IMSplex member name of the local IMS Connect that provides TCP/IP support for the specified node, as defined by the LCLICON= keyword on the logon descriptor or, for statically defined terminals, on the ISCTCPIP keyword in the DFSDCxxx PROCLIB member.
LID	LID	ID	LCL	For ISC VTAM parallel-session terminals, displays the local half-session qualifier of the other system. For ISC TCP/IP parallel-session terminals, displays the dynamic or ISC user that is allocated to the node.
LINE	Line	N/A	LCL	Identifies the line number for system console, SPOOL, SYSOUT, or TCO device. This is only displayed if the node is non-VTAM.
LLTERM	LLterm	LTERM	LCL	Local logical terminal names. The logical terminal names associated with the node.

Table 131. Output fields for the QUERY NODE command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LPRST	LPreset	PRESET	LCL	Identifies the preset destination transaction or LTERM name when the node is in preset destination mode, which is established by the /SET command. All messages entered at this terminal are sent to the preset destination trancode or LTERM.
LQ	LQCnt	QCNT	LCL	Local queue count.
LRCVY	LRcvy	RECOVERY	LCL	The level of recovery for end-user significant status in the local system, which indicates what type of status is recoverable. Any value presented here implies that the corresponding status is recoverable. If SRM is LOCAL, the status will be recovered locally. If SRM is GLOBAL, the status will be recovered globally. These values are not applicable if SRM is NONE or there is no SRM. The status values that can be returned (more than one are possible) are: • CONV: IMS conversations are recoverable (RCVYCONV=YES). • FP: Fast Path status is recoverable (RCVYFP=YES). • RESP: Full-function response mode status is recoverable (RCVYRESP=YES). • STSN: STSN status is recoverable (RCVYSTSN=YES).
LSRM	LSRM	RECOVERY	LCL	The status recovery mode in the local system, which determines where the end-user significant status is maintained and recovered from. The output will be one of the following: • GBL: Status is saved globally in the IMS resource structure. • LCL: Status is saved in local control blocks and log records. • NONE: Status is not saved in the IMS resource structure or log records. • Blank: SRM is not yet established, the node is not logged on and there is no end-user significant status, or there is no user signed on.
LSTT	LclStat	STATUS	LCL	Local node status. See "QUERY NODE status" on page 436 for a list and explanation of the possible node status.
LTERM	Lterm	LTERM	GBL	Global logical terminal names. The logical terminal names associated with the node.
LTYPE	LType	TYPE	LCL	VTAM node type. The possible node types are: AVM, FIN, LUT6, NTO, SLUP, SLU1, SLU2, 3277, 3286, 3790, CONSOLE, RDR/PTR (for SYSOUT and SPOOL), and TCO.
LUID	LUserid	USERID	LCL	Identifies the local user ID signed on to the node.

Table 131. Output fields for the QUERY NODE command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LUSER	LUser	USER	LCL	Identifies the local dynamic or ISC user allocated to the node.
LVER	LVersion#	VERSION	LCL	Version number for the node resource being maintained in the local system. This field applies only when STM is enabled.
LVERSNU	LVersion#SNU	VERSION	LCL	Version number for the static-node-user resource being maintained in the local system. This field applies only to static single-session nodes when STM is enabled.
MBR	MbrName	N/A	N/A	IMSplex member (modular unit) that built the output line. IMS identifier of the IMS that built the output. The IMS identifier is always returned.
NODE	Node	N/A	N/A	The node name. The node name is always returned.
OWNER	Owner	OWNER	GBL	Resource owner. IMS identifier or RSENAME of IMS where the node is active. If no owning IMS system exists and RM contains an entry for the resource, the owner field will be blank.
QCNT	QCnt	QCNT	GBL	Global queue count on the shared queues. Global queue count can be displayed only if shared queues are used.
PTERM	PTerm	N/A	LCL	Identifies the PTERM number for system console, SPOOL, SYSOUT, or TCO device. This is only displayed if the node is non-VTAM.
RCVY	Rcvy	RECOVERY	GBL	The level of recovery for end-user significant status in the resource structure, which indicates what type of status is recoverable. Any value presented here implies that the corresponding status is recoverable. If SRM is LOCAL, the status will be recovered locally. If SRM is GLOBAL, the status will be recovered globally. These values are not applicable if SRM is NONE or there is no SRM. The status values that can be returned (more than one are possible) are: • CONV: IMS conversations are recoverable (RCVYCONV=YES). • STSN: STSN status is recoverable (RCVYSTSN=YES). • FP: Fast Path status is recoverable (RCVYFP=YES).

Table 131. Output fields for the QUERY NODE command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
SRM	SRM	RECOVERY	GBL	The status recovery mode in the resource structure, which determines where the end-user significant status is maintained and recovered from. The output will be one of the following: • GBL: Status is saved globally in the IMS resource structure. • LCL: Status is saved in local control blocks and log records. • NONE: Status is not saved in the IMS resource structure or log records. • Blank: SRM is not yet established, the node is not logged on and there is no end-user significant status, or there is no user signed on.
STT	Status	STATUS	GBL	Global node status. See "QUERY NODE status" for a list and explanation of the possible node status.
TYPE	Type	TYPE	GBL	VTAM node type. The possible node types are AVM, FIN, LUT6, NTO, SLUP, SLU1, SLU2, 3277, 3286, 3790.
UID	Userid	USERID	GBL	Identifies the RACF user ID signed on to the node.
USER	User	USER	GBL	Identifies the dynamic or ISC user signed on to or associated with the node.
VER	Version#	VERSION	GBL	Version number for the node resource being maintained in the resource structure. This field applies only when STM is enabled.
VERSNU	Version#SNU	VERSION	GBL	Version number for the static-node-user resource being maintained in the resource structure. This field applies only to static single-session nodes when STM is enabled.

QUERY NODE status

The following table shows the possible node status that can be displayed. The columns in the table are:

Status The node status that is displayed.

STATUS parameter
  The STATUS() filter that will select nodes with the specified status.

Scope The scope of the status. GBL indicates that the status can be global (it exists in the resource structure when STM is enabled), and is returned with the STT short label. LCL indicates that the status can be local, and is returned with the LSTT short label.

Meaning
  Provides a brief description of the status.

Table 132. QUERY NODE status

Status	STATUS parameter	Scope	Meaning
ACTIVE	ACTIVE	LCL	Node is in an XRF session on the active system.
ASR	ASR	LCL	Node has session initiation option ASR.
BACKUP	BACKUP	LCL	Node is in an XRF session on the alternate system.
CLSDST	CLSDST	LCL	Session is being disconnected.
COLD	COLD	GBL and LCL	For a SLUP or FINANCE terminal, indicates that the next session initiation is cold (message sequence numbers are initialized to 0).
CON	CON	LCL	Node is connected or in session with IMS.
CONVACT	CONV	GBL and LCL	An active conversation exists.
CONVHELD	CONV	GBL and LCL	One or more held conversations exist.
C1INOP C2INOP C3INOP C4INOP	CINOP	LCL	Indicates the inoperable node or terminal component, where C1, C2, C3 and C4 refer to the component as defined by system definition. (See the /COMPT command and the /RCOMPT command for details on how to ready inoperable components).
DEACT	DEACT	LCL	Node has been permanently deactivated. Restart of node requires /STOP DC and /START DC commands. Alternatively, node can be activated with UPDATE NODE STOP(DEACT). Message DFS2473 in the system console log might contain information regarding the reason this status was set. DFS2473 can occur more than once in the system console log.
EXCL	EXCL	GBL and LCL	The node is in exclusive mode set by the /EXCLUSIVE command. Exclusive mode restricts the output received by the terminal affected.
FORCSESS	FORCSESS	LCL	Node has session initiation option of FORCE.
IDLE	IDLE	LCL	No activity of any kind is in progress for the node.
INOP	INOP	LCL	NODE is inoperable.
LOCK	LOCK	LCL	Node is locked, set by the /LOCK command. The sending and receiving of messages for the VTAM node is stopped.

Table 132. QUERY NODE status (continued)

Status	STATUS parameter	Scope	Meaning
LOST	LOST	LCL	The VTAM LOSTERM EXIT has been scheduled for this node but has not yet been recognized by IMS. At the next interrupt for this node, IMS interrogates the LOSTERM value. All values, with one exception, result in an immediate CLSDST, or disconnection from IMS. For the LOSTERM exception, IMS must wait for VTAM to notify IMS (via another LOSTERM) of completion of recovery operation. This indicates an IMS logger's connection to the Transport Manager Subsystem is gone because of TMS or VTAM failure.
MFST	MFST	GBL and LCL	Node is in MFSTEST mode, set by UPDATE NODE START(MFST) command or /TEST MFS command. Terminals supported by Message Format Service use format blocks from a special test library if the requested format block is in the test library; otherwise the blocks are obtained from the production library.
OPNDST	OPNDST	LCL	OPNDST is in progress for this node.
PAGE	PAGE	LCL	Indicates an MFS paged message.
PRI	PRI	GBL and LCL	This node is the primary partner of an ISC session.
PRST	PRST	LCL	The node is in preset destination mode. PRST mode is established by the /SET command. All messages entered at this terminal are sent to the preset destination transaction code or logical terminal.
QUI	QUI	LCL	A VTAM node has sent a VTAM Quiesce-End-of-Chain indicator to suspend IMS output.
RELREQ	RELREQ	LCL	VTAM RELREQ exit routine has been driven but IMS is waiting for an operation in progress to complete before releasing the node.
RESP	RESP	GBL and LCL	The node is in response mode and the response reply message is available for output or in the process of being sent.
RESPINP	RESPINP	GBL and LCL	The node is in response mode and the response mode input is still in-doubt; for example, the response reply message is not available for output.
RESPINPFP	RESPINPFP	GBL and LCL	The node is in Fast Path response mode and the response mode input is still in-doubt; for example, the response reply message is not available for output.

Table 132. QUERY NODE status (continued)

Status	STATUS parameter	Scope	Meaning
RESYNC	RESYNC	LCL	The positive acknowledgment of an IMS recoverable output message was not received when the connection with the VTAM node was terminated. This message is subject to resynchronization when the next connection for this node is attempted.
RM	RM	GBL	The node exists in the resource structure managed by RM.
RMACTIVE	RMACTIVE	GBL	The node is active (logged-on) in the IMSplex, as indicated in the RM structure (RM active).
RMOWNED	RMOWNED	GBL	The node is owned by an IMS system in the IMSplex, as indicated in the RM structure (RM owned).
SEC	SEC	GBL and LCL	This node is the secondary partner of an ISC session.
SHUT	SHUT	LCL	Normal processing has completed for the node and a VTAM shutdown-complete indicator was returned to IMS. The node can receive IMS output but cannot enter data while in this state.
SIGN	SIGN	GBL and LCL	The user is signed on to a node under enhanced security.
SIMLOGON	SIMLOGON	LCL	A logon to IMS has been simulated.
STATIC	STATIC	GBL and LCL	The node was defined during system definition.
STOLGN	STOLGN	GBL and LCL	Node was stopped from logging on by the UPDATE NODE STOP(LOGON) command.
TEST	TEST	LCL	The node is in test mode.
TKOTRA	TKOTRA	LCL	Node in an XRF session is to be traced only during takeover, to help diagnose XRF terminal switch problems.
TRACE	TRACE	GBL and LCL	Node is being traced.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY NODE command. The OM return and reason codes that may be returned as a result of the QUERY NODE command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 133. Return and reason codes for the QUERY NODE command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.

Table 133. Return and reason codes for the QUERY NODE command (continued)

Return code	Reason code	Meaning
X'00000004'	X'00001000'	The command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000008'	X'00002014'	An invalid character was specified in the resource name.
X'00000008'	X'00002040'	An invalid parameter value was specified. An invalid SHOW or STATUS value might have been specified.
X'00000008'	X'00002133'	The NAME() keyword specified is either a generic name or multiple specific names, but the USER() keyword was also specified. When USER() is specified, only one NAME parameter is allowed.
X'0000000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See Table 134 on page 441 for details.
X'0000000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code. See Table 134 on page 441 for details.
X'00000010'	X'00004004'	Command processing terminated because CQS was not active.
X'00000010'	X'00004005'	Command processing terminated because CQS was not connected to the queue structure.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	Command processing terminated because the resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'00004028'	Global queue counts were requested, but shared message queues are not enabled.
X'00000010'	X'0000403C'	VTAM generic resource affinity information was requested, but VGR is not enabled.
X'00000010'	X'00004104'	Command processing terminated because RM is not available.
X'00000010'	X'00004108'	Command processing terminated because SCI is not available.
X'00000014'	X'00005004'	A DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	Command processing terminated because of an RM error.
X'00000014'	X'00005104'	Command processing terminated because of a CQS error.
X'00000014'	X'00005108'	Command processing terminated because of an SCI error.
X'00000014'	X'00005FFF'	Command processing terminated because of an internal IMS error.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 134. Completion codes for the QUERY NODE command

Completion code	Completion code text	Meaning
0		The QUERY NODE command completed successfully for the resource.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
98	CQS REQUEST ERROR	Global queue counts could not be obtained because of a CQS error.
1A1	Node resource is in error	The node resource was found in the resource structure and an associated resource was needed, but it was either not found or appeared to be in error. This is normally an error condition. However, it could be a temporary condition caused by terminal or command activity. The command should be retried.

Examples

The following are examples of the QUERY NODE command:

Example 1 for QUERY NODE command

TSO SPOC input:
```
QRY NODE NAME(NODE2*,XYZ) SHOW(LOCAL)
```

TSO SPOC output:
```
Node      ISCUser  MbrName CC CCText
NODE21             IMS1     0
NODE22             IMS1     0
NODE22             IMS2     0
NODE23             IMS2     0
NODE24    USER24A  IMS2     0
NODE24    USER24B  IMS2     0
NODE24    N/A      IMS2     0
XYZ                IMS1    10 NO RESOURCES FOUND
XYZ                IMS2    10 NO RESOURCES FOUND
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM and shared queues are irrelevant because only local information is requested. IMS1, the command master, displays only local information because no global information is requested. IMS2 displays local information only.
• NODE21 exists on IMS1 only.

- NODE22 exists on IMS1 and IMS2.
- NODE23 exists on IMS2 only.
- NODE24 is an ISC node with 3 parallel sessions available on IMS2, two of which are allocated.
- XYZ does not exist on any system.

Example 2 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE2*)
```

TSO SPOC output:

```
Node      ISCUser  MbrName CC  Gbl
NODE21             IMS1     0  Y
NODE21             IMS1     0
NODE22             IMS1     0
NODE22             IMS2     0
NODE23             IMS1     0  Y
NODE23             IMS2     0
NODE24    USER24A  IMS1     0  Y
NODE24    USER24B  IMS1     0  Y
NODE24    N/A      IMS1     0  Y
NODE24    USER24A  IMS2     0
NODE24    USER24B  IMS2     0
NODE24    N/A      IMS2     0
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information only.
- NODE21 exists on IMS1 and in the resource structure.
- NODE22 exists on IMS1 and IMS2 only.
- NODE23 exists on IMS2 and in the resource structure.
- NODE24 is an ISC node with 3 parallel sessions available on IMS2, two of which are allocated on IMS2 and in the resource structure. IMS1 also displays an output line for NODE24 with N/A as the ISC user, which represents any global status that the node might have (that is not tied to any particular parallel session).

Example 3 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE2*) STATUS(STATIC) SHOW(LOCAL)
```

TSO SPOC output:

```
Node      ISCUser  MbrName CC  LclStat
NODE21             IMS1     0  IDLE,CONVACT,CON,STATIC
NODE22             IMS1     0  IDLE,STATIC
NODE22             IMS2     0  IDLE,STATIC
NODE24    USER24A  IMS2     0  IDLE,CON,PRI,STATIC
NODE24    USER24B  IMS2     0  IDLE,CON,PRI,STATIC
NODE24    N/A      IMS2     0  IDLE,STATIC
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays local information only

because SHOW(LOCAL) is specified. IMS2 displays local information only. All static nodes are displayed, and status is displayed, because the STATUS filter was specified.
- NODE21 exists on IMS1, is logged on, and has an active conversation.
- NODE22 exists on IMS1 and IMS2.
- NODE24 is an ISC node with 3 parallel sessions available on IMS2, two of which are allocated and are the primary partners.

Example 4 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE21) SHOW(GLOBAL,CONV,LTERM,STATUS)
```

TSO SPOC output:

```
(screen 1)
Node    MbrName CC  Gbl  Lterm     ConvID ConvTran ConvStat
NODE21  IMS1     0  Y
NODE21  IMS1     0  Y    LTERM21A
NODE21  IMS1     0  Y    LTERM21B
NODE21  IMS1     0  Y                  1 TRAN1A   CONVHELD
NODE21  IMS1     0  Y                  2 TRAN1A   CONVHELD
NODE21  IMS1     0  Y                  3 TRAN1A   CONVACTV
```

```
(scrolled right to screen 2)
Node    MbrName Gbl  Status
NODE21  IMS1     Y   CONVACT,STATIC,RM,RMACTIVE,RMOWNED
NODE21  IMS1     Y
NODE21  IMS1     Y
NODE21  IMS1     Y
NODE21  IMS1     Y
NODE21  IMS1     Y
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global information only. IMS2 ignores the command (RC=4, RSN=x1000) because only global information is requested.

NODE21 exists in the resource structure. IMS1 displays a global line which shows that the node is active in the IMSplex, and has a conversation active. There are two logical terminals assigned to the node, and are displayed on separate output lines. There are three conversations associated with the node, and are displayed on separate output lines.

Example 5 for QUERY NODE command

TSO SPOC input:

```
QRY NODE NAME(NODE23) SHOW(CONV,STATUS,OWNER,RECOVERY)
```

TSO SPOC output:

```
(screen 1)
Node    MbrName CC CCText             Gbl  Owner  SRM  Rcvy
NODE23  IMS1     0                    Y    IMS2   LCL  CONV,FP
NODE23  IMS1    10 NO RESOURCES FOUND
NODE23  IMS2     0
NODE23  IMS2     0
NODE23  IMS2     0
```

```
(scrolled right to screen 2)
```

```
                         Node      MbrName Gbl  ConvID ConvTran ConvStat Status
                         NODE23    IMS1    Y                              RM,RMACTIVE,RMOWNED
                         NODE23    IMS1
                         NODE23    IMS2
                         NODE23    IMS2
                         NODE23    IMS2

                         (scrolled right to screen 3)
                         Node      MbrName Gbl  LSRM LRcvy     LConvID LConvTran LConvStat
                         NODE23    IMS1    Y
                         NODE23    IMS1
                         NODE23    IMS2         LCL  CONV,FP
                         NODE23    IMS2                        1 TRAN1A    CONVHELD
                         NODE23    IMS2                        2 TRAN1B    CONVACTV

                         (scrolled right to screen 4)
                         Node      MbrName Gbl  LclStat
                         NODE23    IMS1    Y
                         NODE23    IMS1
                         NODE23    IMS2         IDLE,CONVACT,CON
                         NODE23    IMS2
                         NODE23    IMS2
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information.

NODE23 exists in IMS2 and in the resource structure. IMS1 displays a global line which shows the node is active and owned on IMS2, and its status recovery mode is LOCAL, which means conversation information is not known globally. IMS1 also displays a local line showing that NODE23 does not exist locally. IMS2 displays the local information, which includes one status line, and an additional output line for each conversation active or held locally.

Example 6 for QUERY NODE command

TSO SPOC input:
```
QRY NODE NAME(NODE23) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
Node      MbrName CC CCText              Gbl QCnt EMHQCnt Type  Owner  SRM  Rcvy
NODE23    IMS1     0                     Y     0       0 SLU2  IMS2   LCL  CONV,FP
NODE23    IMS1    10 NO RESOURCES FOUND
NODE23    IMS2     0

(scrolled right to screen 2)
Node      MbrName Gbl User    Userid  Affin    Version# Version#SNU Status
NODE23    IMS1    Y   USER23  UID23   IMS2            5           0 RM,RMACTIVE,RMOWNED
NODE23    IMS1
NODE23    IMS2

(scrolled right to screen 3)
Node      MbrName Gbl LQCnt LType CID      RecdCnt SentCnt DefMdtbl ActMdtbl
NODE23    IMS1    Y
NODE23    IMS1
NODE23    IMS2            0 SLU2  02000003       9      13 SLU2MOD2 SLU2MOD2

(scrolled right to screen 4)
Node      MbrName Gbl LSRM Lrcvy    LUser   LUserid LVersion# LVersion#SNU LclStat
```

```
NODE23    IMS1    Y
NODE23    IMS1
NODE23    IMS2          LCL  CONV,FP  USER23   UID23          5           0 IDLE,CONVACT,CON
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are active. IMS1, the command master, displays global and local information. IMS2 displays local information.

NODE23 exists in IMS2 and in the resource structure. IMS1 displays a global line which shows global queue counts and global status from the resource structure. Global status indicates that the node is active on IMS2, has a VGR affinity with IMS2, and its status recovery mode is LOCAL, which means conversation information is not known globally. IMS1 also displays a local line showing that NODE23 does not exist locally. IMS2 displays the local information, which shows that conversation status exists locally.

Example 7 for QUERY NODE command

TSO SPOC input:
```
QRY NODE NAME(DFSLN001,DFSLN002) SHOW(LOCAL,TYPE)
```

TSO SPOC output:
```
Node      Line Pterm MbrName CC LType
DFSLN001   1     1   IMS1     0 CONSOLE
DFSLN001   1     1   IMS2     0 CONSOLE
DFSLN002   2     1   IMS1     0 RDR/PTR
DFSLN002   2     1   IMS2     0 RDR/PTR
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM and shared queues are irrelevant because only local information is requested. IMS1, the command master, displays only local information because no global information is requested. IMS2 displays local information only.

DFSLN001 represents the system console, LINE 1 PTERM 1, on each system. DFSLN002 represents a SYSOUT or SPOOL device, LINE 2 PTERM 1, on each system.

Example 8 for QUERY NODE command: ISC TCP/IP parallel session

Explanation: In the following example, TCP is shown in the LType column to indicate that node CICS is an ISC TCP/IP terminal. In the LclLogonD column, the ETO logon descriptor that was used to establish the current session is LU6TCP1. The local IMS Connect instance that is providing TCP/IP support is identified as HWS1 in the LclImsCon column. Under the CICSApplid column, CICS1 is the APPLID of the remote CICS subsystem with which the ISC TCP/IP terminal is connected. The network ID, IMSNET, of the CICS subsystem that the ISC TCP/IP terminal is connected to is displayed under the CICSNetid column.

TSO SPOC input:
```
QUERY NODE NAME(CICS) SHOW(ALL)
```

TSO SPOC output:
```
(Screen 1)
Node       ISCUser  MbrName    CC     LQCnt LType  LclImsCon   CICSApplid
CICS       IMSUS01  IMS1        0         0 TCP    HWS1        CICS1
(Screen 2)
Node       ISCUser  MbrName    CICSNetid  LclLogonD  RecdCnt  SentCnt  LSRM
```

```
CICS      IMSUS01 IMS1      IMSNET     LU6TCP1      0         0  LCL
(Screen 3)
Node      ISCUser MbrName   LRcvy      LUser   LID        LclStat
CICS      IMSUS01 IMS1      STSN       IMSUS01 IMSUS01    SIGN,IDLE,CON
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

/DISPLAY ASMT command (Commands)

/DISPLAY CONV command (Commands)

/DISPLAY NODE command (Commands)

/DISPLAY STATUS command (Commands)

/DISPLAY TRACE command (Commands)

Command keywords and their synonyms (Commands)

QUERY ODBM commands

Use the QUERY ODBM commands to query information about Open Database Manager (ODBM), a component of the Common Service Layer (CSL).

Subsections:
- "QUERY ODBM TYPE(ALIAS) command"
- "QUERY ODBM TYPE(CONFIG) command" on page 449
- "QUERY ODBM TYPE(DATASTORE) command" on page 454
- "QUERY ODBM TYPE(SCIMEMBER) command" on page 458
- "QUERY ODBM TYPE(THREAD) command" on page 461
- "QUERY ODBM TYPE(TRACE) command" on page 469

QUERY ODBM TYPE(ALIAS) command

Use the QUERY ODBM TYPE(ALIAS) command to query information about Open Database Manager (ODBM) configuration.

Subsections:
- "Environment"
- "Syntax" on page 447
- "Keywords" on page 447
- "Usage notes" on page 447
- "Output fields" on page 447
- "Return, reason, and completion codes" on page 448
- "Examples" on page 449

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax

```
►►─┬─QUERY─┬──ODBM──TYPE(ALIAS)──┬──────────NAME(*)──────────────────┬──►
   └─QRY───┘                     │              ┌─,───────────┐       │
                                 └─NAME(─▼──alias_name──┴─)─┘

►─┬──────────────┬──────────────────────────────────────────────────►◄
  └─SHOW(ALL)──┘
```
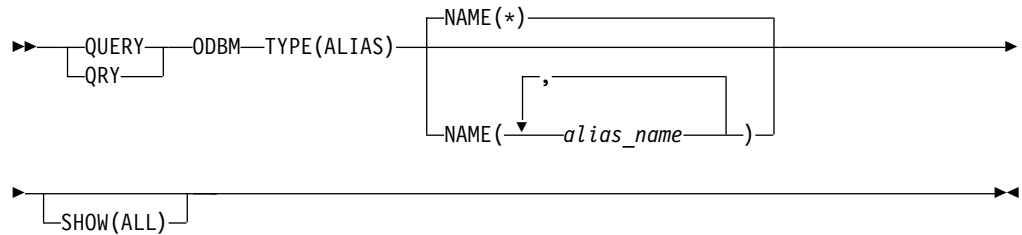
Keywords

The following keywords are valid for the QUERY ODBM TYPE(ALIAS) command.

NAME()
> Specifies the alias name of the IMS data store to be queried, as defined in the ALIAS substatement of the ODBM configuration statement in the CSLDCxxx configuration member. Wildcards (* and %) can be specified for *alias_name*. The *alias_name* is a repeatable parameter. The default is NAME(*), which returns information about all aliases that are known to ODBM.

SHOW()
> Specifies the ODBM TYPE(ALIAS) output fields to be returned. The parameters that are supported with the SHOW keyword are:

> **ALL**
> > Returns all information about ODBM data store alias and its status.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 135. Output field descriptions for the QUERY ODBM TYPE(ALIAS) command

Short label	Long label	Keyword	Meaning
ALCSTT	AliasStatus	N/A	Alias connection status. Status can be either STARTED or STOPPED.

Table 135. Output field descriptions for the QUERY ODBM TYPE(ALIAS) command (continued)

Short label	Long label	Keyword	Meaning
ALIAS	AliasName	N/A	Alias name.
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
DSCSTT	DatastoreStatus	N/A	Connection status of the data store associated with the alias.
DSTR	DatastoreName	N/A	Data store name for the alias.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
THDCT	ThreadCount	N/A	Count of threads that are associated with the alias.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(ALIAS) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 136. Return and reason codes for the QUERY ODBM TYPE(ALIAS) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(ALIAS) command completed successfully.
X'40000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'40000008'	X'00002008'	Insufficient number of keywords.
X'40000008'	X'00002014'	Invalid character in resource name.
X'40000008'	X'0000203C'	Invalid parameter specified.
X'4000000C'	X'00003000'	At least one request was successful.
X'4000000C'	X'00003004'	No requests were successful.
X'40000014'	X'00005034'	An OM response buffer request failed.
X'40000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 137. Completion codes for the QUERY ODBM TYPE(ALIAS) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(ALIAS) command completed successfully.

Completion code	Completion code text	Meaning
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
2	PARAMETER SPECIFIED TOO LONG	The NAME() parameter specified is longer than the valid alias name length of 4 characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(ALIAS) command

TSO SPOC input:

```
QRY ODBM TYPE(ALIAS) NAME(*) SHOW(ALL)
```

TSO SPOC output:

```
Mbrname     AliasName   CC   AliasStatus   ThreadCount   DatastoreName   DatastoreStatus
ODBM010D    I01A        0    STARTED       5             IMS1            STARTED
ODBM010D    I01B        0    STOPPED       0             IMS1            STARTED
ODBM010D    I02A        0    STARTED       10            IMS2            STARTED
ODBM010D    I02B        0    STARTED       0             IMS2            STARTED
ODBM020D    I03A        0    STOPPED       0             IMS3            STOPPED
ODBM020D    I03B        0    STARTED       0             IMS3            STOPPED
```

Explanation: The QUERY command displays alias information for all aliases that are known to ODBM.

QUERY ODBM TYPE(CONFIG) command

Use the QUERY ODBM TYPE(CONFIG) command to query information about Open Database Manager (ODBM) configuration.

Subsections:
- "Environment"
- "Syntax" on page 450
- "Keywords" on page 450
- "Usage notes" on page 451
- "Output fields" on page 451
- "Return, reason, and completion codes" on page 452
- "Examples" on page 453

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax
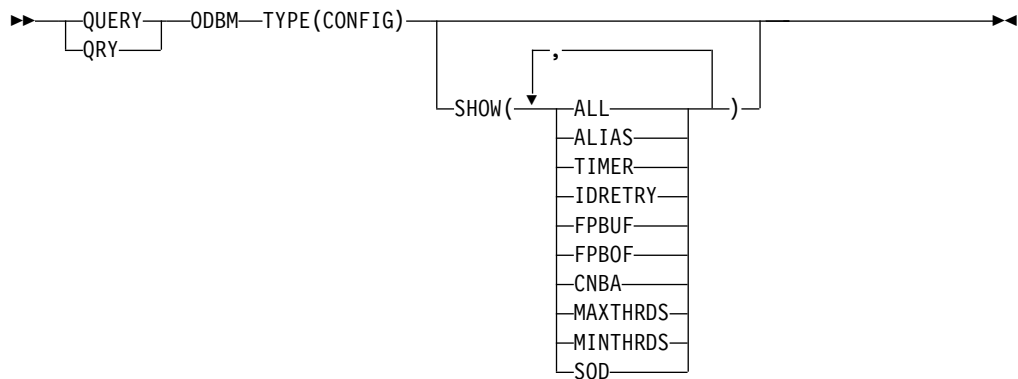
Keywords

The following keywords are valid for the QUERY ODBM TYPE(CONFIG) command.

SHOW()
Specifies the ODBM TYPE(CONFIG) output fields to be returned. The parameters supported with the SHOW keyword are:

ALL
Returns all information about the CSLDCxxx configuration member in PROCLIB.

ALIAS
Returns alias names for the data store, which are used by Open Database applications to access IMS data stores.

TIMER
Returns the amount of time (in seconds) between attempts by ODBM to establish a connection to an IMS data store. The value can be a number between 1–99.

IDRETRY
Returns the number of times ODBM will attempt to connect (identify) to an IMS data store after the initial attempt is unsuccessful. The value can be a number between 0–255.

FPBUF
Returns the number of Fast Path DEDB buffers to be allocated and fixed per thread.

FPBOF
Returns the number of Fast Path DEDB overflow buffers to be allocated per thread.

CNBA
Returns the total number of Fast Path NBA buffers for ODBM use.

MAXTHRDS
Returns the maximum number of concurrent active threads to an individual IMS data store.

MINTHRDS
Returns the minimum number of concurrent active threads to an individual IMS data store.

SOD
> Returns the SYSOUT class of the SNAP dumps produced by the ODBM address space. The value returned is a single alphanumeric character.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 138. Output field descriptions for the QUERY ODBM TYPE(CONFIG) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
CNBA	CNBA	CNBA	CNBA value for a data store connection.
CNFG	ConfigName	N/A	CSLDCxxx PROCLIB member name.
DSTR	DatastoreName	N/A	Data store name.
FPBOF	FPBOF	FPBOF	FPBOF value for a data store connection.
FPBUF	FPBUF	FPBUF	FPBUF value for a data store connection.
GBL	Global	N/A	Y or N value will be presented.
			Y indicates information is from the Global section in the CSLDCxxx configuration member.
			N indicates information is from the Local section in the CSLDCxxx configuration member.
IRTRY	IdRetry	IDRETRY	IDRETRY value of a data store connection.
MALIAS	Aliases	ALIAS	The alias names associated with an IMS data store connection.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
MXTHD	MaxThrds	MAXTHRDS	MAXTHRDS parameter value.
MNTHD	MinThrds	MINTHRDS	MINTHRDS parameter value.

Table 138. Output field descriptions for the QUERY ODBM TYPE(CONFIG) command (continued)

Short label	Long label	Keyword	Meaning
SOD	SOD	SOD	SYSOUT class of ODBM snap dumps.
TIMER	Timer	TIMER	TIMER parameter value for a data store connection.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(CONFIG) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 139. Return and reason codes for the QUERY ODBM TYPE(CONFIG) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(CONFIG) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 140. Completion codes for the QUERY ODBM TYPE(CONFIG) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(CONFIG) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
10	NO RESOURCES	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(CONFIG) command

TSO SPOC input:

```
QRY ODBM TYPE(CONFIG) SHOW(ALL)
```

TSO SPOC output:

MbrName	ConfigName	Global	DatastoreName	CC	FPBUF	FPBOF	CNBA	MaxThrds	MinThrds
ODBM1OD	CSLDC001	Y		0	10	5	100	1	1
ODBM1OD	CSLDC001	N	IMS1	0	0	0	0	10	1
ODBM2OD	CSLDC002	Y		0	50	5	550	10	5
ODBM2OD	CSLDC002	N	IMS1	0	0	0	0	50	10

```
(continued)
IdRetry Timer Aliases                SOD
                                      I
              IMS1,IMSA,IM1A,SYS1     E
      5    30                         I
              RRSN                    E
```

Explanation: The QUERY command will display all SHOW parameter values for the CSLDCxxx configuration members.

Example 2 for QUERY ODBM TYPE(CONFIG) command

TSO SPOC input:

```
QRY ODBM TYPE(CONFIG) SHOW(ALIAS,CNBA)
```

TSO SPOC output:

MbrName	ConfigName	Global	DatastoreName	CC	CNBA	Aliases
ODBM010D	CSLDC000	Y		0	0	
ODBM010D	CSLDC000	N	IMS1	0	0	I01A,I01B
ODBM010D	CSLDC000	N	IMS2	0	500	I02A,I02B
ODBM020D	CSLDC000	Y		0	100	
ODBM020D	CSLDC000	N	IMS3	0	200	I03A,I03B

Explanation: The QUERY command will display information about alias names for the data store and the total number of Fast Path NBA buffers for ODBM use.

Example 3 for QUERY ODBM TYPE(CONFIG) command

TSO SPOC input:

```
QRY ODBM TYPE(CONFIG)
```

TSO SPOC output:

MbrName	ConfigName	Global	DatastoreName	CC
ODBM010D	CSLDC000	Y		0
ODBM010D	CSLDC000	N	IMS1	0
ODBM010D	CSLDC000	N	IMS2	0
ODBM010D	CSLDC000	N	IMS1	0
ODBM010D	CSLDC000	N	IMS1	0
ODBM020D	CSLDC000	Y		0
ODBM020D	CSLDC000	N	IMS3	0

Explanation: The QUERY command will display information about Open Database Manager (ODBM) configuration.

QUERY ODBM TYPE(DATASTORE) command

Use the QUERY ODBM TYPE(DATASTORE) command to query information about an IMS data store and its associated resources.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 455
- "Output fields" on page 455
- "Return, reason, and completion codes" on page 456
- "Examples" on page 457

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax
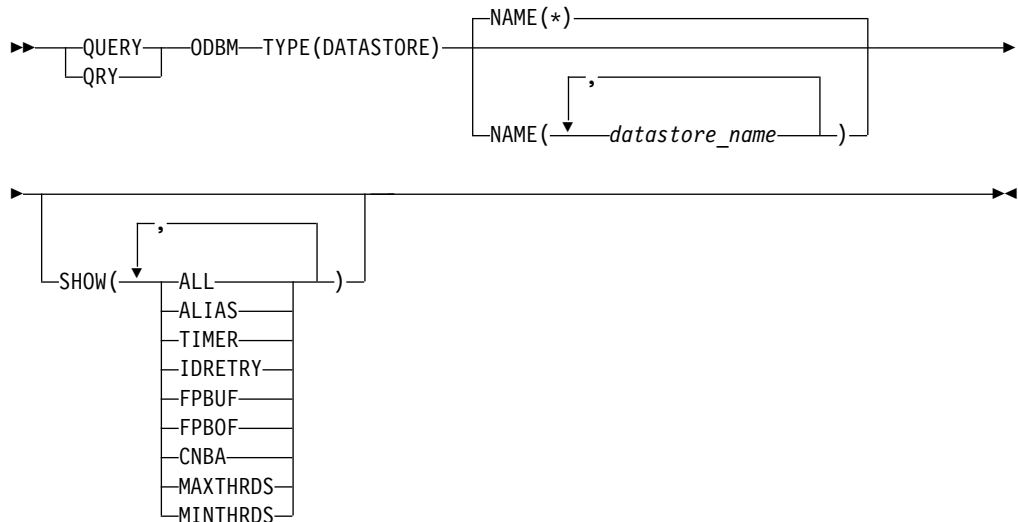
Keywords

The following keywords are valid for the QUERY ODBM TYPE(DATASTORE) command.

NAME()

Specifies the name of the data store to be displayed. Wildcards (* and %) can be specified for *datastore_name*. The *datastore_name* is a repeatable parameter. The default is NAME(*), which returns information about all data stores that are known to ODBM.

SHOW()
>   Specifies the ODBM TYPE(DATASTORE) output fields to be returned. One or
>   more parameters can be specified. The parameters supported with the SHOW
>   keyword are:

>   **ALL**
>   >   Returns all information about the specified IMS data store.

>   **ALIAS**
>   >   Returns alias names for the IMS data store. Client application programs
>   >   identify the IMS data store that they are accessing by the alias name.

>   **TIMER**
>   >   Returns the amount of time (in seconds) between attempts by ODBM to
>   >   establish a connection to an IMS data store. The value can be a number
>   >   between 1–99.

>   **IDRETRY**
>   >   Returns the number of times ODBM will attempt to connect (identify) to
>   >   an IMS data store after the initial attempt is unsuccessful. The value can be
>   >   a number between 0–255.

>   **FPBUF**
>   >   Returns the number of Fast Path DEDB buffers to be allocated and fixed
>   >   per thread.

>   **FPBOF**
>   >   Returns the number of Fast Path DEDB overflow buffers to be allocated
>   >   per thread.

>   **CNBA**
>   >   Returns the total number of Fast Path NBA buffers for ODBM use.

>   **MAXTHRDS**
>   >   Returns the maximum number of concurrent active threads to an
>   >   individual IMS data store.

>   **MINTHRDS**
>   >   Returns the minimum number of concurrent active threads to an
>   >   individual IMS data store.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation
programs that communicate with OM.

Output fields

Short label
>   Contains the short label that is generated in the XML output.

Long label
>   Contains the long label generated in the XML output.

Keyword
>   Identifies keyword on the command that caused the field to be generated.
>   *error* appears for output fields that can appear for a nonzero completion
>   code. N/A (not applicable) appears for output fields that are always
>   returned.

Meaning
> Provides a brief description of the output field.

Table 141. Output field descriptions for the QUERY ODBM TYPE(DATASTORE) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
CNBA	CNBA	CNBA	CNBA value in use for the data store connection.
DSCSTT	ConnectionStatus	N/A	Data store connection status.
DSTR	DatastoreName	N/A	Data store name.
FPBOF	FPBOF	FPBOF	FPBOF value for the data store connection.
FPBUF	FPBUF	FPBUF	FPBUF value for the data store connection.
IRTRY	IdRetry	IDRETRY	IDRETRY value of the data store connection.
MALIAS	Aliases	ALIAS	The alias names associated with the IMS data store connection.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
MXTHD	MaxThrds	MAXTHRDS	MAXTHRDS parameter value.
MNTHD	MinThrds	MINTHRDS	MINTHRDS parameter value.
THDCT	ThreadCount	N/A	Count of threads associated to the data store connection.
TIMER	Timer	TIMER	TIMER parameter value for the data store connection.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(DATASTORE) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 142. Return and reason codes for the QUERY ODBM TYPE(DATASTORE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(DATASTORE) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.

Table 142. Return and reason codes for the QUERY ODBM TYPE(DATASTORE) command (continued)

Return code	Reason code	Meaning
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 143. Completion codes for the QUERY ODBM TYPE(DATASTORE) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(DATASTORE) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
2	PARAMETER SPECIFIED TOO LONG	The NAME() parameter specified is longer than the valid data store name length of 8 characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(DATASTORE) command

TSO SPOC input:
```
QRY ODBM TYPE(DATASTORE) NAME(IMS*) SHOW(ALL)
```

TSO SPOC output:
```
MbrName    DatastoreName  CC  ConnectionStatus  ThreadCount  FPBUF  FPBOF  CNBA
ODBM010D   IMS1           0       STARTED                 5      0      0      0
ODBM010D   IMS2           0       STOPPED                 0     50     50    500
ODBM020D   IMS3           0       STARTED                10     10     10    200

(continued)
MinThrds MaxThrds IDRETRY TIMER  Aliases
       5       10       5    30  I01A,I01B
       2        5       5    30  I02A,I02B
       5       10       5    30  I03A,I03B
```

Explanation: The QUERY command will display all information about the data stores whose name begin with IMS.

Example 2 for QUERY ODBM TYPE(DATASTORE) command

TSO SPOC input:
```
QRY ODBM TYPE(DATASTORE) NAME(IMS*) SHOW(ALIAS,MAXTHRDS)
```

TSO SPOC output:

```
MbrName     DatastoreName  CC  ConnectionStatus  ThreadCount  MaxThrds  Aliases
ODBM010D    IMS1            0  STARTED                     5        10  IO1A,IO1B
ODBM010D    IMS2            0  STOPPED                     0         5  IO2A,IO2B
ODBM020D    IMS3            0  STARTED                    10        10  IO3A,IO3B
```

Explanation: The QUERY command will display information about alias names for the data store and the maximum number of concurrent active threads to an individual IMS data store.

Example 3 for QUERY ODBM TYPE(DATASTORE) command

TSO SPOC input:

```
QRY ODBM TYPE(DATASTORE) NAME(IMS*)
```

TSO SPOC output:

```
MbrName     DatastoreName  CC  ConnectionStatus  ThreadCount  Aliases
ODBM010D    IMS1            0        STARTED              5  IO1A,IO1B
ODBM010D    IMS2            0        STOPPED              0  IO2A,IO2B
ODBM020D    IMS3            0        STARTED             10  IO3A,IO3B
```

Explanation: The QUERY command will display information about data store connection status, thread count, and alias names for the data store.

QUERY ODBM TYPE(SCIMEMBER) command

Use the QUERY ODBM TYPE(SCIMEMBER) command to query information about an Open Database Manager (ODBM) client.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 459
- "Usage notes" on page 459
- "Output fields" on page 459
- "Return, reason, and completion codes" on page 460
- "Examples" on page 461

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax

```
►──────┬──────────────┬───────────────────────────────────────────►◄
       └─SHOW(ALL)──┘
```
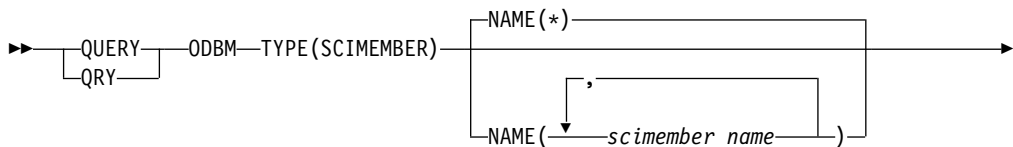
Keywords

The following keywords are valid for the QUERY ODBM TYPE(SCIMEMBER) command.

NAME()
> Specifies the name of the ODBM client to be displayed. Wildcards (* and %) can be specified for *scimember_name*. The *scimember_name* is a repeatable parameter. The default is NAME(*), which returns information about all SCI member names that are known to ODBM.

SHOW()
> Specifies the ODBM TYPE(SCIMEMBER) output fields to be returned. One or more parameters can be specified. The parameters supported with the SHOW keyword are:

> **ALL**
> > Returns all information about the ODBM clients that have the specified SCI member names.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 144. Output field descriptions for the QUERY ODBM TYPE(SCIMEMBER) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
SCMBR	SCIMbrName	N/A	The SCI member name of the client that is connected to ODBM.

Table 144. Output field descriptions for the QUERY ODBM TYPE(SCIMEMBER) command (continued)

Short label	Long label	Keyword	Meaning
THDCT	ThreadCount	N/A	Count of threads that are associated with the SCI member.
TYPE	Type	N/A	SCI member type of the client that is connected to ODBM.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(SCIMEMBER) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 145. Return and reason codes for the QUERY ODBM TYPE(SCIMEMBER) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(SCIMEMBER) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 146. Completion codes for the QUERY ODBM TYPE(SCIMEMBER) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(SCIMEMBER) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
2	PARAMETER SPECIFIED TOO LONG	The NAME() parameter specified is longer than the valid ODBM client name length of 8 characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(SCIMEMBER) command

TSO SPOC input:
```
QRY ODBM TYPE(SCIMEMBER) NAME(HWS*) SHOW(ALL)
```

TSO SPOC output:
```
MbrName     SCIMbrName  CC   Type      ThreadCount
ODBM01OD       HWS1      0   IMSCON        21
ODBM01OD       HWS2      0   IMSCON         5
```

Explanation: The QUERY command will display ODBM client information about SCI members whose names begin with HWS.

QUERY ODBM TYPE(THREAD) command

Use the QUERY ODBM TYPE(THREAD) command to query information about an Open Database Manager (ODBM) thread and its associated resources.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 462
- "Usage notes" on page 464
- "Output fields" on page 464
- "Return, reason, and completion codes" on page 465
- "Examples" on page 466

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax

```
    ┌─PSB(*)──────────────┐      ┌─SCIMEMBER(*)──────────────────┐
▶───┤                     ├──────┤                               ├──────────────▶
    │        ┌─,────────┐ │      │           ┌─,──────────────┐  │
    └─PSB(───┴─psb_name─┴─)┘     └─SCIMEMBER(─┴─scimember_name─┴─)┘


    ┌─STATUS(ALL)──────┐
▶───┤                  ├─────────────────────────────────────────────────────▶◀
    └─STATUS(INDOUBT)──┘    ┌─,──────────────┐
                 └─SHOW(──▼─┬─ALL─────────┬──)─┘
                           ├─ACTIVETIME───┤
                           ├─APPLCLIENTID─┤
                           ├─IDLETIME─────┤
                           ├─PSB──────────┤
                           ├─STARTTIME────┤
                           ├─TIME─────────┤
                           ├─URID─────────┤
                           ├─USERID───────┤
                           └─SCIMEMBER────┘
```
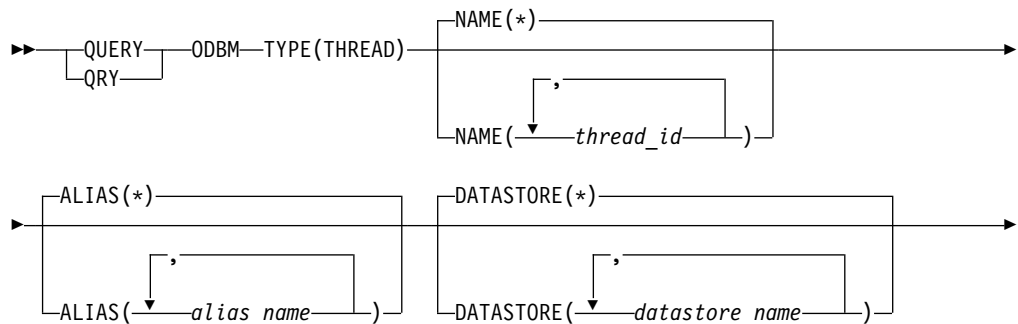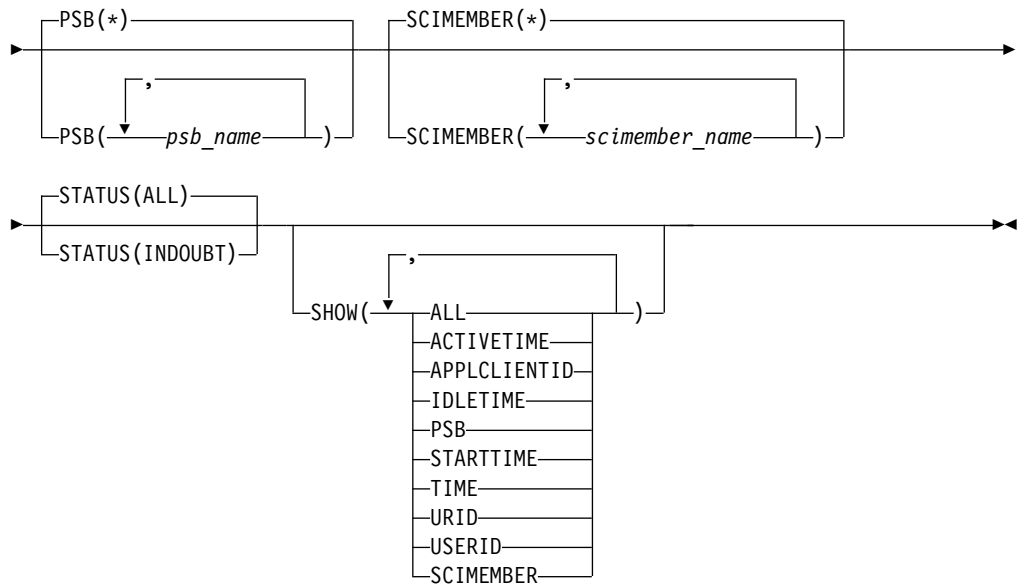
Keywords

The following keywords are valid for the QUERY ODBM TYPE(THREAD) command.

ALIAS(*alias_name*)
Specifies the 1–4 character alias name of the data store to be used as a filter. Thread information is returned if the *thread_id* is associated with the specified *alias_name*.

Wildcards (* and %) can be specified for the *alias_name*. The *alias_name* is a repeatable parameter. The default is ALIAS(*), which returns thread information for the specified *thread_id* regardless of alias affiliation.

DATASTORE(*datastore_name*)
Specifies the 1–8 character name of the data store to be used as a filter. Thread information is returned if the specified *thread_id* is associated with the *datastore_name* specified.

Wildcards can be specified for the *datastore_name*. The *datastore_name* is a repeatable parameter. The default is DATASTORE(*), which returns thread information for the specified *thread_id* regardless of data store affiliation.

NAME(*thread_id*)
Specifies the thread ID of the ODBM thread you want to be displayed. Wildcards (* and %) can be specified for the *thread_id*. The *thread_id* is a repeatable parameter. The default is NAME(*), which returns information about all threads that are known to ODBM. The length of each thread ID can be up to 32 characters.

PSB(*psb_name*)
Specifies the 1–8 character name of the PSB to be used as a filter. Thread information is returned if the specified *thread_id* is associated with the *psb_name* specified.

Wildcards (* and %) can be specified for the *psb_name*. The *psb_name* is a repeatable parameter. The default is PSB(*), which returns thread information for the specified *thread_id* regardless of PSB affiliation.

Note: If a PSB name is specified for the QRY ODBM TYPE(THREAD) command, PSB names will be included in the command output even if SHOW(PSB) is not specified.

SCIMEMBER(*scimember_name***)**

Specifies the 1–8 character name of the client connected to ODBM to be used as a filter. Thread information is returned if the *thread_id* is associated with the *scimember_name* specified.

Wildcards (* and %) can be specified for the *scimember_name*. The *scimember_name* is a repeatable parameter. The default is SCIMEMBER(*), which returns thread information for the *thread_id* regardless of ODBM client affiliation.

Note: If the SCIMEMBER keyword is specified for the QRY ODBM TYPE(THREAD) command, ODBM client names will be returned in the command output even if SHOW(SCIMEMBER) is not specified.

STATUS()

Allows for filtering on the QUERY ODBM TYPE(THREAD) command by the status of the thread. The following parameters can be specified:

ALL

Displays information about threads of any of the following status. STATUS(ALL) is the default.
- ACTIVE - The thread is active and is not in any of the following states.
- INPREPARE
- INCOMMIT
- INBACKOUT
- INDOUBT
- INTERM
- INABTERM

INDOUBT

Displays only threads with an INDOUBT status.

SHOW()

Specifies the ODBM TYPE(THREAD) output fields to be returned. One or more parameters can be specified. The parameters supported with the SHOW keyword are:

ALL

Returns all information about the ODBM thread.

ACTIVETIME

ACTIVETIME is defined as the time spent in a data store. Specifying SHOW(ACTIVETIME) returns the elapsed time from the time the data store is entered to the time that the QUERY command is issued. This value is not cumulative; if processing is not in a data store when the command is issued, a zero value will be returned in the ACTIVETIME column.

The format of ACTIVETIME is:

`HH:MM:SS.th`

If ACTIVETIME exceeds 24 hours, the ACTIVETIME column will display:

`24+ HOURS`

APPLCLIENTID

Returns the ID of the application client that originated the IMS data store connection request.

IDLETIME
    IDLETIME is defined as the time not in a data store. Specifying
    SHOW(IDLETIME) returns the elapsed time between returning from data
    store processing and the time that the QUERY command was issued. This
    value is not cumulative; if processing is in a data store when the command
    is issued, a zero value will be returned in the IDLETIME column.

    The format of IDLETIME is:

    `HH:MM:SS.th`

    If IDLETIME exceeds 24 hours, the IDLETIME column will display:

    `24+ HOURS`

PSB
    Returns the name of the PSB being used by the thread.

SCIMEMBER
    Returns the name of the ODBM client that is associated with the thread.

STARTTIME
    Returns the time that the thread started. STARTTIME is UTC and is
    formatted as follows:

    `DDD-YYYY HH:MM:SS.thmiju`

TIME
    Returns the three time values: STARTTIME, IDLETIME, and ACTIVETIME.

URID
    Returns the UR identifier that is associated with the thread.

USERID
    Returns the user ID associated with the thread.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation
programs that communicate with OM.

Output fields

Short label
    Contains the short label that is generated in the XML output.

Long label
    Contains the long label generated in the XML output.

Keyword
    Identifies keyword on the command that caused the field to be generated.
    error appears for output fields that can appear for a nonzero completion
    code. N/A (not applicable) appears for output fields that are always
    returned.

Meaning
    Provides a brief description of the output field.

Table 147. Output field descriptions for the QUERY ODBM TYPE(THREAD) command

Short label	Long label	Keyword	Meaning
ACID	ApplClientID	APPLCLIENTID	The ID of the application client that originated the IMS data store connection request.
ATIME	ActiveTime	ACTIVETIME or TIME	Elapsed time that the thread has been processing in a data store.
ALIAS	AliasName	N/A	The alias name associated with the thread.
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
DSTR	DatastoreName	N/A	Data store name associated with the thread.
ITIME	IdleTime	IDLETIME or TIME	Elapsed time that the thread has been idle (Not processing in a data store).
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
PSB	PSB	PSB	Name of the PSB scheduled by the thread.
SCMBR	SCIMbrName	SCIMEMBER	The SCI member name of the client that is connected to ODBM for the thread.
STIME	StartTime	STARTTIME or TIME	Start time of the thread.
STT	Status	N/A	Status of the thread.
THID	ThreadID	N/A	Thread identifier.
URID	URID	URID	Unit of recovery identifier for the thread.
UID	Userid	USERID	User name associated with the thread.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(THREAD) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 148. Return and reason codes for the QUERY ODBM TYPE(THREAD) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(THREAD) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.

Table 148. Return and reason codes for the QUERY ODBM TYPE(THREAD) command (continued)

Return code	Reason code	Meaning
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'0000501C'	APSB hash table SCAN failed.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 149. Completion codes for the QUERY ODBM TYPE(THREAD) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(THREAD) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
2	PARAMETER SPECIFIED TOO LONG	The NAME() parameter specified is longer than the valid thread name length of 32 characters.
7	NO MATCHES FOUND FOR FILTER	The resource was found, but one or more of the specified filters did not match.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(*) SHOW(ALL)
```

TSO SPOC output:

```
TSO SPOC output:
MbrName     ThreadID                             CC    Status       DatastoreName
ODBM010D    00000001233344129802334877222344     0     INBACKOUT    IMS1
ODBM010D    00055001233344129802334877222981     0     INDOUBT      IMS2
ODBM010D    01234449333344129802334877222344     0     INPREPARE    IMS3
ODBM020D    92833457233344129802334877222344     0     INDOUBT      IMS4
ODBM030D    63200001233344129802334877222344     0     INABTERM     IMS5

(continued)

AliasName  SCIMbrName  Userid   PSB    URID
I01A       HWS1        User01   PSB1   0000000000000001
I02B       HWS1        User02   PSB2   0000000000000025
I03A       HWS2        User03   PSB3   0000000000000009
I04A       HWS2        User03   PSB4   0000000000000087
I05B       HWS1        User05   PSB5   0000000000000022
```

(continued)

```
StartTime                IdleTime      ActiveTime    ApplClientID
182-2008 15:31:15.123456 00:00:00.01   00:00:00.00   CLIENT1
182-2008 15:31:29.237888 00:00:00.00   00:00:00.90   CLIENT2
182-2008 15:31:45.573900 00:00:00.02   00:00:00.00   CLIENT3
182-2008 15:31:18.000001 00:00:00.02   00:00:00.00   CLIENT3
182-2008 15:31:53.111242 00:00:00.01   00:00:00.00   CLIENT5
```

Explanation: The QUERY command will display ODBM thread information about all threads that are known to ODBM.

Example 2 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(*)
```

TSO SPOC output:

```
MbrName   ThreadID                             CC   Status      DatastoreName
ODBM010D  00000001233344129802334877222344    0    INBACKOUT   IMS1
ODBM010D  00055001233344129802334877222981    0    INDOUBT     IMS2
ODBM010D  01234444933344129802334877222344    0    INPREPARE   IMS3
ODBM020D  92833457233344129802334877222344    0    INDOUBT     IMS4
ODBM030D  63200001233344129802334877222344    0    INABTERM    IMS5
```

(continued)

```
AliasName
I01A
I02B
I03A
I04A
I05B
```

Explanation: The QUERY command will display information about all threads that are known to ODBM.

Example 3 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:

```
QRY ODBM TYPE(THREAD) NAME(*) DATASTORE(IMS1)
```

TSO SPOC output:

```
MbrName   ThreadID                             CC    Status      DatastoreName
ODBM010D  00000001233344129802334877222344    0     INBACKOUT   IMS1
```

(continued)

```
AliasName
I01A
```

Explanation: The QUERY command will display information about the threads that are associated with data store IMS1.

Example 4 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:
```
QRY ODBM TYPE(THREAD) NAME(*) SCIMEMBER(HWS2) SHOW(APPLCLIENTID,USERID)
```

TSO SPOC output:

```
MbrName    ThreadID                            CC  Status     DatastoreName
ODBM010D   01234444933344129802334877222344    0   INPREPARE  IMS3
ODBM020D   92833457233344129802334877222344    0   INDOUBT    IMS4

(continued)

AliasName  SCIMbrName  Userid   ApplClientID
IO3A       HWS2        User03   CLIENT3
IO4A       HWS2        User03   CLIENT3
```

Explanation: The QUERY command will display the application client identifier and the user ID of the threads that are associated with SCI member name HWS2.

Example 5 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:
```
QRY ODBM TYPE(THREAD) NAME(*) ALIAS(IO2*) SHOW(PSB,SCIMEMBER)
```

TSO SPOC output:
```
MbrName    ThreadID                            CC  Status     DatastoreName
ODBM010D   00055001233344129802334877222981    0   INDOUBT    IMS2


(continued)
AliasName       SCIMbrName  PSB
IO2B            HWS1        PSB2
```

Explanation: The QUERY command will display the PSB name and the ODBM client name for the thread whose alias begins with "IO2®".

Example 6 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:
```
QRY ODBM TYPE(THREAD) NAME(000*) PSB(P*) SHOW(USERID)
```

TSO SPOC output:
```
MbrName    ThreadID                            CC  DatastoreName  Status
ODBM010D   00000001233344129802334877222344    0   IMS1           INBACKOUT
ODBM010D   00055001233344129802334877222981    0   IMS2           INDOUBT


(continued)

AliasName   Userid   PSB
IO1A        User01   PSB1
IO2B        User02   PSB2
```

Explanation: The QUERY command will display the user ID of the threads whose names begin with 000 and whose associated PSBs' names begin with "P".

Example 7 for QUERY ODBM TYPE(THREAD) command

TSO SPOC input:
```
QRY ODBM TYPE(THREAD) NAME(*) SHOW(TIME)
```

TSO SPOC output:

```
MbrName     ThreadID                            CC   Status      DatastoreName
ODBM010D    00000001233344129802334877222344    0    INBACKOUT   IMS1
ODBM010D    00055001233344129802334877222981    0    INDOUBT     IMS2
ODBM010D    01234444933344129802334877222344    0    INPREPARE   IMS3
ODBM020D    92833457233344129802334877222344    0    INDOUBT     IMS4
ODBM030D    63200001233344129802334877222344    0    INABTERM    IMS5


(continued)

AliasName        StartTime                   IdleTime      ActiveTime
IO1A             182-2008 15:31:15.123456    00:00:00.01   00:00:00.00
IO2B             182-2008 15:31:29.237888    00:00:00.00   00:00:00.90
IO3A             182-2008 15:31:45.573900    00:00:00.02   00:00:00.00
IO4A             182-2008 15:31:18.000001    00:00:00.02   00:00:00.00
IO5B             182-2008 15:31:53.111242    00:00:00.01   00:00:00.00
```

Explanation: The QUERY command will display the starting time values of the threads that are associated with ODBM.

QUERY ODBM TYPE(TRACE) command

Use the QUERY ODBM TYPE(TRACE) command to query information about the Open Database Manager (ODBM) trace.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 470
- "Output fields" on page 470
- "Return, reason, and completion codes" on page 470
- "Examples" on page 471

Environment

The QUERY ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax

```
►►──┬─QUERY─┬──ODBM──TYPE(TRACE)──────────────────────────────────►◄
    └─QRY───┘                    └─SHOW(ALL)─┘
```

Keywords

The following keywords are valid for the QUERY ODBM TYPE(TRACE) command.

SHOW()
   Specifies the ODBM TYPE(TRACE) output fields to be returned. The parameters supported with the SHOW keyword are:

   ALL
      Returns all information about the ODBM trace.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
   Contains the short label that is generated in the XML output.

Long label
   Contains the long label generated in the XML output.

Keyword
   Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
   Provides a brief description of the output field.

Table 150. Output field descriptions for the QUERY ODBM TYPE(TRACE) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
DSTR	DatastoreName	N/A	Data store name.
MBR	MbrName	N/A	Name of the ODBM member that processed the command.
TRACE	Trace	N/A	Trace information.
TRCSTT	TraceStatus	N/A	Trace status: Active or inactive.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY ODBM TYPE(TRACE) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 151. Return and reason codes for the QUERY ODBM TYPE(TRACE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY ODBM TYPE(TRACE) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.

Table 151. Return and reason codes for the QUERY ODBM TYPE(TRACE) command (continued)

Return code	Reason code	Meaning
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 152. Completion codes for the QUERY ODBM TYPE(TRACE) command

Completion code	Completion code text	Meaning
0		The QUERY ODBM TYPE(TRACE) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
10	NO RESOURCES	No resources were found.

Examples

Example 1 for QUERY ODBM TYPE(TRACE) command

TSO SPOC input:
```
QRY ODBM TYPE(TRACE) SHOW(ALL)
```

TSO SPOC output:
```
MbrName     Trace   DatastoreName  CC    TraceStatus
ODBM010D    System  IMS1            0    Active
ODBM020D    System  IMS3            0    Inactive
```

Explanation: The QUERY command will display all ODBM trace data.

QUERY OLC command

For an IMS running with RM (RMENV=Y), the QUERY OLC command returns information about the OLCSTAT DS that is shared by all of the IMS systems participating in global online change.

Subsections:
- "Environment" on page 472
- "Syntax" on page 472
- "Keywords" on page 472
- "Usage notes" on page 473
- "Output fields" on page 473
- "Return, reason, and completion codes" on page 475
- "Examples" on page 476

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY OLC command and keywords can be issued.

Table 153. Valid environments for the QUERY OLC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY OLC	X	X	X
LIBRARY	X	X	X
SHOW	X	X	X

Syntax
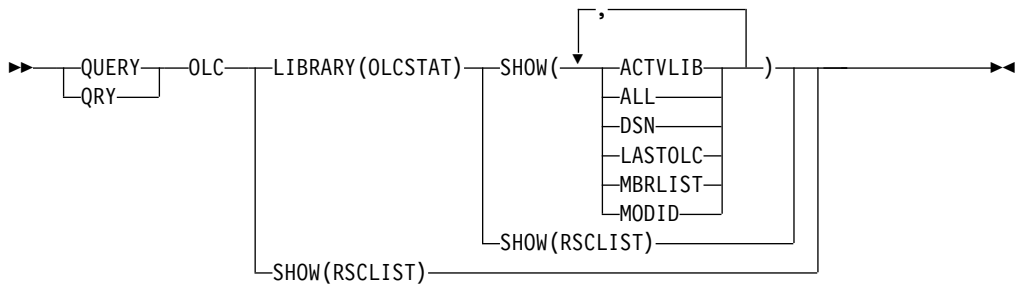
Keywords

The following keywords are valid for the QUERY OLC command:

LIBRARY

Online change library. The library can be the following:

OLCSTAT

OLCSTAT data set contents.

SHOW()

Specifies the library information returned.

ALL

Returns all the output fields except the RSCLIST information.

ACTVLIB

Displays the suffixed online change library names that are currently active. This includes ACBLIBA or ACBLIBB, FMTLIBA or FMTLIBB, and MODBLKSA or MODBLKSB. These are the online change libraries the IMS online system must use at IMS initialization time.

DSN

OLCSTAT data set name.

LASTOLC

Displays the last online change that was successfully performed. If an IMS was down during the last online change and its restart type does not conflict with the last online change that was performed, it will be permitted to warm start. The last online change type is blank, if no online changes have been done. The last online change type is one or more of the following, if at least one online change has been done:

- ACBLIB

- FMTLIB
- MODBLKS

MBRLIST
> Displays the list of IMS systems that are current with the online change libraries. These are the IMS systems that will be permitted to warm start. The IMS systems either participated in the last online change, or cold started since the last online change.

MODID
> Modify ID. The modify ID - 1 represents the number of global online changes that have been performed.

RSCLIST
> Returns the ACBLIB members that are copied from the staging ACBLIB to the active ACBLIB. SHOW(RSCLIST) is valid only when TYPE(ACBMBR) OLC is in progress.

Usage notes

The command response is the same for all of the IMS systems. QUERY OLC is valid for an IMS enabled for global online change, but it is not valid for an IMS enabled for local online change.

QUERY OLC is not supported on an XRF alternate, an RSR tracker, or an FDBR region. It can only be specified through the OM API.

The QUERY OLC LIBRARY command displays information about global online change, such as the current online change libraries and the IMS systems that are current with the online change libraries. QUERY OLC LIBRARY(OLCSTAT) displays the contents of the global online change status data set, OLCSTAT. Specifying the SHOW keyword may optionally show the current active online change libraries, the list of IMS systems that are current with the online change libraries, the modify ID, and the last online change that was done.

For an IMS running without RM services (RMENV=N), the QUERY OLC command returns information about the local OLCSTAT DS of an IMS system. The command response is different for each IMS because each IMS is required to have a unique OLCSTAT DC. In a no RM environment, if more than one IMS is specified in the route list for the QUERY OLC, only the OLCSTAT DS information for the command IMS master is returned. To obtain OLCSTAT DS information from each IMS that is running without RM, the QUERY OLC command must be issued separately to each IMS. To determine which IMS systems are defined with RMENVNO, issue a QUERY MEMBER SHOW(ATTRIB) command.

The command syntax for this command is defined in XML and is available to automation programs which communicate with OM.

Output fields

The following table shows the QUERY OLC output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a nonzero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning
> Provides a brief description of the output field.

Table 154. Output fields for the QUERY OLC command

Short label	Long label	Keyword	Scope	Meaning
ACBL	ACBLIB	ACTVLIB	GBL	The current ACBLIB library. A means that the current ACBLIB library is ACBLIBA. B means that the current ACBLIB library is ACBLIBB.
ADD	ADD	RSCLIST	LCL	Indicates that the resources is being added.
CC	CC	N/A	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified library. Refer to Table 156 on page 476 for more information. The completion code is always returned.
CHG	CHNG	RSCLIST	LCL	Indicates that the resource is being changed.
CPY	COPY	RSCLIST	LCL	Indicates that the resource is being copied.
DBD	DBDName	RSCLIST	LCL	The database or DBD name.
DSN	DSName	DSN	GBL	OLCSTAT data set name.
FMTL	FMTLIB	ACTVLIB	GBL	The current FMTLIB library. A means that the current FMTLIB library is FMTLIBA. B means that the current FMTLIB library is FMTLIBB.
LAST	LastOLC	LASTOLC	GBL	The last successful online change that was successfully performed. If an IMS was down during the last online change and its restart type does not conflict with the last online change that was performed, it will be permitted to warm start. The online change type may include one or more of the following: • ACBLIB • FMTLIB • MODBLKS
LIB	Library	N/A	GBL	Library name. Can be OLCSTAT.
MBR	MbrName	N/A	N/A	IMSplex member that built output line. IMS identifier of the IMS that build the output. IMS identifier is always returned.

Table 154. Output fields for the QUERY OLC command (continued)

Short label	Long label	Keyword	Scope	Meaning
MBRL	MbrList	MBRLIST	GBL	List of IMSplex members that are current with the online change libraries. These are the IMS systems that will be permitted to warm start. The IMS systems either participated in the last online change, or cold started since the last online change.
MODB	MODBLKS	ACTVLIB	GBL	The current MODBLKS and MATRIX libraries. A means that the current libraries are MODBLKSA and MATRIXA. B means that the current libraries are MODBLKSB and MATRIXB.
MODI	Modid	MODID	GBL	The current modify ID. The modify ID - 1 is the number of successful global online changes that have been performed.
PSB	PSBName	RSCLIST	LCL	The program or PSB name.
RFS	RFSH	RSCLIST	LCL	Indicates the resource is being refreshed.
RSC	RSCName	RSCLIST	LCL	The resource name specified on the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY OLC command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 155. Return and reason codes for the QUERY OLC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY OLC command completed successfully.
X'00000004'	X'00001000'	The QUERY OLC command was not processed on the IMS system as the IMS system is not the command master. No information is returned.
X'00000010'	X'0000400C'	The QUERY OLC command failed because it is invalid for an XRF alternate.
X'00000010'	X'00004014'	The QUERY OLC command failed because it is invalid for an RSR tracker.
X'00000010'	X'0000401C'	The QUERY OLC command failed because it is invalid for an FDBR region.

Table 155. Return and reason codes for the QUERY OLC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'0000410C'	The QUERY OLC command is rejected, because global online change is not enabled. Local online change is enabled. Use the /DISPLAY MODIFY command for local online change. If your IMSplex is made up of some IMS systems that support global online change and some that support local online change, route the QUERY OLC command to an IMS that is enabled for global online change. Issue the QUERY MEMBER TYPE(IMS) SHOW(ATTRIB) command to choose an IMS that has global online change enabled.
X'00000010'	X'00004114'	The QUERY OLC LIBRARY(OLCSTAT) command failed because of an error accessing the OLCSTAT data set. A DFS2843 message is sent to the OM output exit as unsolicited output.
X'00000010'	X'00004118'	The QUERY OLC LIBRARY(OLCSTAT) command failed because of an error allocating the OLCSTAT data set. A DFS2848 message is sent to the OM output exit as unsolicited output.
X'00000010'	X'0000411C'	The QUERY OLC LIBRARY(OLCSTAT) command failed because of an error in the OLCSTAT data set contents. One or more of the values is invalid. A DFS2844 message is sent to the OM output exit as unsolicited output.
X'00000014'	X'00005004'	The QUERY OLC command processing failed because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The QUERY OLC command failed because of an internal IMS error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual library.

The following table contains the completion codes that can be returned on a QUERY OLC command.

Table 156. Completion codes for the QUERY OLC command

Completion code	Meaning
0	The QUERY OLC command completed successfully for the library.

Examples

The following are examples of the QUERY OLC command:

Example 1 for QUERY OLC command

TSO SPOC input:
```
QRY OLC LIBRARY(OLCSTAT) SHOW(ACTVLIB,MODID,MBRLIST)
```

TSO SPOC output:

```
 Response for: QUERY OLC LIBRARY(OLCSTAT) SHOW(ACTVLIB,MODID,MBRLIST)
MbrName   CC Library    ACBLIB  FMTLIB  MODBLKS    Modid MbrList
MS3        0 OLCSTAT       B       A       B          1 IMS3,IMS2,SYS3
```

OM API input:

```
CMD (QRY OLC LIBRARY(OLCSTAT) SHOW(ACTVLIB,MODID,MBRLIST))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 15:22:43.468642</statime>
<stotime>2002.16315:22:45.400709</stotime>
<staseq>B7C4A8029DD62884</staseq>
<stoseq>B7C4A80475885248</stoseq>
<rqsttkn1>USRT011 10082243</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS2     ">
<typ>IMS      </typ>
<styp>DBDC</styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntext>IMS not master, cmd ignored</rsntext>
</mbr>
<mbr name="SYS3     ">
<typ>IMS      </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntext>IMS not master, cmd ignored</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMS3     </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>OLC    </kwd>
<input>QUERY OLC LIBRARY(OLCSTAT) SHOW(ACTVLIB,MODID,MBRLIST)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"  llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no" len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"   llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT"  align="right" />
<hdr slbl="LIB"  llbl="Library" scope="GBL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="ACBL" llbl="ACBLIB"  scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
<hdr slbl="FMTL" llbl="FMTLIB"  scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
<hdr slbl="MODB" llbl="MODBLKS" scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
<hdr slbl="MODI" llbl="Modid"   scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="right" />
<hdr slbl="MBRL" llbl="MbrList" scope="GBL" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left"/>
</cmdrsphdr>
<cmdrspdata>
```

```
<rsp>MBR(IMS3   ) CC(   0) LIB(OLCSTAT ) ACBL(B) FMTL(A) MODB(B) MODI(       1)
MBRL(IMS3,IMS2,SYS3) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: QUERY OLC LIBRARY(OLCSTAT) displays the contents of the
OLCSTAT data set, which contains global online change status. This command
displays the active online change libraries, the modify id, and the list of IMS
members that are current with the online change libraries and may therefore warm
list. The output shows that the ACBLIBB data set is active, the FMTLIBB data set is
active, the MODBLKSB data set is active, and the modify id is 2. SYS3 was the
command master that built the output.

Example 2 for QUERY OLC command

TSO SPOC input:

```
QRY OLC LIBRARY(OLCSTAT) SHOW(DSN)
```

TSO SPOC output:

```
MbrName   CC Library  DSName
IMS3       0 OLCSTAT   IMSTESTL.IMS02.OLCSTAT
```

OM API input:

```
CMD (OLC LIBRARY(OLCSTAT) SHOW(DSN))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 15:28:36.353742</statime>
<stotime>2002.16315:28:36.426823</stotime>
<staseq>B7C4A953276CE286</staseq>
<stoseq>B7C4A95339447348</stoseq>
<rqsttkn1>USRT011 10082836</rqsttkn1>
<rc>0200000C</rc><rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS2    ">
<typ>IMS    </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntext>IMS not master; cmd ignored</rsntext>
</mbr>
<mbr name="SYS3    ">
<typ>IMS    </typ>
<styp>DBDC    </styp>
<rc>00000004</rc>
<rsn>00001000</rsn>
<rsntext>IMS not master; cmd ignored</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMS3    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>OLC   </kwd>
<input>QUERY OLC LIBRARY(OLCSTAT) SHOW(DSN)</input>
</cmd>
<cmdrsphdr>
```

```
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"  llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT"  align="right" />
<hdr slbl="LIB" llbl="Library" scope="GBL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="DSN" llbl="DSName"  scope="GBL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS3  ) CC(  0) LIB(OLCSTAT ) DSN(IMSTESTL.IMS02.OLCSTAT) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: This QUERY OLC command displays the OLCSTAT data set name.
IMS3 was the command master that built the output.

Example 3 for QUERY OLC command

TSO SPOC input:

```
QUERY OLC SHOW(RSCLIST)
```

TSO SPOC output:

```
MbrName     CC DBDName  PSBName  ADD
IMS1         0 OLCDB105          Y
IMS2         0 OLCDB105          Y
IMS1         0 OLCDB111          Y
IMS2         0 OLCDB111          Y
IMS1         0 OLCDI111          Y
IMS2         0 OLCDI111          Y
IMS1         0 OLCDX111          Y
IMS2         0 OLCDX111          Y
IMS1         0          OLCPB105 Y
IMS2         0          OLCPB105 Y
IMS1         0          OLCPB111 Y
IMS2         0          OLCPB111 Y
```

OM API input:

```
CMD (QUERY OLC SHOW(RSCLIST))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2006.268 18:28:28.326510</statime>
<stotime>2006.268 18:28:28.341969</stotime>
<staseq>BF75BA3F0266E263</staseq>
<stoseq>BF75BA3F062D136E</stoseq>
<rqsttkn1>USRT001 10112828</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT001 </userid>
<verb>QRY </verb>
<kwd>OLC            </kwd>
<input>QUERY OLC SHOW(RSCLIST) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
```

```
                       len="8" dtype="CHAR" align="left" />
                       <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
                        len="4" dtype="INT" align="right" />
                       <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
                        scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
                       <hdr slbl="RSC" llbl="RSCName" scope="LCL" sort="a" key="3" scroll="no"
                        len="8" dtype="CHAR" align="left" skipb="yes" />
                       <hdr slbl="DBD" llbl="DBDName" scope="LCL" sort="a" key="2" scroll="no"
                        len="8" dtype="CHAR" align="left" skipb="yes" />
                       <hdr slbl="PSB" llbl="PSBName" scope="LCL" sort="a" key="1" scroll="no"
                        len="8" dtype="CHAR" align="left" skipb="yes" />
                       <hdr slbl="ADD" llbl="ADD" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" skipb="yes" />
                       <hdr slbl="CHG" llbl="CHNG" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" skipb="yes" />
                       <hdr slbl="CPY" llbl="COPY" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" skipb="yes" />
                       <hdr slbl="RFS" llbl="RFSH" scope="LCL" sort="n" key="0" scroll="yes"
                        len="1" dtype="CHAR" align="left" skipb="yes" />
                       </cmdrsphdr>
                       <cmdrspdata>
                       <rsp>MBR(IMS2    ) CC(   0) DBD(OLCDB105) ADD(Y) </rsp>
                       <rsp>MBR(IMS2    ) CC(   0) DBD(OLCDX111) ADD(Y) </rsp>
                       <rsp>MBR(IMS2    ) CC(   0) DBD(OLCDB111) ADD(Y) </rsp>
                       <rsp>MBR(IMS2    ) CC(   0) DBD(OLCDI111) ADD(Y) </rsp>
                       <rsp>MBR(IMS2    ) CC(   0) PSB(OLCPB105) ADD(Y) </rsp>
                       <rsp>MBR(IMS2    ) CC(   0) PSB(OLCPB111) ADD(Y) </rsp>
                       <rsp>MBR(IMS1    ) CC(   0) DBD(OLCDB105) ADD(Y) </rsp>
                       <rsp>MBR(IMS1    ) CC(   0) DBD(OLCDX111) ADD(Y) </rsp>
                       <rsp>MBR(IMS1    ) CC(   0) DBD(OLCDB111) ADD(Y) </rsp>
                       <rsp>MBR(IMS1    ) CC(   0) DBD(OLCDI111) ADD(Y) </rsp>
                       <rsp>MBR(IMS1    ) CC(   0) PSB(OLCPB105) ADD(Y) </rsp>
                       <rsp>MBR(IMS1    ) CC(   0) PSB(OLCPB111) ADD(Y) </rsp>
                       </cmdrspdata>
                       </imsout>
```

Explanation: This example shows the output from a QRY OLC SHOW(RSCLIST)
command after an INIT OLC PHASE(PREPARE) TYPE(ACBMBR) command has
been issued.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

QUERY OLREORG command

Use the QUERY OLREORG command to query information about an online
reorganization or alteration of a HALDB partition.

Information that is returned by the QUERY OLREORG command includes the
status, rate, and number of bytes that have been moved.

Reorganizing a database means modifying the structure of a database or changing
how the data in the database is organized in order to improve performance.

Subsections:
- "Environment" on page 481
- "Syntax" on page 481

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY OLREORG command and keywords can be issued.

Table 157. Valid environments for the QUERY OLREORG command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY OLREORG	X	X	
NAME	X	X	
STATUS	X	X	
SHOW	X	X	

Syntax
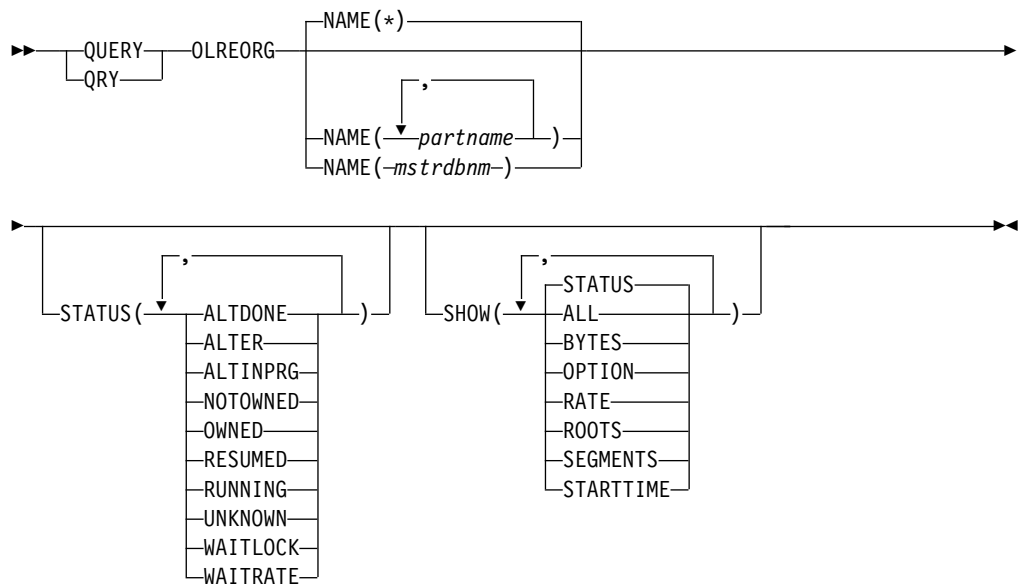
Keywords

The following keywords are valid for the QUERY OLREORG command:

NAME()
>   The names of the PHDAM or PHIDAM partitions that are being reorganized or altered.
>
>   NAME() is optional. A parameter with the wildcard character (*) is not allowed, except as NAME(*) for all defined HALDB partitions. NAME(*) is the default.

SHOW()
> Specifies the output fields to return. The default is SHOW(STATUS).

> **ALL**
>> Returns all of the following output fields on each response line.

> **BYTES**
>> Returns the total number of bytes that have been moved to the output data set.

> **OPTION**
>> Returns the current option, which can be one of the following:
>> - DEL - The output data sets will be deleted at the end of OLR
>> - NODEL - The output data sets will not be deleted at the end of OLR
>> - NOREL - Ownership of the OLR will not be released if IMS terminates before completing the reorganization
>> - REL - Ownership of the OLR will be released if IMS terminates before completing the reorganization

> **RATE**
>> Returns the rate at which the HALDB OLR is running, from 1 to 100.

> **ROOTS**
>> Returns the total number of roots that have been moved to the output data set.

> **SEGMENTS**
>> Returns the total number of segments that have been moved to the output data set.

> **STARTTIME**
>> Returns the local time when OLR was started.

> **STATUS**
>> Returns online reorganization status.

STATUS()
> Displays online reorganizations that possess at least one of the specified statuses. If the STATUS keyword is not specified, any online reorganization with a status of RUNNING, OWNED, NOTOWNED, WAITRATE, or WAITLOCK is returned.

> **ALTDONE**
>> Displays all partitions for which alter processing is complete.

> **ALTER**
>> Displays the status of alter processing for all partitions in a HALDB database that is being altered online. The possible statuses that can be returned are:

>> **ALTINPRG**
>>> Alter processing is not complete for the partition.

>> **ALTDONE**
>>> Alter processing is complete for the specified partition.

>>> Alter processing is not complete for a HALDB database until all partitions show a status of ALTDONE. After all partitions have a status of ALTDONE, you can enable the altered database for new application programs by performing Online Change (OLC) to switch the ACB library from staging to active.

ALTINQUE

The displayed partition is waiting to be altered.

ALTINPRG

Displays all partitions that are currently being altered.

NOTOWNED

Specifies that the output is for the HALDB OLRs that have been temporarily stopped by the TERMINATE OLREORG command and, therefore, are not owned by any IMS.

To inquire on OLRs that have been terminated using the TERMINATE OLREORG command, use the commands QUERY OLREORG STATUS(NOTOWNED) with ROUTE(*) on the command request and /RMLIST DBRC='DB DBD(partname)'. If all systems on the PLEX show status, then OLR has been terminated.

OWNED

Specifies that the output is for HALDB OLRs that are owned by any IMS. The OLRs that are running on the IMS where the command is being processed displays a STATUS of RUNNING. OLRs running on other IMS subsystems displays a STATUS of OWNED.

RESUMED

OLR is resumed after being stopped for some reason such as a TERM OLREORG command or a user abend.

RUNNING

Specifies the output is for the HALDB OLRs that are owned by each IMS for the specified partname or partnames. You can use this keyword to determine which IMS has an online reorganization running for a given partname.

UNKNOWN

Specifies that the output is for those part names on each IMS for which the status of the HALDB OLR cannot be determined. This inability to determine the status can be caused by situations such as the HALDB master or partition being taken offline by a /DBR DB command, the IMS not being authorized to the named partname because of an IRLM failure, or the partition has not been accessed.

WAITLOCK

OLR is waiting for a lock.

WAITRATE

OLR is waiting because of an intentional delay. This intentional delay was caused because a value of less than 100 was specified on the RATE parameter.

Usage notes

You can issue the QUERY OLREORG command only from the OM API. Responses from each IMS to which the command was routed are consolidated by OM. If you specify names on the NAME parameter, response lines are returned for each specified part name. For the default parameter, NAME(*), response lines are returned only for the HALDB partitions that have online reorganizations in progress at each IMS.

A nonzero return code and a nonzero reason code are returned when the command is routed to an XRF alternate system and when the command is routed to an RSR tracking system.

The output for this command is defined in XML and is available to automation programs that communicate with OM.

The QRY OLREORG command can be issued to obtain OLR statistics for an OLR that has been stopped for some reason such as a TERM OLREORG command or a user abend. The data associated with the terminated OLR will be maintained and provided under any of the following conditions:

- The TERMINATE OLR command is issued.
- Abnormal OLR termination occurs (for example, DFS2971W message is issued)
- IMS is normally shut down

Note: If the partition or HALDB master has been taken offline with a /DBR command, or if IMS has been restarted to resume the OLR and the OLR ownership is requested before IMS restart, the status will not be available through the QUERY OLREORG command until the OLR is resumed. The status, however, can be obtained through the LIST.DB or LIST.RECON command.

Output fields

The following table shows the QUERY OLREORG output fields. The columns in the table are as follows:

Short label
    Contains the short label that is generated in the XML output.

Long label
    Contains the long label generated in the XML output.

Show keyword
    Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
    Provides a brief description of the output field.

Table 158. Output fields of QUERY OLREORG

Short label	Long label	Show keyword	Meaning
BYTES	Bytes-Moved	BYTES, ALL	Total number of bytes moved.
CC	CC	N/A	Completion code.
LSTT	LclStat	N/A	Status of HALDB OLR.
MBR	MbrName	N/A	The IMS from which the command was issued.
OPT	Option	OPTION, ALL	Ownership release option specified on the INITIATE OLREORG or UPDATE OLREORG command.
PART	Partition	N/A	Partition name.
RATE	Rate	RATE, ALL	The speed at which HALDB OLR runs. A value of 1 to 100 percent.
RESM	Resumed	N/A	RESUMED status if HALDB OLR is resumed.

Table 158. Output fields of QUERY OLREORG (continued)

Short label	Long label	Show keyword	Meaning
ROOTS	Roots-Moved	ROOTS, ALL	Total number of roots moved.
SEGS	Segs-Moved	SEGMENTS, ALL	Total number of segments moved.
STRTT	StartTime	STARTTIME, ALL	The time OLR started as recorded in the RECON data set.

Return, reason, and completion codes

The OM return and reason codes that might be returned as a result of the QUERY OLREORG command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 159. Return and reason codes for the QUERY OLREORG command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY OLREORG command completed successfully.
X'00000004'	X'00001010'	No matches found for filter.
X'00000008'	X'00002004'	Invalid command keyword.
X'0000000C'	X'00003000'	At least one request was successful.
X'0000000C'	X'00003004'	None of the requests was successful.
X'00000010'	X'00004030'	Command is invalid for LSO=Y.
X'00000010'	X'00004014'	Command was issued on an RSR tracker.
X'00000010'	X'0000400C'	Command was issued on an XRF alternate.
X'00000010'	X'00004200'	The QUERY OLREORG command is not processed because IMS shutdown is in progress.
X'00000014'	X'00005000'	A GETMAIN error occurred.

The following table includes an explanation of the completion codes. Errors unique to the processing of QUERY OLREORG command are returned as completion codes. A completion code is returned for each action against a HALDB partition.

Table 160. Completion codes for the QUERY OLREORG command

Completion code	Meaning
0	The QUERY OLREORG command completed successfully for the partition.
10	Resource name is invalid.
14	Resource is not a partition name.
1C	Resource is a partitioned secondary index.
24	No HALDB OLR is in progress.
28	No DMB is loaded.
CB	Partition is not in specified status.

Example

In this example, the QUERY OLREORG command is routed to IMSA. The command is issued to obtain the information about all of the OLRs that are in progress at IMSA. The output that is returned contains the following information:

- The partition name
- The IMSID
- The status of OLRs in progress
- The rate of OLR
- The number of bytes moved

TSO SPOC input:

```
QRY OLREORG NAME(*) SHOW(ALL)
```

TSO SPOC output:

```
Partition MbrName    CC LclStat  Rate  Bytes-Moved Segs-Moved...
POHIDKA   IMS1        0 RUNNING   100        15678         97...
PDHDOJA   IMS1        0 RUNNING   100         4630         29...


... Roots-Moved  Option          Resumed StartTime
...          11  NODEL, NOREL    Y       2007.296 10:20:21.61
...           5  DEL, REL                2007.296 10:20:21.84
```

OM API input:

```
CMD (QRY OLREORG NAME(*) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1     </xmlvsn>
<statime>2007.296 17:43:42.714976</statime>
<stotime>2007.296 17:43:42.715488</stotime>
<staseq>C163CD37F5860D82</staseq>
<stoseq>C163CD37F5A60342</stoseq>
<rqsttkn1>USRT011 10104342</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>OLREORG         </kwd>
<input>QRY OLREORG NAME(*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PART" llbl="Partition" scope="lcl" sort="a" key="1"
 scroll="no" len="7" dtype="char" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="lcl" sort="n" key="0"
scroll="no"
 len="8" dtype="char" align="left" />
<hdr slbl="CC" llbl="CC" scope="lcl" sort="n" key="0"
scroll="yes"
 len="4" dtype="int" align="right" />
<hdr slbl="LSTT" llbl="LclStat" scope="lcl" sort="n" key="0"
 scroll="yes" len="*" dtype="char" align="left" />
<hdr slbl="RATE" llbl="Rate" scope="lcl" sort="n" key="0"
scroll="yes"
 len="3" dtype="int" align="right" skipb="yes" />
```

```
<hdr slbl="BYTES" llbl="Bytes-Moved" scope="lcl" sort="n"
key="0"
 scroll="yes" len="12" dtype="int" align="right" skipb="yes" />
<hdr slbl="SEGS" llbl="Segs-Moved" scope="lcl" sort="n0" key="0"
 scroll="yes" len="10" dtype="int" align="right" skipb="yes" />
<hdr slbl="ROOTS" llbl="Roots-Moved" scope="lcl" sort="n"
key="0"
 scroll="yes" len="10" dtype="int" align="right" skipb="yes" />
<hdr slbl="OPT" llbl="Option" scope="lcl" sort="n" key="0"
scroll="yes"
 len="8" dtype="char" align="left" skipb="yes" />
<hdr slbl="RESM" llbl="Resumed" scope="lcl" sort="n" key="0"
 scroll="yes" len="1" dtype="char" align="left" skipb="yes" />
<hdr slbl="STRTT" llbl="StartTime" scope="lcl" sort="n" key="0"
 scroll="yes" len="20" dtype="char" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp> PART(POHIDKA ) MBR(IMS1    ) CC(   0) LSTT(RUNNING )
RATE(100)
 BYTES(        15678) SEGS(         97) ROOTS(         11)
 OPT(NODEL   )
 RESM(Y) STRTT(2007.296 10:20:21.61) </rsp>
<rsp> PART(PVHDJ5A ) MBR(IMS1    ) CC(   0) LSTT(RUNNING )
RATE(100)
 BYTES(         4630) SEGS(         29) ROOTS(          5)
 OPT(DEL,REL )
 STRTT(2007.296 10:20:21.84) </rsp>
</cmdrspdata>
</imsout>
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY OTMADESC command

Use the QUERY OTMADESC command to query information about an existing IMS Open Transaction Manager Access (OTMA) destination descriptor.

Subsections:
- "Environment"
- "Syntax" on page 488
- "Keywords" on page 489
- "Usage notes" on page 494
- "Output fields" on page 495
- "Return, reason, and completion codes" on page 496
- Examples

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and its keywords.

Table 161. Valid environments for the QUERY OTMADESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY OTMADESC	X		X

Syntax

```
>>-+-QUERY-+--OTMADESC--+-NAME(--+------------+-destname*--)-+--->
   '-QRY---'            |        '-------,----'              |
                        '-NAME(*)-------------------------'
```

```
>--+----------------------------+--+-------------------------+-->
   |        .----,----.          |  |         .----,----.     |
   '-TYPE(--v-+-IMSCON---+-)-----'  '-TMEMBER(-v--tmem--)-----'
              +-IMSTRAN--+
              +-MQSERIES-+
              '-NONOTMA--'
```

```
>--+-------------------------+--+-MQCOPYMD(-+-Y-+-)-+--+-SYNCTP(-+-N-+-)-+-->
   |        .----,----.       |             '-N-'                '-Y-'
   '-TPIPE(-v--tpipename-)----'
```

```
>--+------------------------+--+-----------------+-->
   '-LTERMOVR(lterm name)---'  '-MQPERST(-+-N-+-)-'
                                          '-Y-'
```

```
>--+------------------------------------+-->
   |        .----,----.                 |
   '-MQREPORT(-v--mqmdreportname-)------'
```

```
>--+----------------------------------+--+-SMEM(-+-N-+-)-+-->
   |        .----,----.               |          '-Y-'
   '-MQFORMAT(-v--mqmdformatname-)----'
```

```
>--+----------------------------+--+-----------------------------+-->
   |        .----,----.          |  |         .----,----.         |
   '-ADAPTER(-v--adaptername-)---'  '-CONVRTR(-v--convertrname-)--'
```

```
         ┌─────────────────┐              ┌────────────────────┐
►────────┤                 ├──────────────┤                    ├────────►
      └─RMTIMS(─▼─remote_IMS─)─┘      └─RMTIMSCON(─▼─rmtIMSconnect─)─┘


         ┌──────────────┐         ┌───────────────┐      ┌──N──┐
►────────┤              ├─────────┤               ├──────┤     ├──►
      └─RMTTRAN(─▼─trancode─)─┘  └─USERID(─▼─user_id─)─┘  └─EXIT(─┴─Y─┴─)─┘


      ┌──────────────┐                    ┌──,──┐
►─────┤              ├────────────────────┤     ├──────────────►
   └─REPLYCHK(─┬─Y─┬─)─┘          └─SHOW(─▼──ADAPTER───)─┘
              └─N─┘                        ├─ALL──────┤
                                           ├─CONVRTR──┤
                                           ├─EXIT─────┤
                                           ├─LTERMOVR─┤
                                           ├─MQAPPLID─┤
                                           ├─MQCOPYMD─┤
                                           ├─MQCORREL─┤
                                           ├─MQFORMAT─┤
                                           ├─MQMSGID──┤
                                           ├─MQPERST──┤
                                           ├─MQREPORT─┤
                                           ├─MQRTF────┤
                                           ├─MQRTQ────┤
                                           ├─MQRTQMGR─┤
                                           ├─REPLYCHK─┤
                                           ├─RMTIMS───┤
                                           ├─RMTIMSCON┤
                                           ├─RMTTRAN──┤
                                           ├─SMEM─────┤
                                           ├─SYNCTP───┤
                                           ├─SYNTIMER─┤
                                           ├─TMEMBER──┤
                                           ├─TPIPE────┤
                                           ├─TYPE─────┤
                                           ├─NONOTMA──┤
                                           ├─IMSCON───┤
                                           ├─IMSTRAN──┤
                                           ├─MQSERIES─┤
                                           └─USERID───┘


      ┌──NOWILDCARD──┐
►─────┤              ├──────────────►◄
   └─OPTION(─┬─NOWILDCARD─┬─)─┘
            └─WILDCARD───┘
```

Keywords

The following keywords are valid for the QUERY OTMADESC command:

ADAPTER(*adaptername*)

A 1- to 8-character name that identifies the IMS Connect adapter. This keyword is an optional parameter for TYPE(IMSCON). When an adapter name is specified, information is displayed from only the descriptors that specify that adapter name. This keyword is ignored if TYPE(NONOTMA) is specified.

CONVRTR(*convertrname*)

A 1- to 8-character converter name associated with the adapter specified on the ADAPTER keyword. When a converter name is specified, information is displayed from only the descriptors that specify that converter name.

EXIT(Y | N)

An optional parameter that filters descriptors based on whether they allow the appropriate exit routine to override the descriptor routing information.

LTERMOVR(*ltermname*)

An optional parameter that filters descriptors based on the logical terminal override name.

MQAPPLID(*mqmdapplidentitydata*)

Filters for descriptors with the specified value or values for the MQAPPLID parameter. MQAPPLID specifies the content of the MQMD_APPLIDENTITYDATA field, a 1- to 32-character user data field in the MQMD data structure for IBM MQ messages.

MQCOPYMD(Y | N)

Filters for descriptors based on whether they specify to copy the MQMD data from the input message to the output message.

MQCORREL(*mqmdcorrel*)

Filters for descriptors with the specified value or values for the MQCORREL parameter. MQCORREL specifies the content of the MQMD_CORRELID field of the MQMD data structure for IBM MQ messages.

MQFORMAT(*mqmdformatname*)

Filters for descriptors with the specified value or values for the MQFORMAT parameter. MQFORMAT specifies the content of the MQMD_FORMAT field of the MQMD data structure for IBM MQ messages. The following values are supported by IMS:

- MQIMSVS
- MQIMS
- NONE

MQMSGID(*mqmdmsgid*)

Filters for descriptors with the specified value or values for the MQMSGID parameter. MQMSGID specifies the content of the MQMD_MSGID field of the MQMD data structure for IBM MQ messages.

MQPERST(N | Y)

Filters TYPE=MQSERIES descriptors based on whether they use message persistance.

MQREPORT(*mqmdreportname*)

Filters for descriptors with the specified value or values for the MQREPORT parameter. MQREPORT specifies the content of the MQMD_REPORT field of the MQMD data structure for IBM MQ messages. The following values are supported:

Parameter value
  Equivalent MQMD structure value

NEWMSGID
   MQRO_NEW_MSG_ID

PASSMSGI
   MQRO_PASS_MSG_ID

COPYMTOC
   MQRO_COPY_MSG_ID_TO_CORREL_ID

PASSCORR
   MQRO_PASS_CORREL_ID

PASSMSCO
   MQRO_PASS_MSG_ID and MQRO_PASS_CORREL_ID

NONE
   MQRO_NONE

MQRTF(*mqmdreplytoformatname*)
   Filters for descriptors with the specified value or values for the MQRTF
   parameter. MQRTF specifies the content of the MQMD_REPLYTOFORMAT
   field of the MQMD data structure for IBM MQ messages.

MQRTQ(*mqmdreplytoqname*)
   Filters for descriptors with the specified value or values for the MQRTQ
   parameter. MQRTQ specifies the content of the MQMD_REPLYTOQ field of the
   MQMD data structure for IBM MQ messages.

MQRTQ(*mqmdreplytoqmgrname*)
   Filters for descriptors with the specified value or values for the MQRTQMGR
   parameter. MQRTQMGR specifies the content of the MQMD_REPLYTOQMGR
   field of the MQMD data structure for IBM MQ messages.

NAME(*destname**)
   A 1- to 8-character destination name that identifies the OTMA destination
   descriptors to display. This keyword is required. If OPTION(WILDCARD) is
   also specified, an asterisk can be used as a wildcard character.

OPTION(NOWILDCARD | WILDCARD)
   Determines whether an asterisk in the NAME keyword is treated as a wildcard
   character. When NOWILDCARD is specified, an asterisk is processed as a
   normal character and a descriptor is displayed only if the destination name
   coded in the descriptor in the DFSYDTx PROCLIB member includes a
   matching asterisk character. When WILDCARD is specified, an asterisk in the
   NAME keyword is treated as wildcard character and any descriptors that have
   a destination name that matches the characters not masked by the asterisk are
   displayed.

REPLYCHK(Y | N)
   An optional parameter that filters descriptors based on whether they have
   OTMA DFS2082 message support enabled.

RMTIMS(*imsname*)
   The name of a remote IMS system that is the destination for OTMA ALTPCB
   output. When a remote IMS name is specified, information is displayed from
   only the descriptors that specify that remote IMS name. The RMTIMS
   parameter is optional and accepts a 1- to 8-character name.

RMTIMSCON(*imsconnectname*)
   The name of a connection to a remote IMS Connect instance that manages the
   TCP/IP communications for a remote IMS system that is the destination for
   OTMA ALTPCB output. When a remote IMS Connect connection name is

specified, information is displayed from only the descriptors that specify that remote IMS Connect connection name. The RMTIMSCON parameter is optional and accepts a 1- to 8-character name of the connection.

RMTTRAN(*trancode*)

An optional parameter that displays the transaction code that is scheduled to process OTMA messages in a destination remote IMS system. When a remote transaction code is specified, information is displayed from only the descriptors that specify that remote transaction code. The RMTTRAN value is a 1- to 8-character name of a transaction.

SHOW()

Specifies which information to display in the output fields.

ADAPTER

Displays the name that identifies the IMS Connect adapter.

ALL

Displays all available information.

CONVRTR

Displays the converter name used by the adapter.

EXIT

For descriptors with TYPE=IMSTRAN, this field displays whether the DFSCMUX0 exit routine can override the descriptor routing information for late response messages.

For other types of descriptors, this field displays whether the DFSYPRX0 and DFSYDRU0 exit routines can override the descriptor routing information for the ALT IOPCB message switch destination.

LTERMOVR

Displays the lterm name specified in the destination descriptor.

MQAPPLID

Displays the IBM MQ user application data that is inserted into the MQMD_APPLIDENTITYDATA field for messages that are routed with this descriptor.

MQCOPYMD

Displays whether the original input values for the IBM MQ data structure (MQMD) are copied to the output message.

MQFORMAT

Displays the value of the MQMD_FORMAT field for messages that are routed to IBM MQ with this descriptor.

MQPERST

Displays whether messages that are routed to IBM MQ with this descriptor are persistent on a synchronous tpipe.

MQREPORT

Displays the value of the MQMD_REPORT field for messages that are routed to IBM MQ with this descriptor.

MQRTF

Displays the value of the MQMD_REPLYTOFORMAT field for messages that are routed to IBM MQ with this descriptor.

MQRTQ

Displays the value of the MQMD_REPLYTOQ field for messages that are routed to IBM MQ with this descriptor.

MQRTQMGR

Displays the value of the MQMD_REPLYTOQMGR field for messages that are routed to IBM MQ with this descriptor.

REPLYCHK

Displays whether DFS2082 message support is enabled for the descriptor.

RMTIMS

Displays the name of the destination remote IMS systems.

RMTIMSCON

Displays the name of remote IMS Connect systems that are managing TCP/IP connections for a destination remote IMS system.

RMTTRAN

The transaction code to be scheduled in the destination remote IMS system.

SMEM

Displays the super member indicator.

SYNCTP

Displays whether messages that are routed to IBM MQ with this descriptor require a synchronous tpipe.

SYNTIMER

Displays the timeout value for synchronous callout processing.

TMEMBER

Displays the name of TMEMBER or the Super Member if SMEM indicates as such.

TPIPE

Displays the TPIPE name under a TMEMBER.

TYPE

Displays the type of descriptor.

NONOTMA

Only displays fields relevant for NONOTMA descriptors.

IMSCON

Only displays fields relevant for IMSCON descriptors.

IMSTRAN

Only displays fields relevant for IMSTRAN descriptors.

MQSERIES

Only displays fields relevant for MQSERIES descriptors.

USERID

Displays the user ID used for transaction authorization in a destination remote IMS system.

SMEM(Y | N)

An optional parameter that specifies whether the TMEMBER name is a super member. When SMEM is specified, information is displayed from only the descriptors that specify the matching Y or N value for SMEM. SMEM and TYPE(NONOTMA) are mutually exclusive.

SYNCTP(N | Y)

Filters TYPE=MQSERIES descriptors based on whether they require a synchronous tpipe.

TMEMBER(*tmem***)**
> An optional parameter that is used to filter by a 1- to 16-character TMEMBER name. When a tmember name is specified, information is displayed from only the descriptors that specify that tmember name. TMEMBER and TYPE(NONOTMA) are mutually exclusive.

TPIPE(*tpipename***)**
> An optional parameter that is used to filter by a 1- to 8-character TPIPE name. When a tpipe name is specified, information is displayed from only the descriptors that specify that tpipe name. TPIPE and TYPE(NONOTMA) are mutually exclusive.

TYPE(IMSCON | IMSTRAN | MQSERIES | NONOTMA)
> An optional keyword that can be IMSCON, IMSTRAN, MQSERIES, or NONOTMA. All output types are displayed if the keyword is not specified. To filter the display, specify IMSCON, IMSTRAN, MQSERIES, or NONOTMA. To include the other keywords, such as TMEMBER, TPIPE, SMEM, ADAPTER, or CONVRTR, use the SHOW keyword. If TYPE(NONOTMA) is specified, the rest of the display parameters will show blanks even if the SHOW keyword is specified.

USERID(*userid***)**
> An optional parameter that displays the user ID used for transaction authorization in a destination remote IMS system. When a user ID is specified, information is displayed from only the descriptors that specify that user ID. The USERID value is a 1- to 8-character RACF user ID.

Usage notes

The QUERY OTMADESC command is used to display the values specified in the destination routing descriptors that are identified in the NAME() keyword. The results returned by the QUERY OTMADESC command can be filtered by secondary keywords, such as TPIPE(). For example, when QUERY OTMADESC NAME(*abc**) TPIPE(*xyz*) OPTION(WILDCARD) is specified, information is returned for only descriptors that have a destination name that starts with *abc* and a tpipe name of *xyz*.

Use the SHOW() keyword to control what information is displayed from the OTMA destination descriptors found by the command. For example, if you include SHOW(TMEMBER) in the command, the TMEMBER values are returned.

The QUERY OTMADESC command does not process asterisks as wildcard characters. However, you can use an asterisk as a wildcard character in the NAME() keyword if you specify OPTION(WILDCARD) when you issue the command.

No log records are written when the command is issued.

The QUERY OTMADESC command can be issued on an Extended Recovery Facility (XRF) alternate or a remote site recovery (RSR) tracking environment. The information retrieved might differ from the active system because of timing issues. For example, take the scenario where the UPDATE OTMADESC command is issued on the active system, the log record is written, and the QUERY OTMADESC command is issued on both the active and alternate system. The information about the active system reflects the current information whereas the alternate system reflects the old information. This is caused by the log record being read and updated in the alternate system.

You can issue a valid QUERY OTMADESC command that does not return any results because one of the filter keywords excluded the entire result set. For example, querying descriptors with both the ADAPTER keyword and the MQFORMAT keyword will never return any results because those keywords are never used together in a valid descriptor. If your query does not return any results, ensure that no mutually exclusive filtering keywords were specified.

Output fields

The following table shows the QUERY OTMADESC output fields. The columns in the table are:

Short label
>Contains the short label generated in the XML output.

Long label
>Contains the column heading for the output field in the formatted output.

Keyword
>Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
>Provides a brief description of the output field.

Table 162. Output fields for the QUERY OTMADESC command

Short label	Long label	Keyword	Meaning
ADAP	Adapter	ADAPTER	Adapter name.
CC	CC	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned.
CCTXT	CCText	N/A	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
CVRTR	Converter	CONVRTR	Converter name.
DEST	DestName	NAME	Destination name.
IMSRPY	IMSReply	IMSREPLY	Indicator that shows whether a DFS2082 message can be received as a response for a synchronous program switch request.
MBR	MbrName	N/A	Member name.
RMTIMS	RmtIMS	RMTIMS	Name of a remote IMS system.
RMTIMSCON	RmtIMSCon	RMTIMSCON	Connection to a remote IMS Connect instance.
RMTTRAN	RmtTran	RMTTRAN	Transaction sent to the remote IMS system.
SMEM	SMem	SMEM	Indicator that shows whether the destination is a super member.
SYNTO	Syntimer	SYNTIMER	Timeout value for synchronous callout processing.
TMEM	TMember	TMEMBER	OTMA TMEMBER name.
TPIPE	TPipe	TPIPE	TPIPE name.
TYPE	Type	TYPE	Output type.
UID	Userid	USERID	User ID.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY OTMADESC command. The OM return and reason codes that might be returned as a result of the QUERY OTMADESC command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 163. Return and reason codes for the QUERY OTMADESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See Table 164 for details.
X'02000008'	X'00002000'	The command contains an invalid verb or no client is registered for the verb.
X'02000008'	X'00002004'	The command contains an invalid primary keyword or no client is registered for the keyword.
X'02000008'	X'00002028'	The command contains an invalid keyword.
X'02000008'	X'0000202C'	The command contains an unknown positional parameter.
X'02000008'	X'00002034'	The command contains an incomplete keyword parameter.
X'02000008'	X'00002038'	The command is missing a required parameter.
X'02000008'	X'0000203C'	The command contains an invalid keyword parameter value.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 164. Completion codes for the QUERY OTMADESC command

Completion code	Completion code text	Meaning
0	Command completed successfully	The QUERY OTMADESC command completed successfully for the resource.
165	No Desc found meet any criteria	No descriptors were found based on the filters specified.
166	No entries found	Command failed for QUERY OTMADESC, because no entries were found in the table of destination routing descriptors.

Example 1 for QUERY OTMADESC command

The following are examples of the QUERY OTMADESC command:

Example 1 for QUERY OTMADESC command

TSO SPOC input:

```
QUERY OTMADESC NAME(OTMACL*) SHOW(TYPE,TMEMBER,SMEM)
```

TSO SPOC output:

```
DestName MbrName  CC Type    TMember  SMem
OTMACL*  IMSA      0 IMSCON   HWS2     N
```

Explanation: An asterisk is specified in the name specified in the NAME() keyword, but because the default value for the OPTION keyword is NOWILDCARD and OPTION(WILDCARD) is not specified, the QUERY command displays only OTMACL* and not OTMACL99. It treats OTMACL* as a stand-alone entry in the table of destination routing descriptors.

Example 2 for QUERY OTMADESC command

TSO SPOC input:

```
QUERY OTMADESC NAME(OTMACL*) SHOW(ALL) OPTION(WILDCARD)
```

TSO SPOC output:

```
DestName MbrName  CC Type    TMember  TPipe     SMem
OTMACL99 IMSA      0 IMSCON   HWS1     HWS1TP01  N
OTMACL*  IMSA      0 IMSCON   HWS2               N
```

Explanation: The QUERY command will display both OTMACL99 and OTMACL*, because the OPTION(WILDCARD) keyword is specified. It treats OTMACL* as a mask for a group of names including OTMACL99 and, the stand-alone entry OTMACL* in the table of destination routing descriptors.

Example 3 for QUERY OTMADESC command

TSO SPOC input:

```
QUERY OTMADESC NAME(OTMD*) TYPE(IMSCON) SHOW(TMEMBER,SMEM,SYNTIMER) OPTION(WILDCARD)
```

TSO SPOC output:

```
DestName MbrName    CC Type    TMember      SMem Syntimer
OTMD*    IMS1        0 IMSCON   HWS1         N     1000
OTMDSC01 IMS1        0 IMSCON   HWS1         N     3500
OTMDSC02 IMS1        0 IMSCON   HWS1         N   999999
OTMDSC03 IMS1        0 IMSCON   HWS1         N     1000
OTMDSC04 IMS1        0 IMSCON   SM01         Y     4000
OTMDSC08 IMS1        0 IMSCON   HWS1         N
```

Explanation: The last descriptor, OTMDSC08, does not show any Syntimer value because this descriptor is normally not used for synchronous callout. Syntimer is only for synchronous callout processing.

Example 4 for QUERY OTMADESC command: RMTIMSCON

The following example shows all OTMA destination descriptors that are used to route ALTPCB output to the remote IMS Connect connection ICON2B.

TSO SPOC input:

```
QRY OTMADESC NAME(*) RMTIMSCON(ICON2B) SHOW(ALL) OPTION(WILDCARD)
```

TSO SPOC output, first screen:

```
PLEX1                       IMS Single Point of Control
Command ===>


 ----------------------   Plex . . PLEX1  Route . . IMS1     Wait . . 5:00
 Response for: QRY OTMADESC NAME(*) RMTIMSCON(ICON2B) SHOW(ALL)... More:    >
DestName MbrName   CC Type      TMember         TPipe    SMem Adapter Converte
DESC0001 IMS1       0 IMSCON    HWS1            TPIPE01  N
DESC0002 IMS1       0 IMSCON    HWS1            TPIPE02  Y
DESC0003 IMS1       0 IMSCON    HWS1            TPIPE03  N
DESC0004 IMS1       0 IMSCON    HWS1            TPIPE03  N
SM01000B IMS1       0 IMSCON    SM01            TPSM01   Y
T01R2BI2 IMS1       0 IMSCON    HWS1            TPIPE01  N
T02R2BI2 IMS1       0 IMSCON    HWS1            TPIPE02  N
```

TSO SPOC output, second screen scrolling right:

```
PLEX1                       IMS Single Point of Control
Command ===>


 ----------------------   Plex . . PLEX1  Route . . IMS1     Wait . . 5:00
 Response for: QRY OTMADESC NAME(*) RMTIMSCON(ICON2B) SHOW(ALL)... More:   <
DestName MbrName   Converter Syntimer RmtIMSCon RmtIMS   RmtTran Userid
DESC0001 IMS1                         ICON2B    IMS2
DESC0002 IMS1                         ICON2B    IMS2
DESC0003 IMS1                         ICON2B    IMS2     APOL12
DESC0004 IMS1                         ICON2B    IMS2     APOL12  BILL
SM01000B IMS1                         ICON2B    IMS2
T01R2BI2 IMS1                         ICON2B    IMS2
T02R2BI2 IMS1                         ICON2B    IMS2
```

Explanation: The QUERY OTMADESC command is issued to IMS1. Because
NAME(*) and OPTION(WILDCARD) are specified, command processing searches
all OTMA destination descriptors. However, because RMTIMSCON(ICON2B) is
also specified, only descriptors that include RMTIMSCON=ICON2B are displayed.
Because SHOW(ALL) is specified, all output fields are shown, even if the
parameter is not specified in the descriptor. The RmtIMS, RmtTran, and Userid
output fields are also related to the IMS-to-IMS TCP/IP connection.

Example 5 for QUERY OTMADESC command: RMTTRAN

The following example shows all OTMA destination descriptors that set a
transaction code, APOL12, in ALTPCB messages that are routed to a remote IMS
system for processing.

TSO SPOC input:
```
QRY OTMADESC NAME(*) RMTTRAN(APOL12) SHOW(ALL) OPTION(WILDCARD)
```

TSO SPOC output, first screen:
```
PLEX1                       IMS Single Point of Control
Command ===>


 ----------------------   Plex . . PLEX1  Route . . IMS1     Wait . . 5:00
 Response for: QRY OTMADESC NAME(*) RMTTRAN(APOL12) SHOW(ALL) O... More:    >
DestName MbrName   CC Type      TMember         TPipe    SMem Adapter Converte
DESC0003 IMS1       0 IMSCON    HWS1            TPIPE03  N
DESC0004 IMS1       0 IMSCON    HWS1            TPIPE03  N
```

TSO SPOC output, second screen scrolling right:
```
PLEX1                       IMS Single Point of Control
Command ===>


 ----------------------   Plex . . PLEX1  Route . . IMS1     Wait . . 5:00
```

```
   Response for: QRY OTMADESC NAME(*) RMTTRAN(APOL12) SHOW(ALL) O...  More: <
DestName MbrName  Converter Syntimer RmtIMSCon RmtIMS   RmtTran Userid
DESC0003 IMS1                         ICON2B    IMS2     APOL12
DESC0004 IMS1                         ICON2B    IMS2     APOL12  BILL
```

Explanation: The QUERY OTMADESC command is issued to IMS1. Because
NAME(*) and OPTION(WILDCARD) are specified, command processing searches
all OTMA destination descriptors. However, because RMTTRAN(APOL12) is also
specified, only descriptors that include RMTTRAN=APOL12 are displayed.
Because SHOW(ALL) is specified, all output fields are shown, even if the
parameter is not specified in the descriptor. The RmtIMSCon, RmtIMS, and Userid
output fields are also related to the processing of ALTPCB messages on remote
IMS systems.

Example 6 for QUERY OTMADESC command: TPIPE for connections to a remote IMS system

The following example shows all OTMA destination descriptors that specify the
same tpipe, TPSM01, which is used to route ALTPCB messages to a remote IMS
system for processing.

TSO SPOC input:

```
QRY OTMADESC NAME(*) TPIPE(TPSM01) SHOW(ALL) OPTION(WILDCARD)
```

TSO SPOC output, first screen:

```
 PLEX1                     IMS Single Point of Control
 Command ===>

   ----------------------  Plex . . PLEX1  Route . . IMS1     Wait . . 5:00
 Response for: QRY OTMADESC NAME(*) TPIPE(TPSM01) SHOW(ALL) OPT...  More:    >
DestName MbrName  CC Type    TMember       TPipe  SMem Adapter  Converte
SM01000A IMS1      0 IMSCON   SM01          TPSM01 Y
SM01000B IMS1      0 IMSCON   SM01          TPSM01 Y
SM01000C IMS1      0 IMSCON   SM01          TPSM01 Y
SM01000D IMS1      0 IMSCON   SM01          TPSM01 Y
SM01000E IMS1      0 IMSCON   SM01          TPSM01 Y
SM01000F IMS1      0 IMSCON   SM01          TPSM01 Y
SM01000G IMS1      0 IMSCON   SM01          TPSM01 Y
SM01000H IMS1      0 IMSCON   SM01          TPSM01 Y
```

TSO SPOC output, second screen scrolling right:

```
 PLEX1                     IMS Single Point of Control
 Command ===>

   ----------------------  Plex . . PLEX1  Route . . IMS1     Wait . . 5:00
 Response for: QRY OTMADESC NAME(*) TPIPE(TPSM01) SHOW(ALL) OPT...  More: <
DestName MbrName  Converter Syntimer RmtIMSCon RmtIMS   RmtTran Userid
SM01000A IMS1                         ICON2A    IMS2
SM01000B IMS1                         ICON2B    IMS2
SM01000C IMS1                         ICON2C    IMS2
SM01000D IMS1                         ICON2D    IMS2
SM01000E IMS1                         ICON2E    IMS2
SM01000F IMS1                         ICON2F    IMS2
SM01000G IMS1                         ICON2G    IMS2
SM01000H IMS1                         ICON2H    IMS2
```

Explanation: The QUERY OTMADESC command is issued to IMS1. Because
NAME(*) and OPTION(WILDCARD) are specified, command processing searches
all OTMA destination descriptors. However, because TPIPE(TPSM01) is also
specified, only descriptors that include TPIPE=TPSM01 are displayed. Because

SHOW(ALL) is specified, all output fields are shown, even if the parameter is not specified in the descriptor. The messages queued to tpipe TPSM01 are retrieved by a local instance of IMS Connect and sent on the connection identified under RmtIMSCon to the remote IMS system identified under RmtIMS for processing.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY OTMATI command

Use the QUERY OTMATI command to display information about IMS OTMA message workload.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 501
- "Usage notes" on page 504
- "Output fields" on page 504
- "Return, reason, and completion codes" on page 506
- "Examples" on page 506

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 165. Valid environments for the QUERY OTMATI command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY OTMATI	X		X
CMTMODE	X		X
GRPNAME	X		X
LTERM	X		X
MODNAME	X		X
MSGAGE	X		X
SHOW	X		X
SYNCLVL	X		X
TMEMBER	X		X
TPIPE	X		X
TRANCODE	X		X
USERID	X		X

Syntax

```
►►─┬─QUERY─┬─OTMATI─┬─TMEMBER(*)────────────────────────┬─►
   └─QRY───┘        │              ┌─,───────────┐       │
                    └─TMEMBER(─▼─┬─tmemname──┬─)──┘
                                 └─tmemname*─┘

   ┌─TPIPE(*)───────────────────┐
►─┤                             ├─┬──────────────────────────────┬─►
   │          ┌─,───────────┐   │ │          ┌─,──────────┐      │
   └─TPIPE(─▼─┬─tpipename──┬─)──┘ └─TRANCODE(─▼─┬─tranname──┬─)──┘
             └─tpipename*─┘                     └─tranname*─┘

►─┬───────────────────────┬─┬────────────────────────┬─►
   │        ┌─,───────┐    │ │         ┌─,───────┐    │
   └─LTERM(─▼─┬─lterm──┬─)─┘ └─USERID(─▼─┬─userid──┬─)─┘
             └─lterm*─┘                 └─userid*─┘

►─┬───────────────────────────┬─┬────────────────────────────┬─►
   │          ┌─,──────────┐   │ │          ┌─,──────────┐    │
   └─GRPNAME(─▼─┬─grpname──┬─)──┘ └─MODNAME(─▼─┬─modname──┬─)──┘
               └─grpname*─┘                   └─modname*─┘

►─┬──────────────────┬─┬──────────────────┬─────────────────────────────►
   │         ┌─,──┐   │ │         ┌─,──┐   │    ┌─MSGAGE(nnnnn)─┐
   └─CMTMODE(─▼┬─0─┬)─┘ └─SYNCLVL(─▼┬─0─┬)─┘
              └─1─┘               ├─1─┤
                                 └─2─┘

   ┌─SHOW(MSGCNT)───────────────┐
►─┤                             ├─►◄
   │       ┌─,──────────┐       │
   └─SHOW(─▼─┬─AGINGVAL───┬─)───┘
            ├─ALL────────┤
            ├─CMTMODE────┤
            ├─CONVID─────┤
            ├─CTTKN──────┤
            ├─GRPNAME────┤
            ├─LTERM──────┤
            ├─MODNAME────┤
            ├─MSGAGE─────┤
            ├─MSGTKN─────┤
            ├─SYNCLVL────┤
            ├─TIMEOUTVAL─┤
            ├─TRANCODE───┤
            ├─TYPE───────┤
            └─USERID─────┘
```
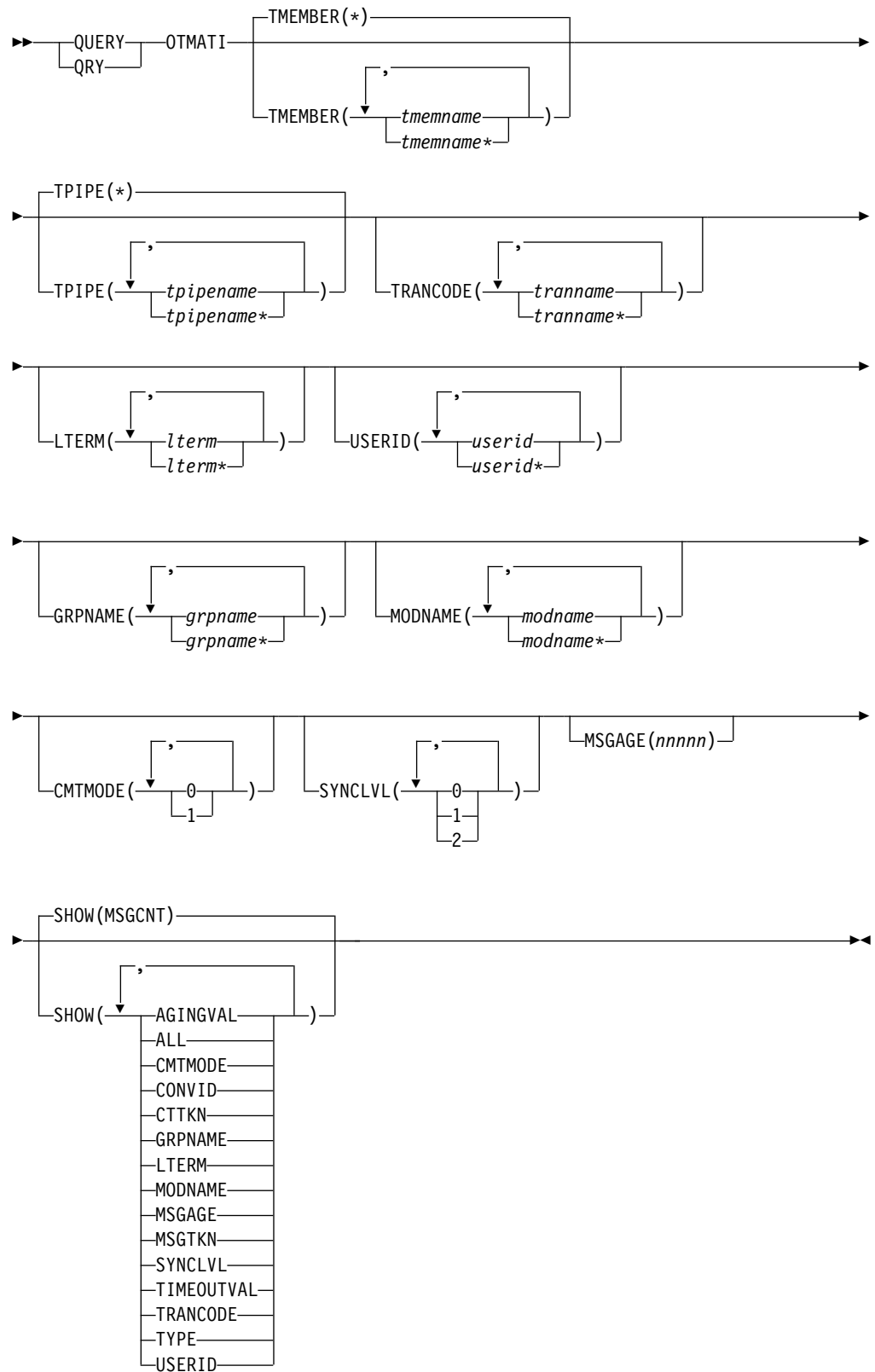
Keywords

The following keywords are valid for the QUERY OTMATI command:

CMTMODE

Specifies the commit mode of the workload to be displayed. Specify 0 to display all workloads that are in commit-then-send (CM0) mode, which is supported on both persistent and transaction sockets and supports only synch level CONFIRM. Specify 1 to display all workloads that are in send-then-commit (CM1) mode, which is supported on both persistent and transaction sockets and supports synch level NONE, CONFIRM, and SYNCH.

GRPNAME

Specifies a 1- to 8-character RACF Group ID that is included in the security prefix of the message. Any group name that exceeds 8 characters is flagged as an error.

LTERM

Specifies a 1- to 8-character override LTERM name that is included in the OTMA message state prefix. Any LTERM name that exceeds 8 characters is flagged as an error.

MODNAME

Specifies a 1- to 8-character message output descriptor name that is included in the OTMA message state prefix. This descriptor is associated with the transaction or the program to be scheduled.

MSGAGE

Specifies the minimum amount of clock time since the message (YTIB) became active. The MSGAGE value has a range of 1 through 86400. This value represents the number of seconds within a 24-hour period. Any values outside of this range is an error.

TMEMBER

Specifies a 1- to 16-character OTMA target member name. The member is a client of OTMA, such as IMS Connect.

TPIPE

Specifies a 1- to 8-character OTMA transaction pipe name.

TRANCODE

Specifies a 1- to 8-character transaction code associated with the program that is scheduled.

SYNCLVL

Specifies the synch level. Specify 0 for a synch level of NONE, which requires no acknowledgment from the client. Specify 1 for a synch level of CONFIRM, which requires the client to acknowledge delivery of output messages. Specify 2 for a synch level of SYNCH, for two-phase commit processing that involves multiple participants in sync point processing managed through z/OS Resource Recovery Services (RRS).

USERID

Specifies a 1- to 8-character RACF user ID that is included in the security prefix of the message.

SHOW

Specifies the output fields to be returned. The filters supported with the SHOW keyword are:

AGINGVAL

Displays the aging value (how often the cached user ID accessor environment element (ACEE) should be refreshed). The aging value is either the message aging value or the client aging value and comes from the message control prefix or from the message state prefix, respectively.

ALL
 Displays all information about the OTMA message workload.

CMTMODE
 Displays the commit mode. 0 represents "Commint-then-send" mode, and 1 represents "Send-then-commit" mode.

CONVID
 Displays the conversation ID that is associated with a message, if any.

 You can use the conversation ID and the message age to identify messages that cannot be returned to the client. If the message age of a message is unusually large, it is likely that the client connection terminated and the conversation cannot be continued. To terminate the conversation, note the conversation ID and specify it in the /EXIT CONV command.

CTTKN
 Displays the context token when a transaction is in a two-phase commit that involves multiple participants in sync point processing managed through RRS.

GRPNAME
 Displays the RACF group ID.

LTERM
 Displays the override LTERM name.

MODNAME
 Displays the override MODNAME.

MSGAGE
 Displays the minimum age in seconds since the message (TIB) became active.

MSGCNT
 Displays the total number of active TIBs associated with messages, depending on the various parameter values requested.

MSGTKN
 Displays the correlator token.

TIMEOUTVAL
 Displays the timeout value for CM1 that is missing ACK.

TRANCODE
 Displays the transaction code that is associated with the message.

TYPE
 Displays the message type, such as SMB transaction, CPIC transaction, IMS command, message switch, message recoverable, conversational transaction, or response.
 SMB SMB transaction
 CPC CPIC transaction
 CMD IMS command
 APC Message switch
 RCV Recoverable transaction
 CON Conversational transaction
 EMH Fast Path transaction
 RSP Transaction response

USERID
 Displays the user ID that is included in the security data prefix of the message.

SYNCLVL
>Displays the synch level of NONE, CONFIRM, or SYNCH.

Usage notes

The QUERY OTMATI command can be issued both in the active and the alternate system, whether it be on an XRF alternate or RSR tracker system.

The QUERY OTMATI command can be specified through the OM API, including the TSO SPOC and IBM Management Console for IMS and Db2 for z/OS (Management Console) user interface. The output of this command is displayed also in the TSO SPOC or in Management Console.

If the QUERY OTMATI command is issued without the SHOW keyword, the display will show the workloads on each of the IMS instances in the sysplex together with the TMEMBER name, TPIPE name, and the total number of active messages. If any keywords other than SHOW are specified, those keywords will be used as a filter to display the total number of messages in the queue as a subset of the total workload. If the SHOW keyword is specified, the total number of active messages will not be displayed; instead, each individual active message will be displayed together, filtered by the other specified keywords. No log records are written.

Output fields

The following table shows the QUERY OTMATI output fields. The columns in the table are:

Short label
>Contains the short label generated in the XML output.

Long label
>Contains the column heading for the output field in the formatted output.

Keyword
>Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
>Provides a brief description of the output field.

Table 166. Output fields for the QUERY OTMATI command

Short label	Long label	Keyword	Meaning
AGINGVAL	AgingVal	AGINGVAL	RACF ACEE aging value. The value can be: • a number between 0 and 99999 (seconds) • ">99999" (the default) • a blank, which indicates that the TIB was processed in the back-end system

Table 166. Output fields for the QUERY OTMATI command (continued)

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 506 for more information. The completion code is always returned.
CCTXT	CCText	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
CONVID	ConvID	CONVID	The four-digit ID of the conversation that a message is a part of.
CMTMODE	CmtMode	CMTMODE	Commit mode. The value can be one of the following: **0** Commit mode 0 **1** Commit mode 1 **1*** Transaction instance block (TIB) is processed in the back-end system and is always a commit mode 1 transaction.
CTTKN	CtTkn	CTTKN	Context token.
GRPNAME	GrpName	GRPNAME	RACF group name.
LTERM	Lterm	LTERM	Override LTERM name.
MBR	MbrName	N/A	IMSplex member that built the output line. The IMS identifier is always returned.
MODNAME	MODname	MODNAME	Override MODNAME.
MSGAGE	MsgAge	MSGAGE	Age of the message in the system. The value can be a number between 0 and 86400. For TIBs that are in the system for more than 86400 seconds, or one day, ">86400" is displayed.
MSGCNT	MsgCnt	MSGCNT	Number of entries that fall in the same filtering criteria. The value can be a number between 0 and 99999, or ">99999", which indicates that the number of TIBs exceeds five digits.
MSGTKN	MsgTkn	MSGTKN	Client token.
MSGTYP	MsgType	TYPE	Message type.
SYNCLVL	SyncLvl	SYNCLVL	Synch level.
TMEM	TMember	TMEM	TMEMBER name.
TMOVAL	TimeoutVal	TIMEOUTVAL	Timeout value for CM1 that is missing ACK. The value can be a number between 0 and 255, or a blank, which indicates that the TIB was processed in the back-end system.
TPIPE	TPipe	TPIPE	Tpipe name.
TRAN	Trancode	TRAN	Transaction name.
UID	Userid	USERID	User ID.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY OTMATI command. The OM return and reason codes that may be returned as a result of the QUERY OTMATI command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 167. Return and reason codes for the QUERY OTMATI command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See Table 168 for details.
X'02000008'	X'00002000'	The command contains an invalid verb or no client is registered for the verb.
X'02000008'	X'00002004'	The command contains an invalid primary keyword or no client registered for the keyword.
X'02000008'	X'00002028'	The command contains an invalid keyword.
X'02000008'	X'0000202C'	The command contains an unknown positional parameter.
X'02000008'	X'00002034'	The command contains an incomplete keyword parameter.
X'02000008'	X'00002038'	The command is missing a required parameter.
X'02000008'	X'0000203C'	The command contains an invalid keyword parameter value.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 168. Completion codes for the QUERY OTMATI command

Completion code	Completion code text	Meaning
0	Command completed successfully	The QUERY OTMATI command completed successfully for the resource.
4	Command completed unsuccessfully	The QUERY OTMATI command failed.
166	Command completed successfully	The queried TMEMBER currently has no transaction instance blocks (YTIBs).

Examples

The following are examples of the QUERY OTMATI command:

Example: QUERY OTMATI

TSO SPOC input:
```
QUERY OTMATI
```

TSO SPOC output:
```
TMember    TPipe       MbrName   CC    MsgCnt
MQ         APPLA       IMSA      0      102
MQ         APPLB       IMSB      0      201
IMSB       WAS         APPLC     0      301
```

Explanation: With the QUERY OTMATI command with no parameters, the display shows the active OTMA send-then-commit messages represented by transaction instance blocks (YTIBs) that are currently running, waiting for a response, or both. TI stands for transaction instance for OTMA.

Example: QUERY OTMATI MSGAGE(8)

TSO SPOC input:
```
QRY OTMATI MSGAGE(8)
```

TSO SPOC output:
```
TMember    TPipe       MbrName   CC    MsgCnt    MsgAge
HWS1       TPIPE1      IMSA      0       2         8
HWS2       TPIPE2      IMSA      0       3         9
```

Explanation: If the QUERY OTMATI command is issued with the MSGAGE keyword, the display will show a subset of the display of Example #1 that has messages with an age of 8 seconds or more. The MsgAge column shows the minimum age found for messages in that subset. For example, the output shows that a subset of the two messages for TPIPE(TPIPE1) contains messages that have an age of 8 seconds or more. The three messages for TPIPE (TPIPE2) contain messages that also have an age of 8 seconds or more and the least age found for its messages was 9 seconds. The display can further be filtered using any keywords, such as TRANCODE and USERID.

Example: QUERY OTMATI MSGAGE(8) SHOW(MODNAME)

TSO SPOC input:
```
QUERY OTMATI MSGAGE(8) SHOW(MODNAME)
```

TSO SPOC output:
```
TMember    TPipe       MbrName   CC    MsgAge    Userid
HWS1       TPIPE1      IMSA      0       10       SVL01
HWS1       TPIPE1      IMSA      0        8       SVL08
HWS2       TPIPE2      IMSA      0        9       IMS02
HWS2       TPIPE2      IMSA      0       16       IMS07
HWS2       TPIPE2      IMSA      0       11       IMS08
```

Explanation: If the QUERY OTMATI command is issued with a MSGAGE keyword and a SHOW keyword specifying MODNAME, the display will expand the display of Example #2 but without the MsgCnt column. Instead of grouping the messages as in Example #2, each message is displayed separately.

Example: QUERY OTMATI TMEMBER(*) TPIPE(*) SHOW(ALL)

In the following example, TMEMBER(*) TPIPE(*) SHOW(ALL) is specified to show all of the attributes and values that are associated with processing messages on all of the tpipes of each OTMA client that is connected to OTMA. In this case, HWS1

is the only OTMA client and it has only one tpipe in OTMA, tpipe 9999. There are three messages queued to tpipe 9999 for two different transactions: CDEBTRN4 and APOL11. Transaction CEDBTRN4 is a conversational transaction, and the messages for the two instances of CEDBTRN4 are differentiated by their conversation IDs (ConvID): 0002 and 0001.

TSO SPOC input:

```
QUERY OTMATI TMEMBER(*) TPIPE(*) SHOW(ALL)
```

TSO SPOC output:

```
TMember        TPipe     MbrName    CC MsgAge Trancode CmtMode SyncLvl MsgType
HWS1           9999      IMS1        0     25 CDEBTRN4 1       0       SMB,CON
HWS1           9999      IMS1        0     82 APOL11   1       1       SMB,RSP
HWS1           9999      IMS1        0     58 CDEBTRN4 1       1       SMB,CON

GrpName Userid  Lterm    MODname MsgTkn              TimeoutVal AgingVal
SYS1    USRT003                  0000000000000000           120 >99999
SYS1    USRT003                  0000000000000000           120 >99999
SYS1    USRT003                  0000000000000000           120 >99999

CtTkn            ConvID
0000000000000000 0002
0000000000000000
0000000000000000 0001
```

Example: QUERY OTMATI TMEMBER(*) TPIPE(*) SHOW(CONVID)

In the following example, TMEMBER(*) TPIPE(*) SHOW(CONVID) is specified to show the conversation ID, if any, that is associated with each message on all of the tpipes of each tmember (OTMA client) that is connected to OTMA. In this case, HWS1 is the only OTMA client and it has only one tpipe in OTMA, tpipe 9999. There are three messages queued to tpipe 9999. Only the first and last messages are part of a conversational transaction, as indicated by the conversation IDs 0002 and 0001.

TSO SPOC input:

```
QUERY OTMATI TMEMBER(*) TPIPE(*) SHOW(CONVID)
```

TSO SPOC output:

```
TMember        TPipe     MbrName    CC ConvID
HWS1           9999      IMS1        0 0002
HWS1           9999      IMS1        0
HWS1           9999      IMS1        0 0001
```

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

QUERY PGM command

Use the QUERY PGM command to query information about program resources.

A program resource defines the application program requirements for application programs that run under the control of the DB/TM environment, as well as for application programs that access databases through DBCTL.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 510
- "Usage notes" on page 515
- "Equivalent IMS type-1 commands" on page 515
- "Output fields" on page 515
- "Return, reason, and completion codes" on page 523
- "Examples" on page 525

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 169. Valid environments for the QUERY PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY PGM	X	X	X
NAME	X	X	X
SHOW	X	X	X
STATUS	X	X	X

Syntax

```
►►─┬─QUERY─┬─PGM─┬─────NAME(*)──────────────┬──
   └─QRY───┘     │          ┌─,─┐           │
                 └─NAME(─▼─┬─name──┬─)─┘
                           └─name*─┘

   ┌──────────────────────┐
   │            ┌─,─┐      │
   ─┬─SHOW(─▼─┬─ALL───────┬─)─┬──►◄
            ├─BMPTYPE────┤
            ├─DEFN───────┤
            ├─DEFNTYPE───┤
            ├─DOPT───────┤
            ├─FP─────────┤
            ├─GLOBAL─────┤
            ├─IMSID──────┤
            ├─GPSB───────┤
            ├─LANG───────┤
            ├─LOCAL──────┤
            ├─MODEL──────┤
            ├─RESIDENT───┤
            ├─SCHDTYPE───┤
            ├─STATUS─────┤
            ├─TIMESTAMP──┤
            └─TRANSTAT───┘
    ├─SHOW(─┬─DB───┬─)─┤
    │       ├─RTC──┤   │
    │       ├─TRAN─┤   │
    │       └─WORK─┘   │
    └─SHOW(EXPORTNEEDED)─┘
```

```
                                                              ,
                                                ┌────────────────┐
                                             ┌──▼──┬─DB-NOTAVL─┬──┐
                            ─STATUS(─────────┘     ├─IOPREV────┤  )─┘
                                                   ├─LOCK──────┤
                                                   ├─NOTINIT───┤
                                                   ├─STOSCHD───┤
                                                   └─TRACE─────┘
```
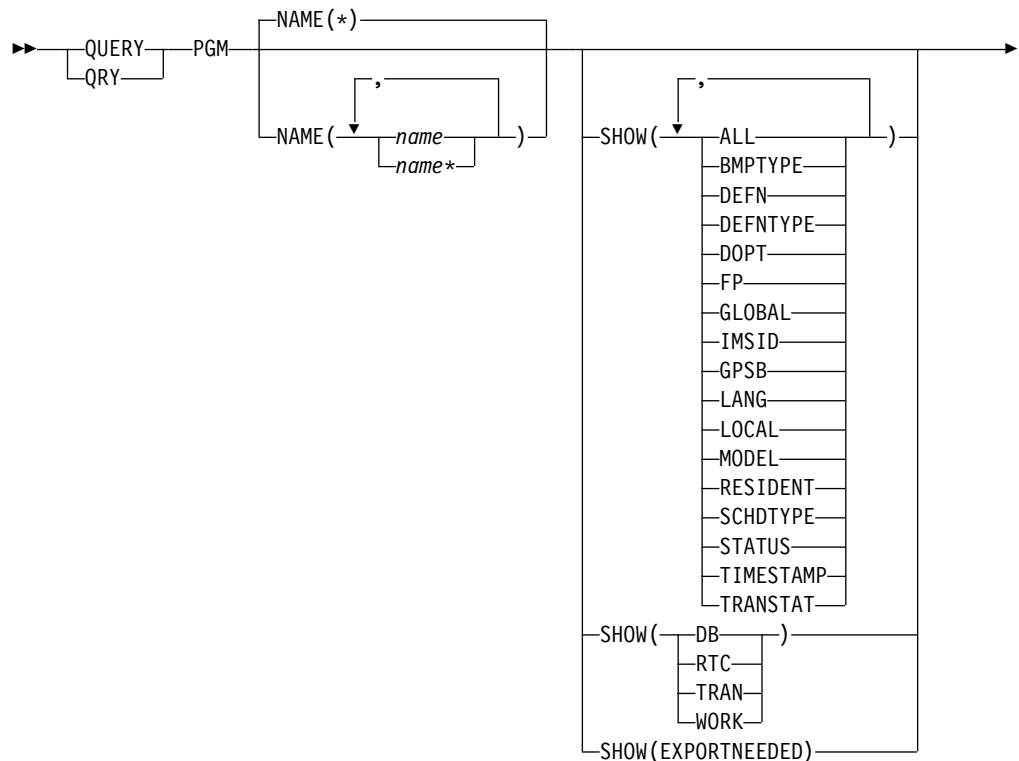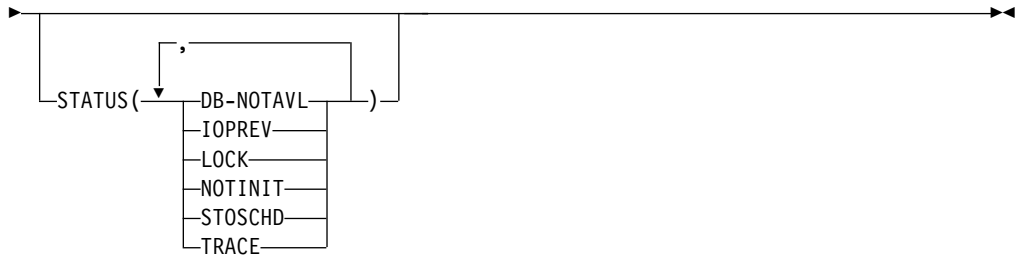
Keywords

The following keywords are valid for the QUERY PGM command:

NAME
Specifies the 1-8 character name of the program. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*) which returns all program resources.

SHOW
Specifies the program output fields to be returned. The program name is always returned, along with the name of the IMS that created the output, the region type, and the completion code. The filters supported with the SHOW keyword are:

ALL
Returns all information about the program itself. Other SHOW keywords can be specified to return information about resources related to the program.

Restriction: The `ExportNeeded` column, which is returned when SHOW(EXPORTNEEDED) is specified, is not returned with SHOW(ALL).

BMPTYPE
BMP type option. Specifies whether the program executes in a BMP type region or not. A BMP type region might be a BMP region or a JBP region.

PSBs scheduled by DB2 stored procedures, by programs running under WebSphere® Application Server, and by other users of the ODBA interface may be defined with BMPTYPE Y or N.

DB The databases referenced by the PSB associated with this program. If the PSB intent list is not resident, the ACBLIB is searched to obtain the database information.

The QRY PGM SHOW(DB) command will not show the database names for which the dynamic (DOPT) PSB has intent.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(DB) individually.

DEFN
Specifies that the resource definitions are to be returned.

The program attributes that can be returned are: BMPTYPE, DOPT, FP, GPSB, LANG, RESIDENT SCHDTYPE, TRANSTAT, the repository create and update time stamps, and the IMS runtime create, update, import, and access time stamps.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes, including those defined globally in the repository and those defined locally in the IMS system, are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE, MODEL, STATUS, WORK, DB, RTC, or TRAN.
- The LclStat, LModelName, LModelType, and LDefnType columns, which are returned on the QRY PGM SHOW(ALL) command, are not returned with SHOW(DEFN).
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).
- When querying program information from the repository, the SHOW(DEFN) filter is not supported when used with the STATUS filter. The runtime filter of STATUS is not valid with SHOW(DEFN), SHOW(DEFN,GLOBAL), SHOW(DEFN,LOCAL), SHOW(DEFN,IMSID), SHOW(DEFN,IMSID,GLOBAL) or SHOW(DEFN,IMSID,LOCAL).

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

DEFNTYPE
Definition type that the descriptor or resource was defined with.

DOPT
Dynamic option.

EXPORTNEEDED
Specifies that only the program resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the programs that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each program name specified on the command. If the specified program does not need to be exported to the repository, an N is returned in the ExportNeeded column.

SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords and is valid only when IMS is enabled with the IMSRSC repository.

The QUERY PGM SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources and descriptors that have not been exported to the IMSRSC repository, you can issue the EXPORT DEFN TARGET(REPO) OPTION(CHANGESONLY) command to write the resource and descriptor definitions to the repository so that they are available for the next IMS cold start.

FP Fast Path option.

GLOBAL
Specifies that the stored resource definitions from the repository are to be returned. If SHOW(GLOBAL,DEFN) is specified, the global resource definitions from the repository are returned by the command master IMS. SHOW(GLOBAL,DEFN) is valid only when the command master IMS is enabled to use the repository.

GPSB
Generated PSB option.

IMSID
Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only when command master IMS is enabled to use the repository.

When SHOW(IMSID) is specified with the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

SHOW(IMSID) cannot be specified with any other SHOW filters other than DEFN and GLOBAL. If SHOW(IMSID,GLOBAL) is specified, GLOBAL is ignored; that is, SHOW(IMSID,GLOBAL) is treated as SHOW(IMSID). SHOW(DEFN,IMSID,LOCAL) is treated as SHOW(DEFN,LOCAL).

LANG
Language interface of the application program. The language interface is displayed only for programs defined as GPSB(Y) and programs defined as DOPT(Y) LANG(JAVA).

LOCAL

Specifies that the runtime resource definitions from the IMS system are to be returned.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

MODEL

Model name and model type used to create this resource. If the descriptor or resource is created with one or more of the attributes defined and no model specified, the model name and model type is the default descriptor. The model name and model type are blank for IMS-defined resources and descriptors and queue-only transactions created by the DFSINSX0 exit. The CREATE command specified without the LIKE keyword creates a descriptor or resource using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSPG1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor or resource using a model. The descriptor or resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor or resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

RESIDENT

Resident option, which indicates whether the PSB is accessed in local storage. The local runtime value for the resident option is shown. The resident option definition is also shown, if it is different from the runtime value.

The RESIDENT(Y) option takes effect at the next IMS restart, unless an error is encountered such as no ACB for the program specification block (PSB) in the ACBLIB or IMS catalog, or if the program was created or updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart.

RTC

routing code descriptors associated with this program.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(RTC) individually.

SCHDTYPE

Scheduling type, which indicates whether this application program can be scheduled into more than one message region or batch message region simultaneously.

STATUS

Program status. For a description of the possible program status returned, see the STATUS keyword under the Output fields table.

TIMESTAMP

The creation time (TIMECREATE), last update time (TIMEUPDATE), last

access time (TIMEACCESS), and last import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

TRAN
Transactions associated with this program.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(TRAN) individually.

TRANSTAT
Transaction level statistics option.

WORK

Work in progress for the program specified on NAME parameter and its associated resources. The QRY PGM SHOW(WORK) command can be issued before a DELETE, IMPORT or UPDATE command to check for any work in progress for the specified program and any of its associated resources. Any work in progress might cause the subsequent DELETE, IMPORT or UPDATE commands to fail. The QRY PGM SHOW(WORK) command returns the work status for the program specified. If no work is in progress for the specified resource, a response line is returned with a work status of blanks.

Notes:
1. SHOW(WORK) specified with NAME(*) might have a performance impact on the processing of the command.
2. You cannot specify this filter with other SHOW filters; you must specify SHOW(WORK) individually.
3. The QRY PGM SHOW(WORK) command is not valid on an XRF alternate.

STATUS()

Selects programs for display that possess at least one of the specified program status. This selection allows for additional filtering by program status. The program status is returned as output, even if the SHOW(STATUS) was not specified.

DB-NOTAVL
A database used by this program is not available, either because it is not defined, or because it is not authorized.

IOPREV
A BMP, IFP, or JBP program cannot complete scheduling, because I/O prevention has not completed. Further I/O requests to data sets are inhibited.

LOCK
Sets the STATUS() filter to return information about programs that are locked.

NOTINIT

Sets the STATUS() filter to return information about programs that are not initialized and therefore cannot be used.

STOSCHD

Sets the STATUS() filter to return information about programs for which program scheduling is stopped.

TRACE

Sets the STATUS() filter to return information about programs that are being traced.

Usage notes

Program resources combined with transactions define the scheduling and resource requirements for an application program. The program resource describes an application program that operates in a message processing region, Fast Path message-driven program region, batch processing region, batch message processing region, or CCTL threads.

This command can be issued only through the Operations Manager API. This command applies to DB/DC, DBCTL and DCCTL systems.

This command is allowed on XRF alternate and RSR tracker systems.

The output of the QUERY PGM command displays the sizes of PSBs, which you can use to determine the impact of the PSB on the PSB pools that are used in the online IMS system. This provides an alternative to using the output of the ACB Maintenance utility to calculate PSB pool size requirements.

If you want to display information about resource definitions, specify SHOW(DEFN). If you want to know which IMS systems have the resource defined and also know the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource defined, specify SHOW(IMSID).

Equivalent IMS type-1 commands

The following table shows variations of the QUERY PGM command and the type-1 IMS commands that perform similar functions.

Table 170. Type-1 equivalents for the QUERY PGM command

QUERY PGM command	Similar IMS type-1 command
QUERY PGM SHOW(ALL)	/DISPLAY PROGRAM, /DISPLAY STATUS PROGRAM
QUERY PGM SHOW(DB)	/DISPLAY PSB
QUERY PGM SHOW(RTC)	/DISPLAY PSB
QUERY PGM SHOW(TRAN)	/DISPLAY PROGRAM
QUERY PGM SHOW(WORK)	/DISPLAY MODIFY ALL

Output fields

The following table shows the QUERY PGM output fields. The columns in the table are:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
> Provides a brief description of the output field.

Table 171. Output fields for the QUERY PGM command

Short label	Long label	Keyword	Scope	Meaning
BMPT	LBmpType	BMPTYPE, DEFN	LCL	BMP type. The output is returned from the local IMS.
				N The program does not execute in a BMP type region. It might execute in an IMS TM MPP, JMP or IFP region, or it might use the ODBA interface or the DRA interface. Use this specification for programs that run in IMS TM MPP, JMP, and IFP regions, or PSBs that are scheduled by CICS programs using DBCTL and other users of the DRA interface. This is the default.
				Y The program executes in a BMP type region. It might execute in an IMS BMP region or a JBP region. Any associated transactions are assigned normal and limit priority values of zero.
CC	CC	N/A	LCL	Completion code.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DB	DBName	DB	LCL	Database referenced by the program.

Table 171. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
DFNT	LDefnType	DEFNTYPE	LCL	Definition type, which can be one of the following: **CREATE** Defined by a CREATE command. **DDLCRE** Defined by the DDL CREATE PROGRAMVIEW command. **DFSINSX0** Defined by user exit DFSINSX0. The program can only be exported if the export option was set. **IMPORT** Defined by an IMPORT command. **IMS** Defined by IMS. DBF#FPU0 is a program created by IMS for the Fast Path Utility. **MODBLKS** Defined by system definition in the MODBLKS data set. The definition type changes from MODBLKS to UPDATE if an UPDATE PGM command is issued to change the attributes of a MODBLKS-defined program. **UPDATE** Defined by system definition in the MODBLKS data set, but changed into a dynamic resource by an UPDATE command.
DOPT	LDOPT	DOPT, DEFN	LCL	Dynamic option (Y) or not (N). The output is returned from the local IMS. **N** The PSB associated with this application program is not located dynamically. The control blocks for the PSB must exist in an ACBLIB or the IMS catalog, otherwise the program is set to a NOTINIT-*xx-reason* status and cannot be scheduled. **Y** The PSB associated with this program is located dynamically. Each time the program associated with this PSB is scheduled, the latest copy of the PSB is loaded. When the program terminates, the PSB is deleted from the PSB pool. Until the PSB is required to process a transaction, the ACB for the PSB does not need to be in the ACBLIB or, if the IMS management of ACBs is enabled, in the IMS catalog. A new version of the PSB can be defined and added to the ACBLIB or the IMS catalog at any time. IMS loads the new version of the ACB the next time the PSB is scheduled. A DOPT PSB cannot be scheduled if the control blocks for the DBDs that the PSB references are not in the ACBLIB or the IMS catalog.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the program has been exported to the IMSRSC repository. The value can be Y or N.

Table 171. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
FP	LFP	FP, DEFN	LCL	Fast Path exclusive program (E) or not (N). The output is returned from the local IMS.
				E The program is a Fast Path-exclusive application program.
				N The program is not a Fast Path application program.
GPSB	LGPSB	GPSB, DEFN	LCL	Generated PSB generated by IMS (Y) or not (N). The output is returned from the local IMS.
				N The PSB associated with the program is not generated by IMS. The control blocks for the PSB must exist in the ACBLIB or, if the IMS management of ACBs is enabled, in the IMS catalog, otherwise the program is set to a NOTINIT-*xx-reason* status and cannot be scheduled.
				Y The PSB and application control block (ACB) that are associated with the program are generated by IMS. The scheduling process of all environments generates a PSB containing an I/O PCB and an alternate modifiable PCB.
				You do not need to generate the PSB and ACB or submit DDL to define the PSB. The ACB for the PSB is not loaded from either ACBLIB or, if the IMS management of ACBs is enabled, the IMS catalog. Thus, I/O to retrieve the ACB from either the ACBLIB or the IMS catalog is eliminated.
				The generated PSB contains an I/O PCB named IOPCBbbb and a modifiable, alternate PCB named TPPCB1bb. With an alternate modifiable PCB, an application can use the CHNG call to change the output destination and send output to a destination other than the input destination.
IMSID	IMSid	IMSID	GBL	Returns from the repository the IMSIDs that have the resource defined.
LANG	LPgmLang	LANG, DEFN	LCL	Language interface. The output is returned from the local IMS.
				ASM/CBL Assembler or COBOL
				JAVA Java™ (can only run in a Java dependent region)
				PASCAL PASCAL
				PLI PL/I

Table 171. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
LRSDNT	LRsdnt	ALL, RESIDENT	LCL	Local runtime value of the resident option. Indicates whether the program PSB resides in local storage.
				N The PSB associated with the named program resource is not made resident in storage. If a program is defined as resident but encounters an error during IMS restart, N is set. The PSB is loaded at scheduling time.
				Y The PSB associated with the named program resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the PSB and initializes it. A resident program is accessed from local storage, which eliminates I/O to retrieve the ACB.
LSTT	LcLStat	STATUS	LCL	Local application program status.
				DB-NOTAVL A database used by this program is not available, either because it is not defined, or because it is not authorized.
				IOPREV A BMP program containing GSAM cannot complete scheduling because I/O prevention has not completed. Further I/O requests to data sets are inhibited.
				LOCK Program is locked.
				NOTINIT-*xx-reason* Programs that are not initialized and therefore cannot be used. NOTINIT is displayed in the format NOTINIT-*xx-reason*. *xx* is the code that identifies the unique location in one module where this reason code is set, which is used by IBM for diagnostic purposes. If the suggested action is to call IBM, the *xx* value helps IBM identify exactly where this resource was marked bad. DFSPDIR MACRO defines each reason code that might be set in the program bad reason code (field PDIRBADR) and identifies the module that sets it. NOTINIT-00 indicates that the reason is unknown. Action: 1. *reason* explains the reason code *xx* in abbreviated text format up to 13 characters.
				STOSCHD Program scheduling is stopped.
				TRACE Program is being traced.
MBR	MbrName	N/A	LCL	IMSplex member that build the output line.
MDLN	LModelName	MODEL	LCL	Model name. Name of the resource used as a model to create this resource. DFSDSPG1 is the IMS descriptor name for programs.

Table 171. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
MDLT	LModelType	MODEL	LCL	Model type, either RSC or DESC. RSC means that the resource was created using another resource as a model. DESC means that the resource was created using a descriptor as a model.
PGM	PgmName	PGM	LCL	Program name that references the database.
RBMPT	BmpType	BMPTYPE, DEFN	GBL	BMP type. The output is returned from the repository.
RDOPT	DOPT	DOPT, DEFN	GBL	Dynamic option (Y) or not (N). The output is returned from the repository.
REPO	Repo	DEFN	GBL	Indicates whether the line shows the stored resource definitions. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
RFP	FP	FP, DEFN	GBL	Fast Path exclusive program (E) or not (N). The output is returned from the repository.
RGNT	LRgnType	N/A	LCL	Region type in which program can run. Some programs can run in additional region types. For example, a program defined with a program type of MSG can run in a BMP under certain conditions. The output is returned from the local IMS. BMP indicates a batch message processing region. FPU indicates a Fast Path utility region. IFP indicates a Fast Path message processing region. JBP indicates a Java batch message processing region. JMP indicates a Java message processing region. MPP indicates an MPP processing region.
RGPSB	GPSB	GPSB, DEFN	GBL	Generated PSB generated by IMS (Y) or not (N). The output is returned from the repository.
RLANG	PgmLang	LANG, DEFN	GBL	Language interface. The output is returned from the repository.
RRSDNT	Rsdnt	DEFN, RESIDENT	GBL	Resident value from the repository.
RRGNT	RgnType	N/A	GBL	Region type in which program can run. The output is returned from the repository.
RSCHD	SchdType	SCHDTYPE, DEFN	GBL	Schedule type. The output is returned from the repository.
RSDNT	LDRsdnt	ALL, RESIDENT	LCL	Local deferred resident value that takes effect at the next IMS restart. A value of Y is shown if a program was defined as resident but could not be made resident at IMS restart time because no ACB existed for the PSB. This program can become resident during the next IMS restart only if there is an ACB for the PSB in the ACBLIB or IMS catalog.
RTC	Rtcode	RTC	LCL	Routing code associated with the program.

Table 171. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
RTLS	TranStat	TRANSTAT, DEFN	GBL	Transaction level statistics logged (Y) or not (N). The output is returned from the repository.
RTMCR	TimeCreate	DEFN	GBL	Create time from the repository. This is the time the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN	GBL	Update time from the repository. This is the time the resource was last updated in the repository.
SCHD	LSchdType	SCHDTYPE, DEFN	LCL	Schedule type. The output is returned from the local IMS. **PARALLEL** The application program can be scheduled into more than one message region or batch message region simultaneously. **SERIAL** The application program can only be scheduled in one region at a time.
TLS	LTranStat	TRANSTAT, DEFN	LCL	Transaction level statistics logged (Y) or not (N). The output is returned from the local IMS. **N**　Transaction level statistics logging is not active. **Y**　Transaction level statistics logging is active.
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the resource was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. The output is returned from the local IMS. For a program resource, the following actions update the last access time: • Program is scheduled. • CREATE command or DFSINSX0 exit references the resource as a model.
TMCR	LTimeCreate	TIMESTAMP	LCL	The time that the resource was created with a CREATE PGM command, an IMPORT command that creates the program, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT. The output is returned from the local IMS.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the resource was last imported, if applicable. The import time is retained across warm start and emergency restart. The output is returned from the local IMS.
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated as a result of the UPDATE PGM command or the IMPORT command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
TRAN	Tran	TRAN	LCL	Transaction associated with the program.

Table 171. Output fields for the QUERY PGM command (continued)

Short label	Long label	Keyword	Scope	Meaning
WRK	Work	WORK	LCL	Work is in progress for the program or one of its associated resources. The work in progress can be one of the following:
				ANOTHER CMD IN PROGRESS Another command (such as DELETE or UPDATE) to delete or update the program is already in progress, or a command to delete or update a transaction referencing the program to be updated is in progress.
				SCHEDULED Program is scheduled.

Table 172. Reason information for NOTINIT-xx-reason status

Reason	Meaning
ACBLIBREAD	I/O error reading ACBLIB.
ALIAS	Alias name error.
BLDL	BLDL miscellaneous error trying to build ACBLIB directory.
DBFINTP0	Module DBFINTP0 returns nonzero return code.
DBNOTINIT	Program references a database that is not initialized and has a NOTINIT status. Action: 2.
DMBINCOMPTBL	DMB incompatibility.
DMBNUM	No DDIR control block for database with this DMB number exists. Action: 1.
DMBPOOL	DMB pool shortage.
EOD	EOD marker found before DMB.
FPDB	The database is a Fast MSDB or DEDB that is defined in a non-Fast Path (FP=N) system. The database cannot be used.
FPMISMATCH	The Fast Path database DDIR control block does not match with the non-Fast Path PCB.
FPRESTART	A Fast Path error occurred during restart. Action: 1.
INTLISTDB	Intent list contains unknown database. Action: 5.
INTLISTLEN	Intent list read length is invalid.
INTLISTLEN0	Intent list read length is zero.
INTLISTREAD	Intent list read error.
KSDSESDS	Missing KSDS/ESDS.
NAMEMISMATCH	PSB/PDIR names do not match.
NOACB	No block found in ACBLIB.
NOAMP	DMB not pointing to access method prefix block (AMP).
NOAMPOFLWDCB	Access method prefix block (AMP) missing overflow DCB.
NODB	A database this program references is not defined. No database DDIR control block exists. Action: 5.
	See message DFS563I for the name of the database that is not defined.
NODMB	No DMB exists in ACBLIB for the database. Action: 2, 5, or both.

Table 172. Reason information for NOTINIT-xx-reason status (continued)

Reason	Meaning
NOPSB	No PSB exists in ACBLIB for the program. Action: 3.
NOTDMB	Not a DMB.
NOTPSB	Not a PSB. A DMB by the same name as the program is defined in ACBLIB, instead of a PSB. If this resource should be a database, create the database with a CREATE DB command. If this resource should be a program, perform a DBDGEN, PSBGEN, ACBGEN and ACBLIB online change to define this resource as a PSB instead of a DMB. Action: 4.
PROCOPTL	PCB specifies PROCOPT=L for online.
PSBLEN	Bad PSB length.
PSBLEVEL	The IMS release level at which this PSB is generated using ACBGEN does not match the IMS release level of this IMS. Perform a PSBGEN, ACBGEN, and ACBLIB online change as needed to generate this PSB at the correct IMS release level. Action: 4.
PSBLOGICAL	PSB is missing a logical relationship database. Action: 4.
PSBPOOL	PSB pool shortage.
REPOCHGLIST	The program name exists in the change list in the IMSRSC repository for this IMS, and the change list is being processed or the change list processing failed because of an error. This program cannot be used, scheduled, updated, exported, or referenced as a model until it is successfully imported so that the runtime program definition and the stored definition in the repository are synchronized. Action: 6.
SIZEMISMATCH	Size mismatch.
WRONGPSB	Wrong PSB.

Note: Actions that can be taken to initialize the program are:
1. Call IBM.
2. Perform an ACBLIB online change to add the DMB to ACBLIB.
3. Perform an ACBLIB online change to add the PSB to ACBLIB.
4. Perform an ACBLIB online change to correct PSBs, DMBs, or both.
5. Perform a MODBLKS online change or issue a CREATE DB command to create the database; then issue an UPDATE PGM START(SCHD) command to reset the NOTINIT status.
6. If IMS change list processing failed and message DFS4411E was issued, correct the problem and take one of the following actions:
   - Shut down IMS and restart it in order to try processing the change list again.
   - Issue the IMPORT DEFN SOURCE(REPO) NAME(*pgmname*) TYPE(PGM) command to import the program, where *pgmname* is the program name.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 173. Return and reason codes for the QUERY PGM command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000004'	X'00001010'	No resources were found to be returned. The resource names specified might be invalid, or: • There were no resources that match the filter specified. • There were no resources that had work to display for the SHOW(WORK) specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'00000008'	X'00002040'	Invalid filter or filter combination. An invalid filter might be an invalid parameter specified with the SHOW keyword or STATUS keyword. An invalid filter combination might be more than one filter specified (SHOW, STATUS), or more than one SHOW filter specified that includes DB, RTC, TRAN, or WORK.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'0000000C'	X'00003004'	Command was successful for none of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'00004004'	No CQS address space.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004016'	The command failed because it is not valid in a DCCTL environment.
X'00000010'	X'00004018'	No resource structure, or resource structure is not available.
X'00000010'	X'00004024'	No Fast Path defined. SHOW(RTC) is not permitted.
X'00000010'	X'00004100'	Resource structure is full.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.

Table 173. Return and reason codes for the QUERY PGM command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Repository access is denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server is being shut down.
X'00000010'	X'0000450B'	Repository Server is not available.
X'00000010'	X'0000450C'	Repository Server is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields related to the IMSRSC repository.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'0000501C'	IMODULE GETMAIN error.
X'00000014'	X'00005100'	RM request error.
X'00000014'	X'00005104'	CQS error.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY PGM command is rejected because the resource list is unusable. Contact IBM Software Support. Send in the output from the command in error, and a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the IMSRSC repository RID and RMD data sets.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 174. Completion codes for the QUERY PGM command

Completion code	Completion code text	Meaning
0		Command completed successfully for program.
10	NO RESOURCES FOUND	Program name is invalid, or the wildcard parameter specified does not match any resource names.
28	NO DMB LOADED	The DMB that is associated with the program does not exist in ACBLIB.

Examples

The following are examples of the QUERY PGM command:

Example 1 for QUERY PGM command

TSO SPOC input:

```
QUERY PGM NAME(PGM0000%,APOL1,BMP255,DBF*0,DFSIVP67,AUTPSB7,JAVPSB1,
JAVTESTJ,DCSQL7A,DCSQL6C,JVMJBP1) SHOW(ALL)
```

TSO SPOC output:

(screen 1)

PgmName	MbrName	CC	LRgnType	LBMPType	LFP	LDOPT	LGPSB	LRsdnt	LTranStat	LPgmLang
APOL1	IMS1	0	MPP	N		N	N	N	N	N	
AUTPSB7	IMS1	0	JBP	Y		N	N	N	N	N	
BMP255	IMS1	0	BMP	Y		N	N	N	N	N	
DBF#FPU0	IMS1	0	FPU	N	E	Y	N	N	N	
DCSQL6C	IMS1	0	MPP	N		N	N	Y	N	N	PLI
DCSQL7A	IMS1	0	MPP	N		N	N	Y	N	N	ASM/CBL
DFSIVP67	IMS1	0	JBP	Y		N	N	N	N	N	
JAVPSB1	IMS1	0	IFP	N	E	N	N	Y	N	
JAVTESTJ	IMS1	0	JMP	N		N	N	N	N	N	
JVMJBP1	IMS1	0	JBP	Y		N	N	Y	N	N	JAVA
PGM00001	IMS1	0	BMP	Y		N	N	N	N	N	
PGM00002	IMS1	0	IFP	N	E	Y	N	N	N	
PGM00003	IMS1	0	MPP	N		N	N	Y	N	N	PASCAL
PGM00004	IMS1	0	BMP	Y	N	Y	N	N	N	

(scrolled right to screen 2)

PgmName	MbrName	LRgnType	LSchdType	LclStat	LModelName	LModelType
APOL1	IMS1	MPP	SERIAL			
AUTPSB7	IMS1	JBP	SERIAL			
BMP255	IMS1	BMP	PARALLEL			
DBF#FPU0	IMS1	FPU	PARALLEL			
DCSQL6C	IMS1	MPP	SERIAL			
DCSQL7A	IMS1	MPP	SERIAL			
DFSIVP67	IMS1	JBP	SERIAL			
JAVPSB1	IMS1	IFP	SERIAL			
JAVTESTJ	IMS1	JMP	PARALLEL			
JVMJBP1	IMS1	JBP	SERIAL			
PGM00001	IMS1	BMP	PARALLEL	NOTINIT-26-NOPSB	DFSDSPG1	DESC
PGM00002	IMS1	IFP	SERIAL		FPEDESC	DESC
PGM00003	IMS1	MPP	PARALLEL		DFSDSPG1	DESC
PGM00004	IMS1	BMP	SERIAL		BMP011	RSC

(scrolled right to screen 3)

PgmName	MbrName	LRgnType	LTimeCreate	LTimeUpdate	LTimeAccess
APOL1	IMS1	MPP	2011.181 15:22:52.55		
AUTPSB7	IMS1	JBP	2011.181 15:22:52.55		
BMP255	IMS1	BMP	2011.181 15:22:52.55		
DBF#FPU0	IMS1	FPU	2011.181 15:22:52.55		
DCSQL6C	IMS1	MPP	2011.181 15:22:52.55		
DCSQL7A	IMS1	MPP	2011.181 15:22:52.55		
DFSIVP67	IMS1	JBP	2011.181 15:22:52.55		
JAVPSB1	IMS1	IFP	2011.181 15:22:52.55		
JAVTESTJ	IMS1	JMP	2011.181 15:22:52.55		
JVMJBP1	IMS1	JBP	2011.181 15:22:52.55		
PGM00001	IMS1	BMP	2011.181 16:53:07.34		
PGM00002	IMS1	IFP	2011.181 16:53:08.12		
PGM00003	IMS1	MPP	2011.181 16:53:08.86		
PGM00004	IMS1	BMP	2011.181 16:53:09.46		

(scrolled right to screen 4)

PgmName	MbrName	LRgnType	LTimeAccess	LTimeImport	LDefnType
APOL1	IMS1	MPP			MODBLKS
AUTPSB7	IMS1	JBP			MODBLKS
BMP255	IMS1	BMP			MODBLKS
DBF#FPU0	IMS1	FPU			IMS
DCSQL6C	IMS1	MPP			MODBLKS
DCSQL7A	IMS1	MPP			MODBLKS
DFSIVP67	IMS1	JBP			MODBLKS
JAVPSB1	IMS1	IFP			MODBLKS
JAVTESTJ	IMS1	JMP			MODBLKS
JVMJBP1	IMS1	JBP			MODBLKS
PGM00001	IMS1	BMP			CREATE

```
PGM00002 IMS1      IFP                                              CREATE
PGM00003 IMS1      MPP                                              CREATE
PGM00004 IMS1      BMP                                              CREATE
```

OM API input:

```
CMD(QUERY PGM NAME(PGM0000%,APOL1,BMP255,DBF*0,DFSIVP67,AUTPSB7,JAVPSB1,
JAVTESTJ,DCSQL7A,DCSQL6C,JVMJBP1) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.182 00:57:35.598184</statime>
<stotime>2011.182 00:57:35.599380</stotime>
<staseq>C8007C9945E68682</staseq>
<stoseq>C8007C9946314F42</stoseq>
<rqsttkn1>USRT005 10175735</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>PGM            </kwd>
<input>QUERY PGM
 NAME(PGM0000%,APOL1,BMP255,DBF*0,DFSIVP67,AUTPSB7,JAVPSB1,JAVTESTJ,
DCSQL7A,DCSQL6C,JVMJBP1) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PGM" llbl="PgmName" sort="a" key="1" scroll="no" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="MBR" llbl="MbrName" sort="a" key="4" scroll="no" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="CC" llbl="CC" sort="n" key="0" scroll="yes" len="4"
 dtype="INT" align="right" skipb="no" scope="LCL" />
<hdr slbl="CCTXT" llbl="CCText" sort="n" key="0" scroll="yes" len="*"
dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="RGNT" llbl="LRgnType" sort="n" key="0" scroll="no" len="7"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="BMPT" llbl="LBMPType" sort="n" key="0" scroll="yes" len="7"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="FP" llbl="LFP" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="DOPT" llbl="LDOPT" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="GPSB" llbl="LGPSB" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RSDNT" llbl="LDRsdnt" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="LRSDNT" llbl="LRsdnt" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="TLS" llbl="LTranStat" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="LANG" llbl="LPgmLang" sort="n" key="0" scroll="yes" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="SCHD" llbl="LSchdType" sort="n" key="0" scroll="yes" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="LSTT" llbl="LclStat" sort="n" key="0" scroll="yes" len="*"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="MDLN" llbl="LModelName" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="MDLT" llbl="LModelType" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="left" scope="LCL" />
```

```
<hdr slbl="TMCR" llbl="LTimeCreate" sort="n" key="0" scroll="yes"
 len="20" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="TMUP" llbl="LTimeUpdate" sort="n" key="0" scroll="yes"
 len="20" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="TMAC" llbl="LTimeAccess" sort="n" key="0" scroll="yes"
 len="20" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="TMIM" llbl="LTimeImport" sort="n" key="0" scroll="yes"
 len="20" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="DFNT" llbl="LDefnType" sort="n" key="0" scroll="yes" len="8"
 dtype="CHAR" align="left" scope="LCL" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PGM(PGM00001) MBR(IMS1    ) CC(   0) RGNT(BMP) BMPT(Y) FP(N)
 DOPT(N) GPSB(N) LRSDNT(N) TLS(N) SCHD(PARALLEL) LSTT(NOTINIT-26-NOPSB)
 MDLT(DESC) MDLN(DFSDSPG1) TMCR(2011.181 16:53:07.34) DFNT(CREATE)
</rsp>
<rsp>PGM(PGM00002) MBR(IMS1    ) CC(   0) RGNT(IFP) BMPT(N) FP(E)
 DOPT(Y) GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL) MDLT(DESC) MDLN(FPEDESC
 ) TMCR(2011.181 16:53:08.12) DFNT(CREATE) </rsp>
<rsp>PGM(PGM00003) MBR(IMS1    ) CC(   0) RGNT(MPP) BMPT(N) FP(N)
 DOPT(N) GPSB(Y) LRSDNT(N) TLS(N) LANG(PASCAL) SCHD(PARALLEL)
 MDLT(DESC) MDLN(DFSDSPG1) TMCR(2011.181 16:53:08.86) DFNT(CREATE)
</rsp>
<rsp>PGM(PGM00004) MBR(IMS1    ) CC(   0) RGNT(BMP) BMPT(Y) FP(N)
 DOPT(Y) GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL) MDLT(RSC) MDLN(BMP011  )
 TMCR(2011.181 16:53:09.46) DFNT(CREATE) </rsp>
<rsp>PGM(APOL1   ) MBR(IMS1    ) CC(   0) RGNT(MPP) BMPT(N) FP(N)
 DOPT(N) GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL) TMCR(2011.181
 15:22:52.55) DFNT(MODBLKS) </rsp>
<rsp>PGM(BMP255 ) MBR(IMS1    ) CC(   0) RGNT(BMP) BMPT(Y) FP(N)
 DOPT(N) GPSB(N) LRSDNT(N) TLS(N) SCHD(PARALLEL) TMCR(2011.181
 15:22:52.55) DFNT(MODBLKS) </rsp>
<rsp>PGM(DBF#FPU0) MBR(IMS1    ) CC(   0) RGNT(FPU) BMPT(N) FP(E)
 DOPT(Y) GPSB(N) LRSDNT(N) TLS(N) SCHD(PARALLEL) TMCR(2011.181
 15:22:52.55) DFNT(IMS) </rsp>
<rsp>PGM(DFSIVP67) MBR(IMS1    ) CC(   0) RGNT(JBP) BMPT(Y) FP(N)
 DOPT(N) GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL) TMCR(2011.181
 15:22:52.55) DFNT(MODBLKS) </rsp>
<rsp>PGM(AUTPSB7 ) MBR(IMS1    ) CC(   0) RGNT(JBP) BMPT(Y) FP(N)
 DOPT(N) GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL) TMCR(2011.181
 15:22:52.55) DFNT(MODBLKS) </rsp>
<rsp>PGM(JAVPSB1 ) MBR(IMS1    ) CC(   0) RGNT(IFP) BMPT(N) FP(E)
DOPT(N) GPSB(N) LRSDNT(Y) TLS(N) SCHD(SERIAL) TMCR(2011.181
 15:22:52.55) DFNT(MODBLKS) </rsp>
<rsp>PGM(JAVTESTJ) MBR(IMS1    ) CC(   0) RGNT(JMP) BMPT(N) FP(N)
 DOPT(N) GPSB(N) LRSDNT(N) TLS(N) SCHD(PARALLEL) TMCR(2011.181
 15:22:52.55) DFNT(MODBLKS) </rsp>
<rsp>PGM(DCSQL7A ) MBR(IMS1    ) CC(   0) RGNT(MPP) BMPT(N) FP(N)
 DOPT(N) GPSB(Y) LRSDNT(N) TLS(N) LANG(ASM/CBL) SCHD(SERIAL)
 TMCR(2011.181 15:22:52.55) DFNT(MODBLKS) </rsp>
<rsp>PGM(DCSQL6C ) MBR(IMS1    ) CC(   0) RGNT(MPP) BMPT(N) FP(N)
 DOPT(N) GPSB(Y) LRSDNT(N) TLS(N) LANG(PLI) SCHD(SERIAL) TMCR(2011.181
 15:22:52.55) DFNT(MODBLKS) </rsp>
<rsp>PGM(JVMJBP1 ) MBR(IMS1    ) CC(   0) RGNT(JBP) BMPT(Y) FP(N)
 DOPT(N) GPSB(Y) LRSDNT(N) TLS(N) LANG(JAVA) SCHD(SERIAL) TMCR(2011.181
 15:22:52.55) DFNT(MODBLKS) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Lots of different types of programs are displayed. SHOW(ALL) is specified to show all of the possible output fields. All of the program output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The program name, member name that built the line of output, and region type in which the program can run are displayed on every screen. Program names starting with PGM0000 were dynamically created with CREATE PGM commands. DBF#FPU0 is the Fast Path Utility. JAVTESTJ is a Java message processing

program. PGM00001 has no PSB in ACBLIB, so it shows a local status of NOTINIT-26-NOPSB and cannot be scheduled. PGM00003 is a generated PSB GPSB(Y), so it does not need a PSB in ACBLIB. PGM00002 and PGM00004 are dynamic option DOPT(Y) programs, so IMS will not detect that there is no PSB for them in ACBLIB and mark their status as NOTINIT until the first time they are scheduled. A few GPSBs are displayed with different language definitions. The language attribute applies to GPSBs and DOPT PSBs defined as Java programs.

Example 2 for QUERY PGM command

TSO SPOC input:
```
QUERY PGM NAME(APOL1) SHOW(DEFN,DOPT,GPSB,FP)
```

TSO SPOC output:
```
PgmName   MbrName     CC Repo IMSid     LRgnType FP LFP DOPT LDOPT GPSB LGPSB
APOL1     IMS1        0 Y                          N      N         N
APOL1     IMS1        0       IMS1      MPP        N      N         N
APOL1     IMS2        0       IMS2      MPP        N      N         N
APOL1     IMS3        0       IMS3      MPP        N      N         N
```

OM API input:
```
CMD(QUERY PGM NAME(APOL1) SHOW(DEFN,DOPT,GPSB,FP))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.180 22:01:10.263785</statime>
<stotime>2011.180 22:01:10.344614</stotime>
<staseq>C7FF134CD9BE9F55</staseq>
<stoseq>C7FF134CED7A691C</stoseq>
<rqsttkn1>USRT005 10150110</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>PGM            </kwd>
<input>QUERY PGM NAME(APOL1) SHOW(DEFN,DOPT,GPSB,FP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PGM" llbl="PgmName" sort="a" key="1" scroll="no" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="MBR" llbl="MbrName" sort="a" key="4" scroll="no" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="CC" llbl="CC" sort="n" key="0" scroll="yes" len="4"
 dtype="INT" align="right" skipb="no" scope="LCL" />
<hdr slbl="CCTXT" llbl="CCText" sort="n" key="0" scroll="yes" len="*"
 dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="REPO" llbl="Repo" sort="d" key="2" scroll="no" len="1"
dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="IMSID" llbl="IMSid" sort="n" key="0" scroll="yes" len="4"
 dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="RGNT" llbl="LRgnType" sort="n" key="0" scroll="no" len="7"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RFP" llbl="FP" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="GBL" />
<hdr slbl="FP" llbl="LFP" sort="n" key="0" scroll="yes" len="1"
```

```
                              dtype="CHAR" align="left" scope="LCL" />
                             <hdr slbl="RDOPT" llbl="DOPT" sort="n" key="0" scroll="yes" len="1"
                              dtype="CHAR" align="left" scope="GBL" />
                             <hdr slbl="DOPT" llbl="LDOPT" sort="n" key="0" scroll="yes" len="1"
                              dtype="CHAR" align="left" scope="LCL" />
                             <hdr slbl="RGPSB" llbl="GPSB" sort="n" key="0" scroll="yes" len="1"
                              dtype="CHAR" align="left" scope="GBL" />
                             <hdr slbl="GPSB" llbl="LGPSB" sort="n" key="0" scroll="yes" len="1"
                              dtype="CHAR" align="left" scope="LCL" />
                             </cmdrsphdr>
                             <cmdrspdata>
                             <rsp>PGM(APOL1   ) MBR(IMS3   ) CC(   0) RGNT(MPP) IMSID(IMS3    )
                              FP(N) DOPT(N) GPSB(N) </rsp>
                             <rsp>PGM(APOL1   ) MBR(IMS1   ) CC(   0) RGNT(MPP) IMSID(IMS1    )
                              FP(N) DOPT(N) GPSB(N) </rsp>
                             <rsp>PGM(APOL1   ) MBR(IMS1   ) CC(   0) REPO(Y) RFP(N) RDOPT(N)
                              RGPSB(N) </rsp>
                             <rsp>PGM(APOL1   ) MBR(IMS2   ) CC(   0) RGNT(MPP) IMSID(IMS2    )
                              FP(N) DOPT(N) GPSB(N) </rsp>
                             </cmdrspdata>
                             </imsout>
```

Explanation: The stored resource definitions and the runtime resource definitions for the specified resources are returned. Only the FP, DOPT, and GPSB definitions are returned according to the SHOW filters specified.

Example 3 for QUERY PGM command

TSO SPOC input:

```
QUERY PGM NAME(APOL*) SHOW(DEFN)
```

TSO SPOC output:

```
PgmName  MbrName      CC Repo IMSid     LRgnType BMPType LBMPType FP LFP DOPT LDOPT
APOL1    IMS1          0 Y                        N               N   N
APOL1    IMS1          0    IMS1        MPP              N             N    N
APOL1    IMS2          0    IMS2        MPP              N             N    N
APOL1    IMS3          0    IMS3        MPP              N             N    N
(scrolled to the right screen 2)
PgmName  MbrName   Repo LRgnType GPSB LGPSB Rsdnt LRsdnt TranStat LTranStat
APOL1    IMS1      Y             N         N      N
APOL1    IMS1           MPP      N         N      N              N
APOL1    IMS2           MPP      N         N      N              N
APOL1    IMS3           MPP      N         N      N              N
(scrolled to the right screen 3)
PgmName  MbrName   Repo LRgnType PgmLang LPgmLang SchdType LSchdType TimeCreate>
APOL1    IMS1      Y                               SERIAL             2011.180 12:37
APOL1    IMS1           MPP                                 SERIAL
APOL1    IMS2           MPP                                 SERIAL
APOL1    IMS3           MPP                                 SERIAL
(scrolled to the right screen 4)
PgmName  MbrName   Repo LRgnType Create LTimeCreate       TimeUpdate
APOL1    IMS1      Y             :31.44
APOL1    IMS1           MPP             2011.180 12:37:35.41
APOL1    IMS2           MPP             2011.180 12:37:35.53
APOL1    IMS3           MPP             2011.180 12:37:31.44
(scrolled to the right screen 5)
PgmName  MbrName   Repo LRgnType LTimeUpdate       LTimeAccess       LTimeImport
APOL1    IMS1      Y
APOL1    IMS1           MPP
APOL1    IMS2           MPP
APOL1    IMS3           MPP
```

OM API input:

```
CMD(QUERY PGM NAME(APOL*1) SHOW(DEFN))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.180 22:03:33.533960</statime>
<stotime>2011.180 22:03:33.666998</stotime>
<staseq>C7FF13D57BD080D2</staseq>
<stoseq>C7FF13D59C4B6B9A</stoseq>
<rqsttkn1>USRT005 10150333</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>PGM            </kwd>
<input>QUERY PGM NAME(APOL*) SHOW(DEFN) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PGM" llbl="PgmName" sort="a" key="1" scroll="no" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="MBR" llbl="MbrName" sort="a" key="4" scroll="no" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="CC" llbl="CC" sort="n" key="0" scroll="yes" len="4"
 dtype="INT" align="right" skipb="no" scope="LCL" />
<hdr slbl="CCTXT" llbl="CCText" sort="n" key="0" scroll="yes" len="*"
 dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="REPO" llbl="Repo" sort="d" key="2" scroll="no" len="1"
dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="IMSID" llbl="IMSid" sort="n" key="0" scroll="yes" len="4"
 dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="RGNT" llbl="LRgnType" sort="n" key="0" scroll="no" len="7"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RBMPT" llbl="BMPType" sort="n" key="0" scroll="yes" len="7"
 dtype="CHAR" align="left" scope="GBL" />
<hdr slbl="BMPT" llbl="LBMPType" sort="n" key="0" scroll="yes" len="7"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RFP" llbl="FP" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="GBL" />
<hdr slbl="FP" llbl="LFP" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RDOPT" llbl="DOPT" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="GBL" />
<hdr slbl="DOPT" llbl="LDOPT" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RGPSB" llbl="GPSB" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="GBL" />
<hdr slbl="GPSB" llbl="LGPSB" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RRSDNT" llbl="Rsdnt" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" skipb="yes" scope="GBL" />
<hdr slbl="RSDNT" llbl="LDRsdnt" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="LRSDNT" llbl="LRsdnt" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RTLS" llbl="TranStat" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="GBL" />
<hdr slbl="TLS" llbl="LTranStat" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RLANG" llbl="PgmLang" sort="n" key="0" scroll="yes" len="8"
 dtype="CHAR" align="left" scope="GBL" />
<hdr slbl="LANG" llbl="LPgmLang" sort="n" key="0" scroll="yes" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RSCHD" llbl="SchdType" sort="n" key="0" scroll="yes" len="8"
```

```
                dtype="CHAR" align="left" scope="GBL" />
              <hdr slbl="SCHD" llbl="LSchdType" sort="n" key="0" scroll="yes" len="8"
               dtype="CHAR" align="left" scope="LCL" />
              <hdr slbl="RTMCR" llbl="TimeCreate" sort="n" key="0" scroll="yes"
               len="20" dtype="CHAR" align="left" scope="GBL" />
              <hdr slbl="TMCR" llbl="LTimeCreate" sort="n" key="0" scroll="yes"
               len="20" dtype="CHAR" align="left" scope="LCL" />
              <hdr slbl="RTMUP" llbl="TimeUpdate" sort="n" key="0" scroll="yes"
               len="20" dtype="CHAR" align="left" scope="GBL" />
              <hdr slbl="TMUP" llbl="LTimeUpdate" sort="n" key="0" scroll="yes"
              len="20" dtype="CHAR" align="left" scope="LCL" />
              <hdr slbl="TMAC" llbl="LTimeAccess" sort="n" key="0" scroll="yes"
               len="20" dtype="CHAR" align="left" scope="LCL" />
              <hdr slbl="TMIM" llbl="LTimeImport" sort="n" key="0" scroll="yes"
               len="20" dtype="CHAR" align="left" scope="LCL" />
              </cmdrsphdr>
              <cmdrspdata>
              <rsp>PGM(APOL1   ) MBR(IMS3    ) CC(   0) RGNT(MPP) IMSID(IMS3    )
               BMPT(N) FP(N) DOPT(N) GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL)
               TMCR(2011.180 12:37:31.44) </rsp>
              <rsp>PGM(APOL1   ) MBR(IMS1    ) CC(   0) RGNT(MPP) IMSID(IMS1    )
               BMPT(N) FP(N) DOPT(N) GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL)
               TMCR(2011.180 12:37:35.41) </rsp>
              <rsp>PGM(APOL1   ) MBR(IMS1    ) CC(   0) REPO(Y) RBMPT(N) RFP(N)
               RDOPT(N) RGPSB(N) RRSDNT(N) RTLS(N) RSCHD(SERIAL  ) RTMCR(2011.180
               12:37:31.44) </rsp>
              <rsp>PGM(APOL1   ) MBR(IMS2    ) CC(   0) RGNT(MPP) IMSID(IMS2    )
               BMPT(N) FP(N) DOPT(N) GPSB(N) LRSDNT(N) TLS(N) SCHD(SERIAL)
               TMCR(2011.180 12:37:35.53) </rsp>
              </cmdrspdata>
              </imsout>
```

Explanation: A line is returned for each resource that matches the wildcard name.
The resource definitions from each IMS that has the resource defined and the
global repository definition are returned. The repository information is returned by
the command master IMS. There are no IMS specific sections in the repository for
each resource name that matches the wildcard name.

Example 4 for QUERY PGM command

TSO SPOC input:
```
QUERY PGM SHOW(EXPORTNEEDED)
```

TSO SPOC output:
```
PgmName  MbrName    CC LRgnType ExportNeeded
PGM00001 IMS1        0 MPP      Y
PGM00002 IMS1        0 MPP      Y
```

OM API input:
```
CMD(QUERY PGM SHOW(EXPORTNEEDED))
```

Explanation: Two programs, PGM00001 and PGM00002, need to be exported to the
IMSRSC repository.

Related concepts:

How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

/DISPLAY MODIFY command (Commands)

/DISPLAY PGM command (Commands)

QUERY PGMDESC command

Use the QUERY PGMDESC command to query information about program descriptors. A descriptor is a model that can be used to create descriptors or resources.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 534
- "Usage notes" on page 538
- "Output fields" on page 538
- "Return, reason, and completion codes" on page 543
- "Examples" on page 545

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 175. Valid environments for the QUERY PGMDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY PGMDESC	X	X	X
NAME	X	X	X
SHOW	X	X	X
STATUS	X	X	X
DEFAULT	X	X	X

Syntax

```
►►─┬─QUERY─┬─PGMDESC─┬─NAME(*)────────────────────┬───────────►◄
   └─QRY───┘         │              ┌─,─────┐      │
                     └─NAME(─▼─┬─name──┬─)──┘
                               └─name*─┘
```

```
|                                                                                          ◀
                  ┌─────────────,─────────────┐          ┌─DEFAULT(─┬─N─┬─)─┐
   ►──────────────┤                           │──────────┴──────────┴─Y─┴───┴──────────────►◀
          ┌─SHOW(─▼─┬─ALL────────┬─)─┐
          │         ├─BMPTYPE────┤
          │         ├─DEFAULT────┤
          │         ├─DEFN───────┤
          │         ├─DEFNTYPE───┤
          │         ├─DOPT───────┤
          │         ├─FP─────────┤
          │         ├─GLOBAL─────┤
          │         ├─GPSB───────┤
          │         ├─IMSID──────┤
          │         ├─LANG───────┤
          │         ├─LOCAL──────┤
          │         ├─MODEL──────┤
          │         ├─RESIDENT───┤
          │         ├─SCHDTYPE───┤
          │         ├─TIMESTAMP──┤
          │         └─TRANSTAT───┘
          └─SHOW(EXPORTNEEDED)──────────────┘
```
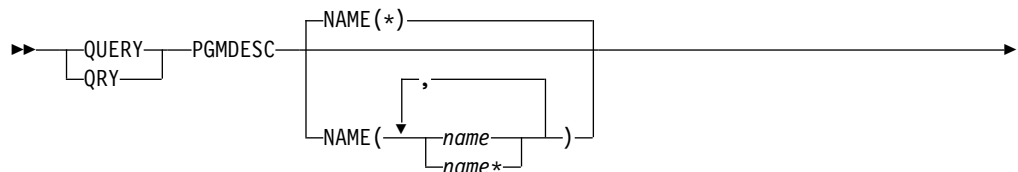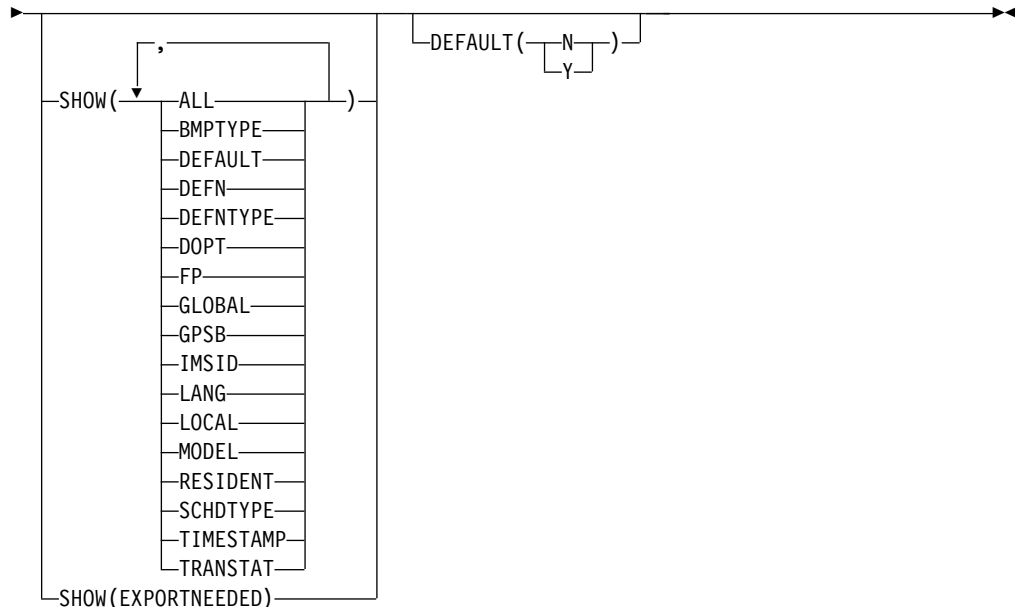
Keywords

The following keywords are valid for the QUERY PGMDESC command:

NAME

Specifies the 1-8 character name of the program descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*) which returns all program descriptors.

DEFAULT()

Specifies which descriptor or descriptors to display.

N   Displays all the program descriptors that are not the default.

Y   Displays the default program descriptor.

SHOW

Specifies the program descriptor output fields to be returned. The program descriptor name is always returned, along with the name of the IMS that created the output, the region type, and the completion code. The filters that are supported with the SHOW keyword are:

ALL

Returns all information about the program descriptor itself.

Restriction: The ExportNeeded column, which is returned when SHOW(EXPORTNEEDED) is specified, is not returned with SHOW(ALL).

BMPTYPE

BMP type option. Specifies whether the program runs in a BMP type region or not. A BMP type region might be a BMP region or a JBP region.

PSBs scheduled by DB2 stored procedures, by programs running under WebSphere Application Server, and by other users of the ODBA interface can be defined with BMPTYPE Y or N.

DEFAULT

Default descriptor option.

DEFN

Specifies that the resource definitions are to be returned.

The program descriptor attributes that can be returned are: BMPTYPE, DOPT, FP, GPSB, LANG, RESIDENT, SCHDTYPE, TRANSTAT, DEFAULT, the repository create and update time stamps, and the IMS runtime create, update, import, and access time stamps.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes, including those defined globally in the repository and those defined locally in the IMS system, are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE or MODEL.
- The LModelName, LModelType, and LDefnType columns, which are returned on the QRY PGMDESC SHOW(ALL) command, are not returned with SHOW(DEFN).
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).

When querying program descriptor information from the repository, resource definitions stored in the repository are used to determine the response lines with the repository information, and the runtime resource definitions are used to determine the response lines with the IMS runtime resource information. The response lines are returned for each stored resource or runtime resource definition that matches the specified filter. If SHOW(DEFN,GLOBAL) is specified, only the stored resource definitions that match the specified filter are returned. If SHOW(DEFN,LOCAL) is specified, only the runtime resource definitions that match the specified filter are returned.

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

DEFNTYPE
Definition type. This is how the descriptor was defined.

DOPT
Dynamic option.

EXPORTNEEDED
Specifies that only the program descriptor resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the program descriptors that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each program descriptor name specified on the command. If the specified program descriptor does not need to be exported to the repository, an N is returned in the ExportNeeded column.

SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords and is valid only when IMS is enabled with the IMSRSC repository.

The QUERY PGMDESC SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources and descriptors that have not been exported to the IMSRSC repository, you can issue the EXPORT DEFN TARGET(REPO) OPTION(CHANGESONLY) command to write the resource and descriptor definitions to the repository so that they are available for the next IMS cold start.

FP Fast Path option.

GLOBAL
Specifies that the stored resource definitions from the repository are to be returned. If SHOW(GLOBAL,DEFN) is specified, the global resource definitions from the repository are returned by the command master IMS. SHOW(GLOBAL,DEFN) is valid only when the command master IMS is enabled to use the repository.

GPSB
Generated PSB option.

IMSID
Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only if the command master IMS is enabled to use the repository.

When SHOW(IMSID) is specified with the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

SHOW(IMSID) cannot be specified with any other SHOW filters other than DEFN and GLOBAL. If SHOW(IMSID,GLOBAL) is specified, GLOBAL is ignored; that is, SHOW(IMSID,GLOBAL) is treated as SHOW(IMSID). SHOW(DEFN,IMSID,LOCAL) is treated as SHOW(DEFN,LOCAL).

LANG
Language interface of the application program for a GPSB.

LOCAL
Specifies that the runtime resource definitions from the IMS system are to be returned.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

MODEL
Model name and model type used to create this descriptor. If the descriptor is created with one or more attributes defined and no model specified, the model name and model type is the default descriptor. The model name and model type are blank for IMS-defined descriptors. The CREATE command specified without the LIKE keyword creates a descriptor using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSPG1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor using a model. The descriptor is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify descriptors that were created with the same model. The model name and model type of a descriptor are exported and imported. The IMPORT command does not use the model name and model type when creating a descriptor.

RESIDENT
Resident option.

SCHDTYPE
Scheduling type, which indicates whether this application program can be scheduled into more than one message region or batch message region simultaneously.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:
- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).

- TH is the tenths and hundredths of a second (00 - 99).

TRANSTAT
> Transaction level statistics option.

Usage notes

This command can be issued only through the Operations Manager API. This command applies to DB/DC, DBCTL and DCCTL systems.

This command is allowed on XRF alternate and RSR tracker systems.

The QUERY PGMDESC command is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

Many of the attributes defined with the program descriptor become meaningful only for programs that are created using the program descriptor. Program descriptors are never scheduled; only programs created using a program descriptor are scheduled.

If you want to display information about resource definitions, specify SHOW(DEFN). If you want to know which IMS systems have the resource defined and also know the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource defined, specify SHOW(IMSID).

Output fields

The following table shows the QUERY PGMDESC output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
> Provides a brief description of the output field.

Table 176. Output fields for the QUERY PGMDESC command

Short label	Long label	Keyword	Scope	Meaning
BMPT	LBMPType	BMPTYPE, DEFN	LCL	BMP type. This value is obtained from IMS.
				N The program does not execute in a BMP type region. It might execute in an IMS TM MPP, JMP, or IFP region, or it might use the ODBA interface or the DRA interface. Use this specification for programs that run in IMS TM MPP, JMP, and IFP regions, or PSBs scheduled by CICS programs using DBCTL and other users of the DRA interface. This is the default.
				Y The program executes in a BMP type region. It might execute in an IMS BMP region or a JBP region. Any associated transactions are assigned normal and limit priority values of zero.
CC	CC	N/A	LCL	Completion code.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DESC	DESCName	PGMDESC	LCL	Program descriptor name.
DFLT	LDflt	DEFAULT	LCL	Default descriptor (Y) or not (N).
				N The descriptor is not the default.
				Y The descriptor is the default. When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a program descriptor called DFSDSPG1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor.
DFNT	LDefnType	DEFNTYPE	LCL	Definition type, which can be one of the following:
				CREATE Defined by a CREATE command. The DEFNTYPE is not changed if the descriptor is updated with an UPDATE command.
				IMPORT Defined by an IMPORT command. The DEFNTYPE is not changed if the descriptor is updated with an UPDATE command.
				IMS Defined by IMS. DFSDSPG1 is an IMS-defined program descriptor containing the default program values.

Table 176. Output fields for the QUERY PGMDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
DOPT	LDOPT	DOPT, DEFN	LCL	Dynamic option (Y) or not (N). This value is obtained from IMS.
				N The PSB associated with this application program is not located dynamically.
				Y The PSB associated with this program is located dynamically. Each time the program associated with this PSB is scheduled, the latest copy of the PSB is loaded. When the program terminates, the PSB is deleted from the PSB pool.
				Until the PSB is required to process a transaction, the ACB for the PSB does not need to be in the ACBLIB or, if the IMS management of ACBs is enabled, in the IMS catalog. A new version of the PSB can be defined and added to the ACBLIB or the IMS catalog at any time. IMS loads the new version of the ACB the next time the PSB is scheduled.
				A DOPT PSB cannot be scheduled if the control blocks for the DBDs that the PSB references are not in the ACBLIB or the IMS catalog.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the program descriptor has been exported to the IMSRSC repository. The value can be Y or N.
FP	LFP	FP, DEFN	LCL	Fast Path exclusive program (E) or not (N). This value is obtained from IMS.
				E The program is a Fast Path-exclusive application program.
				N The program is not a Fast Path application program.

Table 176. Output fields for the QUERY PGMDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
GPSB	LGPSB	GPSB, DEFN	LCL	Generated PSB generated by IMS (Y) or not (N). This value is obtained from IMS. **N** The PSB associated with the program is not generated by IMS. **Y** The PSB and application control block (ACB) that are associated with the program are generated by IMS. The scheduling process of all environments generates a PSB containing an I/O PCB and an alternate modifiable PCB. You do not need to generate the PSB and ACB or submit DDL to define the PSB. The ACB for the PSB is not loaded from either ACBLIB or, if the IMS management of ACBs is enabled, the IMS catalog. Thus, I/O to retrieve the ACB from either the ACBLIB or the IMS catalog is eliminated. The generated PSB contains an I/O PCB named IOPCBbbb and a modifiable, alternate PCB named TPPCB1bb. With an alternate modifiable PCB, an application can use the CHNG call to change the output destination and send output to a destination other than the input destination.
IMSID	IMSid	IMSID	GBL	IMSIDs that have the resource defined from the repository.
LANG	LPgmLang	LANG, DEFN	LCL	Language interface. This value is obtained from IMS. **ASM/CBL** Assembler or COBOL **JAVA** Java (can run only in a Java dependent region). **PASCAL** PASCAL **PLI** PL/I
MBR	MbrName	N/A	LCL	IMSplex member that built the output line.
MDLN	LModelName	MODEL	LCL	Model name. Name of the descriptor used as a model to create this resource. DFSDSPG1 is the IMS descriptor name for programs.
MDLT	LModelType	MODEL	LCL	Model type, either RSC or DESC. RSC means that the descriptor was created using another resource as a model. DESC means that the resource was created using a descriptor as a model.
RBMPT	BmpType	BMPTYPE, DEFN	GBL	BMP type. This value is obtained from the repository.
RDOPT	DOPT	DOPT, DEFN	GBL	Dynamic option (Y) or not (N). This value is obtained from the repository.
RDFLT	Dflt	DEFN	GBL	Default descriptor (Y) or not (N). This value is obtained from the repository.

Table 176. Output fields for the QUERY PGMDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
REPO	Repo	DEFN	GBL	Indicates whether the output line contains stored resource definitions from the repository. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
RFP	FP	FP, DEFN	GBL	Fast Path exclusive program (E) or not (N). This value is obtained from the repository.
RGNT	LRgnType	N/A	LCL	Region type in which program can run. This value is obtained from IMS. Some programs can run in additional region types. For example, a program defined with a program type of MSG can run in a BMP under certain conditions. BMP indicates a batch message processing region. FPU indicates a Fast Path utility region. IFP indicates a Fast Path message processing region. JBP indicates a Java batch message processing region. JMP indicates a Java message processing region. MPP indicates an MPP processing region.
RGPSB	GPSB	GPSB, DEFN	GBL	Generated PSB generated by IMS (Y) or not (N). This value is obtained from the repository.
RLANG	PgmLang	LANG, DEFN	GBL	Language interface. This value is obtained from the repository.
RRGNT	RgnType	N/A	GBL	Region type in which the program can run. This value is obtained from the repository.
RSCHD	SchdType	SCHDTYPE, DEFN	GBL	Schedule type. This value is obtained from the repository.
RSDNT	LDRsdnt	RESIDENT	LCL	Resident option value. For a program created from the descriptor, it indicates if the PSB is to reside in local storage at the next IMS restart. **N** The PSB for a program created from the named program descriptor resource is not made resident in storage. The PSB is loaded at scheduling time. **Y** The PSB for a program created from the named program descriptor resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the PSB and initializes it. A resident program is accessed from local storage, which eliminates I/O to retrieve the ACB.
RTLS	TranStat	TRANSTAT, DEFN	GBL	Transaction level statistics logged (Y) or not (N). This value is obtained from the repository.
RTMCR	TimeCreate	DEFN	GBL	Create time from the repository. This is the time the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN	GBL	Update time from the repository. This is the time the resource was last updated in the repository.

Table 176. Output fields for the QUERY PGMDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
SCHD	LSchdType	SCHDTYPE, DEFN	LCL	Schedule type. This value is obtained from IMS blocks. **PARALLEL** The application program can be scheduled into more than one message region or batch message region simultaneously. **SERIAL** The application program can be scheduled in only one region at a time.
TLS	LTranStat	TRANSTAT, DEFN	LCL	Transaction level statistics logged (Y) or not (N). This value is obtained from IMS blocks. **N** Transaction level statistics logging is not active. **Y** Transaction level statistics logging is active.
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the descriptor was last accessed. This value is obtained from the local IMS. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. For a program descriptor, when the CREATE command or DFSINSX0 exit references the descriptor as a model, the last access time is updated.
TMCR	LTimeCreate	TIMESTAMP	LCL	The time that the descriptor was created. This value is obtained from the local IMS. This is the result of a CREATE PGMDESC command, an IMPORT command that creates the descriptor, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the descriptor was last imported, if applicable. The import time is retained across warm start and emergency restart. This value is obtained from the local IMS.
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated as a result of the UPDATE PGMDESC command or the IMPORT command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 177. Return and reason codes for the QUERY PGMDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000004'	X'00001010'	No resources were found to be returned. The resource names specified might be invalid, or: • There were no resources that match the filter specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'00000008'	X'00002040'	Invalid filter or filter combination. An invalid filter might be an invalid parameter specified with the DEFAULT keyword.
X'0000000C'	X'00003000'	Command was successful for some descriptors but failed for others. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'0000000C'	X'00003004'	Command was successful for none of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'00004004'	No CQS address space.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	No resource structure exists, or resource structure is not available.
X'00000010'	X'00004100'	Resource structure is full.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Repository access is denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server is being shut down.

Table 177. Return and reason codes for the QUERY PGMDESC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'0000450B'	Repository Server is not available.
X'00000010'	X'0000450C'	Repository Server is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields related to the IMSRSC repository.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'0000501C'	IMODULE GETMAIN error.
X'00000014'	X'00005100'	RM request error.
X'00000014'	X'00005104'	CQS error.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY PGMDESC command is rejected because the resource list is unusable. Contact IBM Software Support. Send in the output from the command in error, and a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the IMSRSC repository RID and RMD data sets.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 178. Completion codes for the QUERY PGMDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for program descriptor.
10	NO RESOURCES FOUND	Program descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.

Examples

The following are examples of the QUERY PGMDESC command:

Example 1 for QUERY PGMDESC command

TSO SPOC input:
```
QRY PGMDESC SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
DescName MbrName    CC LRgnType LBMPType LFP LDOPT LGPSB LDRsdnt LTranStat
DFSDSPG1 IMS1        0 MPP      N        N   N     N     N       N
DOPTDESC IMS1        0 BMP      Y        N   Y     N     N       N
FPEDESC  IMS1        0 IFP      N        E   Y     N     N       N
GPSBDESC IMS1        0 BMP      Y        N   N     Y     N       N
```

```
JAVADESC IMS1      0 JMP       N      N   Y    N     N     N
```

(scrolled to the right screen 2)

DescName	MbrName	LRgnType	LPgmLang	LSchdType	LModelName	LModelType	LDflt
DFSDSPG1	IMS1	MPP		PARALLEL			Y
DOPTDESC	IMS1	BMP		SERIAL	DFSDSPG1	DESC	N
FPEDESC	IMS1	IFP		SERIAL	DFSDSPG1	DESC	N
GPSBDESC	IMS1	BMP	ASM/CBL	SERIAL	DFSDSPG1	DESC	N
JAVADESC	IMS1	JMP	JAVA	SERIAL	DFSDSPG1	DESC	N

(scrolled to the right screen 3)

DescName	MbrName	LRgnType	LTimeCreate	LTimeUpdate
DFSDSPG1	IMS1	MPP	2011.181 15:22:52.55	
DOPTDESC	IMS1	BMP	2011.181 16:53:04.61	
FPEDESC	IMS1	IFP	2011.181 16:53:05.23	
GPSBDESC	IMS1	BMP	2011.181 16:53:06.09	
JAVADESC	IMS1	JMP	2011.181 16:53:06.75	

(scrolled to the right screen 4)

DescName	MbrName	LRgnType	LTimeAccess	LTimeImport	LDefnType
DFSDSPG1	IMS1	MPP	2011.181 16:53:08.86		IMS
DOPTDESC	IMS1	BMP			CREATE
FPEDESC	IMS1	IFP	2011.181 16:53:08.12		CREATE
GPSBDESC	IMS1	BMP			CREATE
JAVADESC	IMS1	JMP			CREATE

OM API input:

```
CMD(QUERY PGMDESC SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.182 00:32:39.456322</statime>
<stotime>2011.182 00:32:39.457450</stotime>
<staseq>C800770670E4215E</staseq>
<stoseq>C8007706712AA69E</stoseq>
<rqsttkn1>USRT005 10173239</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>PGMDESC         </kwd>
<input>QRY PGMDESC SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" sort="a" key="1" scroll="no" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="MBR" llbl="MbrName" sort="a" key="4" scroll="no" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="CC" llbl="CC" sort="n" key="0" scroll="yes" len="4"
 dtype="INT" align="right" skipb="no" scope="LCL" />
<hdr slbl="CCTXT" llbl="CCText" sort="n" key="0" scroll="yes" len="*"
 dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="RGNT" llbl="LRgnType" sort="n" key="0" scroll="no" len="7"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="BMPT" llbl="LBMPType" sort="n" key="0" scroll="yes" len="7"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="FP" llbl="LFP" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="DOPT" llbl="LDOPT" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
```

```
<hdr slbl="GPSB" llbl="LGPSB" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="RSDNT" llbl="LDRsdnt" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="TLS" llbl="LTranStat" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="LANG" llbl="LPgmLang" sort="n" key="0" scroll="yes" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="SCHD" llbl="LSchdType" sort="n" key="0" scroll="yes" len="8"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="MDLN" llbl="LModelName" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="MDLT" llbl="LModelType" sort="n" key="0" scroll="yes"
 len="4" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="DFLT" llbl="LDflt" sort="n" key="0" scroll="yes" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="TMCR" llbl="LTimeCreate" sort="n" key="0" scroll="yes"
 len="20" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="TMUP" llbl="LTimeUpdate" sort="n" key="0" scroll="yes"
 len="20" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="TMAC" llbl="LTimeAccess" sort="n" key="0" scroll="yes"
 len="20" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="TMIM" llbl="LTimeImport" sort="n" key="0" scroll="yes"
 len="20" dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="DFNT" llbl="LDefnType" sort="n" key="0" scroll="yes" len="8"
 dtype="CHAR" align="left" scope="LCL" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DOPTDESC) MBR(IMS1   ) CC(   0) RGNT(BMP) BMPT(Y) FP(N)
 DOPT(Y) GPSB(N) RSDNT(N) TLS(N) SCHD(SERIAL) MDLT(DESC)
 MDLN(DFSDSPG1) DFLT(N) TMCR(2011.181 16:53:04.61) DFNT(CREATE) </rsp>
<rsp>DESC(DFSDSPG1) MBR(IMS1   ) CC(   0) RGNT(MPP) BMPT(N) FP(N)
 DOPT(N) GPSB(N) RSDNT(N) TLS(N) SCHD(PARALLEL) DFLT(Y) TMCR(2011.181
 15:22:52.55) TMAC(2011.181 16:53:08.86) DFNT(IMS) </rsp>
<rsp>DESC(JAVADESC) MBR(IMS1   ) CC(   0) RGNT(JMP) BMPT(N) FP(N)
 DOPT(Y) GPSB(N) RSDNT(N) TLS(N) LANG(JAVA) SCHD(SERIAL) MDLT(DESC)
 MDLN(DFSDSPG1) DFLT(N) TMCR(2011.181 16:53:06.75) DFNT(CREATE) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1   ) CC(   0) RGNT(IFP) BMPT(N) FP(E)
 DOPT(Y) GPSB(N) RSDNT(N) TLS(N) SCHD(SERIAL) MDLT(DESC)
 MDLN(DFSDSPG1) DFLT(N) TMCR(2011.181 16:53:05.23) TMAC(2011.181
 16:53:08.12) DFNT(CREATE) </rsp>
<rsp>DESC(GPSBDESC) MBR(IMS1   ) CC(   0) RGNT(BMP) BMPT(Y) FP(N)
 DOPT(N) GPSB(Y) RSDNT(N) TLS(N) LANG(ASM/CBL) SCHD(SERIAL)
 MDLT(DESC) MDLN(DFSDSPG1) DFLT(N) TMCR(2011.181 16:53:06.09)
 DFNT(CREATE) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: All program descriptors are returned with all output fields. All of the program descriptor output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The program descriptor name, the member name that built the line of output, and the region type in which the program can run are displayed on every screen. The fields that are blank are not applicable to the specified program descriptor.

Example 2 for QUERY PGMDESC command

TSO SPOC input:

```
QUERY PGMDESC NAME(*) SHOW(DEFN,DOPT,GPSB,FP)
```

TSO SPOC output:

```
DescName MbrName   CC Repo IMSid    LRgnType FP LFP DOPT LDOPT GPSB LGPSB
DFSDSPG1 IMS1       0      IMS1     MPP         N        N          N
DFSDSPG1 IMS2       0      IMS2     MPP         N        N          N
```

```
| DFSDSPG1 IMS3      0      IMS3    MPP        N        N           N
| DOPTDESC IMS1      0 Y                       N        Y        N
| DOPTDESC IMS1      0      IMS1    BMP        N        Y           N
| DOPTDESC IMS2      0      IMS2    BMP        N        Y           N
| DOPTDESC IMS3      0      IMS3    BMP        N        Y           N
| FPEDESC  IMS1      0 Y                       E        Y        N
| FPEDESC  IMS1      0      IMS1    IFP        E        Y           N
| FPEDESC  IMS2      0      IMS2    IFP        E        Y           N
| FPEDESC  IMS3      0      IMS3    IFP        E        Y           N
| GPSBDESC IMS1      0 Y                       N        N        Y
| GPSBDESC IMS1      0      IMS1    BMP        N        N           Y
| GPSBDESC IMS2      0      IMS2    BMP        N        N           Y
| GPSBDESC IMS3      0      IMS3    BMP        N        N           Y
```

| OM API input:

| CMD(QUERY PGMDESC NAME(*) SHOW(DEFN,DOPT,GPSB,FP))

| OM API output:

```
| <imsout>
| <ctl>
| <omname>OM1OM    </omname>
| <omvsn>1.5.0</omvsn>
| <xmlvsn>20   </xmlvsn>
| <statime>2011.182 00:43:45.113457</statime>
| <stotime>2011.182 00:43:45.201311</stotime>
| <staseq>C800798142D7121A</staseq>
| <stoseq>C80079815849F99C</stoseq>
| <rqsttkn1>USRT005 10174345</rqsttkn1>
| <rc>00000000</rc>
| <rsn>00000000</rsn>
| </ctl>
| <cmd>
| <master>IMS1    </master>
| <userid>USRT005 </userid>
| <verb>QRY </verb>
| <kwd>PGMDESC         </kwd>
| <input>QUERY PGMDESC NAME(*) SHOW(DEFN,DOPT,GPSB,FP) </input>
| </cmd>
| <cmdrsphdr>
| <hdr slbl="DESC" llbl="DescName" sort="a" key="1" scroll="no" len="8"
|  dtype="CHAR" align="left" scope="LCL" />
| <hdr slbl="MBR" llbl="MbrName" sort="a" key="4" scroll="no" len="8"
|  dtype="CHAR" align="left" scope="LCL" />
| <hdr slbl="CC" llbl="CC" sort="n" key="0" scroll="yes" len="4"
|  dtype="INT" align="right" skipb="no" scope="LCL" />
| <hdr slbl="CCTXT" llbl="CCText" sort="n" key="0" scroll="yes" len="*"
|  dtype="CHAR" align="left" skipb="yes" scope="LCL" />
| <hdr slbl="REPO" llbl="Repo" sort="d" key="2" scroll="no" len="1"
| dtype="CHAR" align="left" skipb="yes" scope="LCL" />
| <hdr slbl="IMSID" llbl="IMSid" sort="n" key="0" scroll="yes" len="4"
|  dtype="CHAR" align="left" skipb="yes" scope="LCL" />
| <hdr slbl="RGNT" llbl="LRgnType" sort="n" key="0" scroll="no" len="7"
|  dtype="CHAR" align="left" scope="LCL" />
| <hdr slbl="RFP" llbl="FP" sort="n" key="0" scroll="yes" len="1"
|  dtype="CHAR" align="left" scope="GBL" />
| <hdr slbl="FP" llbl="LFP" sort="n" key="0" scroll="yes" len="1"
|  dtype="CHAR" align="left" scope="LCL" />
| <hdr slbl="RDOPT" llbl="DOPT" sort="n" key="0" scroll="yes" len="1"
|  dtype="CHAR" align="left" scope="GBL" />
| <hdr slbl="DOPT" llbl="LDOPT" sort="n" key="0" scroll="yes" len="1"
|  dtype="CHAR" align="left" scope="LCL" />
| <hdr slbl="RGPSB" llbl="GPSB" sort="n" key="0" scroll="yes" len="1"
|  dtype="CHAR" align="left" scope="GBL" />
| <hdr slbl="GPSB" llbl="LGPSB" sort="n" key="0" scroll="yes" len="1"
|  dtype="CHAR" align="left" scope="LCL" />
| </cmdrsphdr>
```

```
<cmdrspdata>
<rsp>DESC(DOPTDESC) MBR(IMS1    ) CC(   0) RGNT(BMP) IMSID(IMS1    )
FP(N) DOPT(Y) GPSB(N) </rsp>
<rsp>DESC(DFSDSPG1) MBR(IMS1    ) CC(   0) RGNT(MPP) IMSID(IMS1    )
FP(N) DOPT(N) GPSB(N) </rsp>
<rsp>DESC(JAVADESC) MBR(IMS1    ) CC(   0) RGNT(JMP) IMSID(IMS1    )
FP(N) DOPT(Y) GPSB(N) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1    ) CC(   0) RGNT(IFP) IMSID(IMS1    )
FP(E) DOPT(Y) GPSB(N) </rsp>
<rsp>DESC(GPSBDESC) MBR(IMS1    ) CC(   0) RGNT(BMP) IMSID(IMS1    )
FP(N) DOPT(N) GPSB(Y) </rsp>
<rsp>DESC(DOPTDESC) MBR(IMS1    ) CC(   0) REPO(Y) RFP(N) RDOPT(Y)
RGPSB(N) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1    ) CC(   0) REPO(Y) RFP(E) RDOPT(Y)
RGPSB(N) </rsp>
<rsp>DESC(GPSBDESC) MBR(IMS1    ) CC(   0) REPO(Y) RFP(N) RDOPT(N)
RGPSB(Y) </rsp>
<rsp>DESC(JAVADESC) MBR(IMS1    ) CC(   0) REPO(Y) RFP(N) RDOPT(Y)
RGPSB(N) </rsp>
<rsp>DESC(DOPTDESC) MBR(IMS3    ) CC(   0) RGNT(BMP) IMSID(IMS3    )
FP(N) DOPT(Y) GPSB(N) </rsp>
<rsp>DESC(DFSDSPG1) MBR(IMS3    ) CC(   0) RGNT(MPP) IMSID(IMS3    )
FP(N) DOPT(N) GPSB(N) </rsp>
<rsp>DESC(JAVADESC) MBR(IMS3    ) CC(   0) RGNT(JMP) IMSID(IMS3    )
FP(N) DOPT(Y) GPSB(N) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS3    ) CC(   0) RGNT(IFP) IMSID(IMS3    )
FP(E) DOPT(Y) GPSB(N) </rsp>
<rsp>DESC(GPSBDESC) MBR(IMS3    ) CC(   0) RGNT(BMP) IMSID(IMS3    )
FP(N) DOPT(N) GPSB(Y) </rsp>
<rsp>DESC(DOPTDESC) MBR(IMS2    ) CC(   0) RGNT(BMP) IMSID(IMS2    )
FP(N) DOPT(Y) GPSB(N) </rsp>
<rsp>DESC(DFSDSPG1) MBR(IMS2    ) CC(   0) RGNT(MPP) IMSID(IMS2    )
FP(N) DOPT(N) GPSB(N) </rsp>
<rsp>DESC(JAVADESC) MBR(IMS2    ) CC(   0) RGNT(JMP) IMSID(IMS2    )
FP(N) DOPT(Y) GPSB(N) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS2    ) CC(   0) RGNT(IFP) IMSID(IMS2    )
FP(E) DOPT(Y) GPSB(N) </rsp>
<rsp>DESC(GPSBDESC) MBR(IMS2    ) CC(   0) RGNT(BMP) IMSID(IMS2    )
FP(N) DOPT(N) GPSB(Y) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The stored resource definitions and the runtime resource definitions for the specified resources are returned. Only the FP, DOPT, and GPSB definitions are returned because the SHOW options are specified. The Dflt (default) column, which identifies the default descriptor, is returned because the local IMS runtime definitions are returned. DFSDSPG1 is the default descriptor and is only at each IMS system. The IMS generated system descriptor definitions are not in the repository.

Example 3 for QUERY PGMDESC command

TSO SPOC input:
```
QUERY PGMDESC NAME(PGMD0001,PGMD0002) SHOW(EXPORTNEEDED)
```

TSO SPOC output:
```
DescName MbrName    CC LRgnType ExportNeeded
PGMD0001 IMS1        0 MPP      N
PGMD0002 IMS1        0 MPP      Y
```

OM API input:
```
CMD(QUERY PGMDESC SHOW(EXPORTNEEDED))
```

Explanation: Program descriptor PGMD0001 has been exported to the IMSRSC
repository, but program descriptor PGMD0002 has not been exported to the
IMSRSC repository.

Related concepts:

How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY POOL command

Use the QUERY POOL command to display information about the current usage of
the buffers that are managed by the Fast Path 64-bit buffer manager, processor
storage utilization statistics for the pool type specified, and full-function database
(OSAM or VSAM) buffer pools.

If you modify buffer pool definitions in the DFSDFxxx member and run the
UPDATE POOL TYPE(DBAS) command, and an emergency restart occurs, you can
issue a QUERY POOL TYPE(DBAS) command to determine whether the updates
were successful.

Subsections:
* "Environment"
* "Syntax"
* "Keywords" on page 551
* "Usage notes" on page 553
* "Output fields" on page 553
* "Return, reason, and completion codes" on page 560
* "Examples" on page 561

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 179. Valid environments for the QUERY POOL command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY POOL	X	X	X
TYPE	X	X	X
SHOW	X	X	X
POOLID	X	X	
SIZE	X	X	
SUBTYPE	X	X	

Restriction: The dynamic database buffer pools function is not supported in a DBCTL warm
standby environment.

Syntax

```
         ┌─QUERY─┐                 ┌─────────<,──────────┐
►►───────┤       ├──POOL──┬─TYPE(ACBIN64)────────────────────────────────────────►◄
         └─QRY───┘        │              ┌─SHOW(─▼──ALL────────┬─)─┐
                          │                     ├─FREE─────────┤
                          │                     ├─INUSE────────┤
                          │                     ├─OVERFLOW─────┤
                          │                     ├─SIZE─────────┤
                          │                     └─STATISTICS───┘
                          ├─TYPE(DBAS)─┬──┤ AA ├──────────────────
                          ├─TYPE(FPBP64)─┘
                          │              ┌─SHOW(─┬─ALL────────┬─)─┐
                          │                     └─STATISTICS──┘
                          └─TYPE(GURIN64)────────────────────────────────
                                        ┌────────<,──────────┐
                                        └─SHOW(─▼──ALL────────┬─)─┐
                                                ├─FREE─────────┤
                                                ├─INUSE────────┤
                                                ├─OVERFLOW─────┤
                                                ├─SIZE─────────┤
                                                └─STATISTICS───┘
```

AA:

```
├──┬─────────────┬──┬─────────────┬────────────────────────────────────►
   └─POOLID(id)──┘  └─SIZE(size)──┘      ┌──────<,──────┐
                                  └─SUBTYPE(─▼──┬─OSAM─┬──)─┘
                                               └─VSAM─┘


      ┌─SHOW(ALL)──────────────┐
►─────┤                        ├─────────────────────────────────────────┤
      └─SHOW(─┬─MEMBER──────┬─)─┘
              └─STATISTICS──┘
```
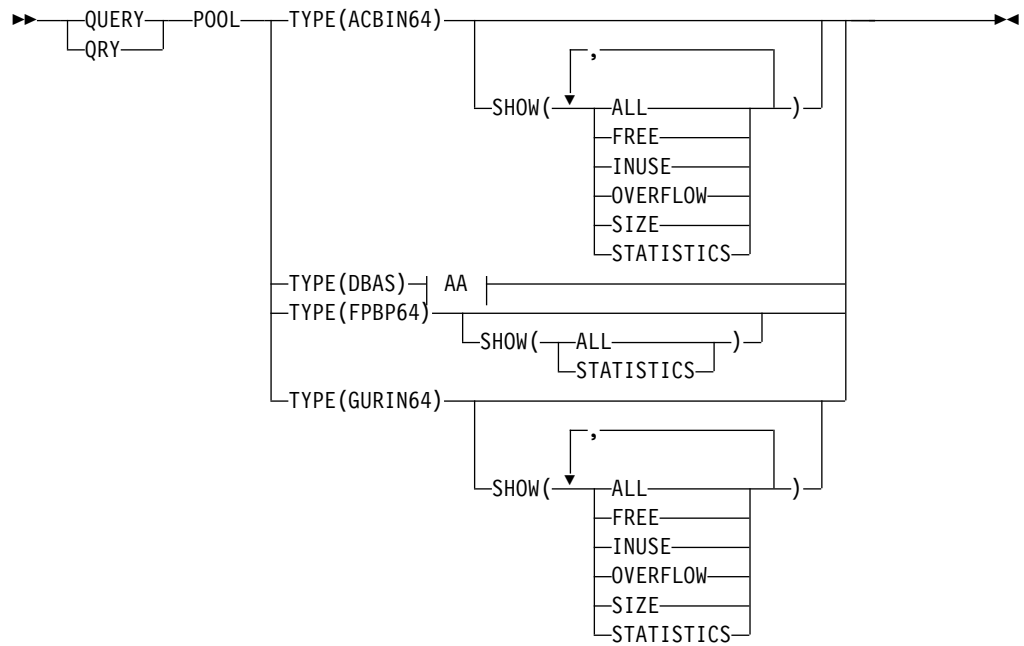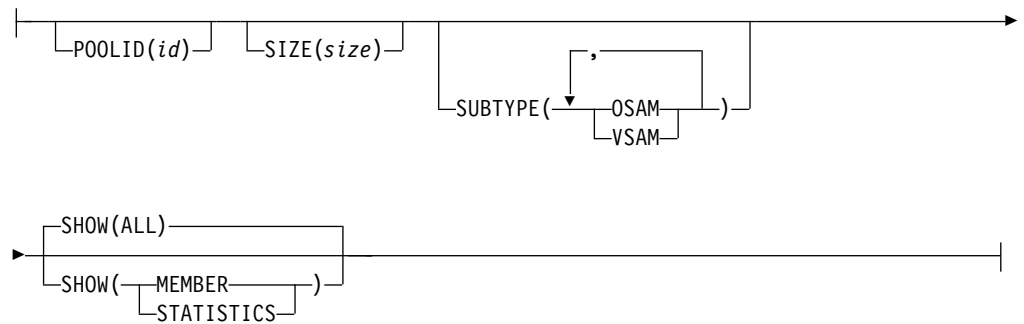
Keywords

The following keywords are valid for the QUERY POOL command:

TYPE()

Required keyword that specifies the type of pool that you want the information to display for. The pool types are mutually exclusive. The command is rejected with a syntax error if multiple types are specified.

FPBP64

Returns information about the Fast Path 64-bit buffer pool.

ACBIN64

Returns information about the 64-bit cache pools. The 64-bit ACB storage pool is ACBIN64.

GURIN64

Returns information about the 64-bit cache pools. The 64-bit cache for XML documents created as responses to GUR calls is GURIN64.

DBAS

Returns information about full-function database (OSAM or VSAM) buffer pools.

POOLID()

Specifies the user-defined identifier assigned to a specific OSAM subpool or VSAM shared resource pool. The identifier is a 1- to 4-character alphanumeric field that filters which subpools are displayed in the QUERY command output. This keyword is optional.

SIZE()

Specifies the size of the buffers in the subpool to be filtered on. The size value acts filters which subpools are displayed in the QUERY command output. The size value can be from 512 to 32768 bytes. The command rounds up the size value to 512, 1024, 2048, and thereafter to multiples of 2048 bytes. You can code specifications of 1024 and greater as 1KB, 2KB, 4KB, and thereafter round up values to multiples of 2KB to a maximum of 32KB.

SUBTYPE()

Specifies the subpool types that are returned to the issuer of the command. The following subpool types can be specified with this keyword.

OSAM

Specifies that OSAM subpool information is displayed.

VSAM

Specifies that VSAM subpool information is displayed.

OSAM,VSAM

Specifies that OSAM subpool information and VSAM subpool information is displayed.

SHOW()

Specifies the pool output fields to be returned. There is no default.

ALL

Returns all information for the pool type that you specified.

FREE

Returns the percentage of the pool that was not reserved for buffers or control data. The SHOW(FREE) keyword is valid only with the TYPE(ACBIN64) keyword.

INUSE

Returns the percentage of the pool that is used. The SHOW(INUSE) keyword is valid only with the TYPE(ACBIN64) keyword.

OVERFLOW

Returns the total number of overflow buffers in use in the pool. Overflow buffers are used to store the members that are too large to store in standard 64-bit cache buffers. The SHOW(OVERFLOW) keyword is valid only with the TYPE(ACBIN64) keyword.

SIZE

Returns the size of the 64-bit storage pool as specified by the user. The SHOW(SIZE) keyword is valid only with the TYPE(ACBIN64) keyword.

MEMBER

Lists the subpools and shows the active members of the IMS PROCLIB data set where full-function database buffer pool definitions are obtained.

STATISTICS

Returns specific point-in-time statistics information for the specified pool type.

If TYPE(DBAS) is specified, this option is equivalent to the information that is displayed when the /DIS POOL DBAS type-1 command is issued.

Usage notes

The QUERY POOL command can be specified only through the OM API.

Output fields

The following table shows the QUERY POOL output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 180. Output fields for the QUERY POOL command

Short label	Long label	Keyword	Meaning
ACTBUF	ACTBUF	TYPE(ACBIN64) TYPE(GURIN64)	The total number of active buffers in the 64-bit storage pool, that is, buffers that contain valid ACBLIB data in binary format or whole catalog DBD or PSB records in XML format.
ADDS	Isrt	TYPE(ACBIN64) TYPE(GURIN64)	For TYPE(ACBIN64), the number of times IMS inserted an ACB member into the 64-bit storage pool. For TYPE(GURIN64), the number of times IMS inserted a catalog record into the 64-bit storage pool. This count is initialized at IMS startup.
ALTREQ	AltReq _____ BfAlt	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of buffer alter calls for this subpool. This count includes NEW BLOCK and BYTALT calls. For VSAM subpools, it is the number of logical records altered.
B64BT	64b_Buf	TYPE(FPBP64) SHOW(ALL)	The 64 bit storage used for the buffers for each subpool and extent.
B64BTOT	64b_Tot	TYPE(FPBP64)	The total amount of 64-bit storage used for each subpool and extents, and the overall 64-bit storage total.
BFSRCH	BfSrch _____ VRds	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of buffers searched by all LOCATE-type calls for this subpool. For VSAM subpools, it is the number of VSAM control interval reads.

Table 180. Output fields for the QUERY POOL command (continued)

Short label	Long label	Keyword	Meaning
BFSTLW	BfStlW VWts	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of single block writes initiated by buffer steal routine for this subpool. For VSAM subpools, it is the number of VSAM control interval writes.
BUFSIZE	Buf_Size	TYPE(FPBP64) SHOW(STATISTICS)	The buffer size being displayed.
BUFAVL	Buf_Avl	TYPE(FPBP64)	The number of buffers available for use from this subpool or extents, or the total for the entire buffer pool or the entire system. The system total is found in row DBF_TOTB.
BUFS	BufSize	TYPE(DBAS)	The buffer size.
BUFUSE	Buf_Use	TYPE(FPBP64)	The number of buffers being used by a process (IMS, dependent region, or external thread (such as ODBM)) from this subpool or extents, or the grand total for the entire buffer pool or the entire system. The system total is found in row DBF_TOTB.
BUFA	Buf_A	TYPE(FPBP64) SHOW(ALL)	The number of available buffers in the subpool base section or extent. This column does not contain the total number of available buffers for the entire subpool.
BUFQ	Buf_Q	TYPE(FPBP64) SHOW(ALL)	The number of quiesced buffers in the subpool base section or extent. This column does not contain the total number of quiesced buffers for the entire subpool.
BUFT	Buf_T	TYPE(FPBP64) SHOW(ALL)	The total number of buffers in the subpool base section or extent. This column does not contain the total number of buffers for the entire subpool.
BUFU	Buf_U	TYPE(FPBP64) SHOW(ALL)	The number of used buffers in the subpool base section or extent. This column does not contain the total number of used buffers for the entire subpool.
CACHEOPT	CacheOption	TYPE(DBAS)	The subpool cache option that is specified on the IOBF statement of either the DFSVSM*xx* member or the OSAM section of the DFSDF*xxx* member. This field is displayed for OSAM buffers when the SHOW(ALL) or SHOW(STATISTICS) option is used. The subpool cache option has one of the following values: **ALL**　Cache all data. **CHG**　Cache only changed data. **NO**　No data caching.
CC	CC	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.
DATECR	TimeCreate	TYPE(FPBP64) SHOW(ALL)	The date this subpool or extent was created.

Table 180. Output fields for the QUERY POOL command (continued)

Short label	Long label	Keyword	Meaning
DELETES	Del	TYPE(ACBIN64) TYPE(GURIN64)	For TYPE(ACBIN64), the number of times IMS deleted an ACB member from the 64-bit storage pool. For TYPE(GURIN64), the number of times IMS deleted a catalog record from the 64-bit storage pool. This count is initialized at IMS startup.
ECSA	ECSA_Buf	TYPE(FPBP64) SHOW(ALL)	The ECSA used for the buffers for each subpool and for each extent.
ECSAT	ECSA_Tot	TYPE(FPBP64)	The total amount of ECSA used for each subpool and extents, and the overall ECSA total.
ECSAB	ECSA_B	TYPE(FPBP64) SHOW(ALL)	The amount of ECSA used for buffers by each subpool base section or extent. This column does not contain the total amount of ESCA for the entire subpool.
ECSAO	ECSA_O	TYPE(FPBP64) SHOW(ALL)	The amount of ECSA used for control blocks by each subpool base section or extent. This column does not contain the total amount of ESCA for the entire subpool.
EPVTT	EPVT_T	TYPE(FPBP64) SHOW(ALL)	The amount of EPVT used by subpool base section or extent. This column does not contain the total amount of EPVT for the entire subpool.
EPVTTOT	EPVT_Tot	TYPE(FPBP64)	The total amount of EPVT used for each pool, subpool, and extents. EPVT is used only for control blocks, except for FDBR, which can use EPVT for Fast Path database buffers and control blocks.
ERRORS	NumErrors	TYPE(DBAS) SHOW(STATISTICS)	For OSAM, it is the total number of I/O errors for this subpool, or number of buffers locked in the pool due to write errors for this subpool. For VSAM, it is the number of permanent write errors now in the subpool, or the largest number of errors in this execution.
EXTPER	%Ext	TYPE(FPBP64) SHOW(ALL)	Percentage of the base section to be used before an extent is taken. This value might change over time based on buffer usage.
FINDS	Gets	TYPE(ACBIN64) TYPE(GURIN64)	For TYPE(ACBIN64), the number of times IMS attempted to retrieve a member from the 64-bit storage pool. For TYPE(GURIN64), the number of times IMS attempted to retrieve a catalog record from the 64-bit storage pool. This count is initialized at IMS startup.
FIXOPT	FixOpt	TYPE(DBAS) SHOW(STATISTICS)	Buffer and pool fix options. For OSAM, Y/N indicates whether the DATA BUFFER PREFIX /DATA BUFFERS are fixed. For VSAM, Y/N indicates whether the INDEX BUFFERS / DATA BUFFER PREFIX / DATA BUFFERS are fixed.
FREE	Free	TYPE(ACBIN64) TYPE(GURIN64)	The percentage of the pool that has not been reserved for buffers or control data.

Table 180. Output fields for the QUERY POOL command (continued)

Short label	Long label	Keyword	Meaning
FNDIPL	FndIpl / SyncPt	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of LOCATE-type calls for this subpool where data is already in the OSAM pool. For VSAM subpools, it is the number of system checkpoint (synchronization point) requests.
HITS	Hits	TYPE(ACBIN64) TYPE(GURIN64)	The number of times a GET request from the 64-bit pool was successful. That number is displayed as a percentage.
HWM	HWM	TYPE(FPBP64) SHOW(STATISTICS)	The High Water Mark of buffers used.
ID	PoolId	N/A	The OSAM subpool or VSAM shared pool ID.
INUSE	Used	TYPE(ACBIN64) TYPE(GURIN64)	The percentage of the pool that is used.
LCTREQ	LctReq / RRba	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of LOCATE-type calls for this subpool. For VSAM subpools, it is the number of retrieval requests by RBA.
LGBFNM	Lmbr	TYPE(ACBIN64) TYPE(GURIN64)	The name of the largest member in the 64-bit storage pool.
LGBFTP	Ltype	TYPE(ACBIN64) TYPE(GURIN64)	The resource type of the largest member in the 64-bit storage pool.
LGBFSZ	Lsize	TYPE(ACBIN64) TYPE(GURIN64)	The size in kilobytes (KB) of the largest member specified.
MBR	MbrName	N/A	The IMS identifier of the IMS for which the database information is displayed. The IMS identifier is always returned.
MBRS	Mbrs	TYPE(ACBIN64) TYPE(GURIN64)	The total number of members in the 64-bit storage pool.
MISSES	Miss		The number of times a GET request from the 64-bit pool was unsuccessful. That number is displayed as a percentage.
NBUF	NBuf	TYPE(DBAS)	The number of buffers.
NEWBLK	NewBlk / RKey	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of blocks created in the pool. For VSAM subpools, it is the number of retrieval requests by KEY.
OVERFLOW	Overflow	TYPE(ACBIN64) TYPE(GURIN64)	The total number of overflow buffers in use in the pool. Overflow buffers are used to store the members that are too large to store in standard 64-bit cache buffers.
PMBR	ProcMbr	SHOW(MEMBER)	The name of the member of the IMS PROCLIB data set.

Table 180. Output fields for the QUERY POOL command (continued)

Short label	Long label	Keyword	Meaning

POOL	Subpool	TYPE(FPBP64) SHOW(ALL)	The IMS-generated subpool name, where *xxxx* is a numeric value:
			DBFC*xxxx*
			    A common subpool used for DEDB data. The buffers reside in 64-bit addressable storage. This subpool name is only valid for TYPE(FPBP64).
			DBFX*xxxx*
			    A system subpool used for all other buffer requests, including IMS internal buffers. The buffers reside in ECSA. This subpool name is only valid for TYPE(FPBP64).

		TYPE(ACBIN64)
		TYPE(GURIN64)
			The subpool type.
			ACBIN64
			    A 64-bit ACB storage pool.
			GURIN64
			    A 64-bit cache for XML documents that are created as responses to GUR calls.

		TYPE(DBAS)	The subpool type.
			OSAM An OSAM subpool.
			VSAM-D
			    A VSAM data subpool.
			VSAM-I
			    A VSAM index subpool.

PURGRQ	PurgRq NRec	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of PURGE calls for this subpool. For VSAM subpools, it is the number of new VSAM logical records created.
PURGWR	PurgWr HSR-S	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of buffers written by purge. For VSAM subpools, it is the number of successful VSAM reads from Hiperspace™ buffers.
QUIBUF	Qui_Buf	TYPE(FPBP64) SHOW(ALL)	The total number of buffers being quiesced for the subpool, or the total number of quiesced buffers in the system. The system total is found in row DBF_TOTB.
RDREQ	RdReq Found	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of READ I/O requests for this subpool. For VSAM subpools, it is the number of control intervals that VSAM found in the subpool through lookaside.
SECT	Section	TYPE(DBAS) SHOW(MEMBER)	Section of the DFSDFxxx or DFSVSMxx member processed.
SIZE	Size	TYPE(FPBP64) SHOW(ALL)	The size of the 64-bit buffer or 64-bit storage pool as specified by the user.
SMBFNM	Smbr	TYPE(ACBIN64) TYPE(GURIN64)	The name of the smallest member in the 64-bit storage pool.
SMBFTP	Stype		The resource type of the smallest member in the 64-bit storage pool.
SMBFSZ	Ssize		The size in kilobytes (K) of the smallest member specified.

Table 180. Output fields for the QUERY POOL command (continued)

Short label	Long label	Keyword	Meaning
SPT	SPT	TYPE(FPBP64) SHOW(STATISTICS)	This field is returned only for SHOW(STATISTICS). For SHOW(ALL), see the TYPE field.Pool type, which can be one of the following: **C** Common (64-bit) buffers. **S** System (ESCA) buffers
STATUS	Status	TYPE(FPBP64) SHOW(ALL)	The status of the subpool or extent. If the subpool or extent is in use, this field is blank. If all subpools and extents are in use, this column is not displayed. The possible status descriptions are: **Comp** This subpool is being compressed. This status indicates that one or more extents of the subpool is no longer needed and is being retired from service. **Dlet** This subpool or extent is being deleted. The associated buffers are not counted in the buffers totals, but the associated storage is counted in the storage totals. **Qsc** This subpool extent is being quiesced as part of a subpool compression or deletion. **QscW** This subpool or extent is waiting for a buffer to be returned before completing a quiesce operation.
TOTBUF	Tot_Buf	TYPE(FPBP64)	The total number of buffers in this subpool, including the base section and the extents, or the total number of buffers in the entire system. The system total is in row DBF_TOTB.
TOTBUF	TOTBUF	TYPE(ACBIN64) TYPE(GURIN64)	The total number of buffers in the 64-bit storage pool, including those that do not currently contain ACB member or catalog member data.
TYPE	Type	TYPE(FPBP64) SHOW(ALL)	This field is returned only for SHOW(ALL). For SHOW(STATISTICS), see the SPT field. Pool type, which can be one of the following: **G** This row contains overall totals for the entire buffer pool. **Tot** The total values for the subpool and extents, with the name of the subpool in the SUBPOOL column. **Base** This is the base section of the subpool, and does not include the extent values. **Ext** This is an extent for the subpool, and does not include the base section of the subpool.

Table 180. Output fields for the QUERY POOL command (continued)

Short label	Long label	Keyword	Meaning
TYPEID	T_id	TYPE(FPBP64) SHOW(ALL)	A combined numeric identifier that gives both the type of subpool being described in the row and its status: **10** Total values for a subpool. **15** The base section of a subpool that is in use. **20** A subpool extent that is in use. **25** A subpool extent that is being quiesced. **30** A subpool extent that is waiting for buffers to be returned. **35** A subpool extent that is being deleted. **50** The base section of a subpool that is being deleted. **55** The base section of a subpool that is being deleted. **60** The base section of a subpool that is being deleted and is waiting for buffers to be returned. **65** A subpool extent that is being quiesced, which is part of a subpool that is being deleted. **70** A subpool extent that is part of a subpool that is being deleted, and is currently waiting for one or more buffers to be returned. **75** A subpool extent that is being deleted. This extent is part of a subpool that is being deleted.
USE	%Use	TYPE(FPBP64)	Percentage of the buffers that are currently in use by a process for a subpool or extents.
WBSYID	WBsyId HSW-S	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of LOCATE calls for this subpool that waited due to a busy ID. For VSAM subpools, it is the number of successful VSAM writes to Hiperspace buffers.
WBSYRD	WBsyRd HS-R-F	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of LOCATE-type calls for this subpool that waited due to a busy buffer reading. For VSAM subpools, it is the number of failed VSAM reads from Hiperspace buffers. This indicates the number of times a VSAM READ request from Hiperspace resulted in DASD I/O.
WBSYWR	WBsyWr HSNBf	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of LOCATE-type calls for this subpool that waited due to a busy writing. For VSAM subpools, it is the number of Hiperspace buffers defined for this subpool.
WNOBFR	WNoBfr	TYPE(DBAS) SHOW(STATISTICS)	Number of buffer steal requests for this subpool that waited because no buffers were available to be stolen. This field is only applicable to OSAM subpools.

Table 180. Output fields for the QUERY POOL command (continued)

Short label	Long label	Keyword	Meaning
WRLSEO	WRlseO HS-W-F	TYPE(DBAS) SHOW(STATISTICS)	For OSAM subpools, it is the number of buffer steal or purge requests for this subpool that waited for ownership to be released. For VSAM subpools, it is the number of failed VSAM writes to Hiperspace buffers. This indicates the number of times a VSAM WRITE request to Hiperspace resulted in DASD I/O.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY POOL command. The OM return and reason codes that might be returned as a result of the QUERY POOL command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 181. Return and reason codes for the QUERY POOL command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See Table 182 on page 561 for details.
X'00000004'	X00001010	The command found that no resources exist that match the specified filter.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'00004014'	The command failed because it is not valid on the RSR tracker.
X'00000010'	X'00004016'	The command failed because it is not valid in a DCCTL environment.
X'00000010'	X'00004024'	Fast Path is not enabled for this IMS system, so there is no data to display.
X'00000010'	X'00004404'	The Fast Path 64-bit buffer manager is not enabled, so there is no data to display.
X'00000014'	X'00005000'	IMODULE GETSTOR error.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005010'	Unable to obtain latch.
X'00000014'	X'00005FFF'	The QUERY POOL command could not be processed because an internal error occurred.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 182. Completion codes for the QUERY POOL command

Completion code	Completion code text	Meaning
0	X'00000000'	The QUERY POOL command completed successfully.
10	NO RESOURCES FOUND	Either the specified pool does not exist, or the specified wildcard name does not match any existing pool names.

Completion codes are not issued as part of this QUERY POOL command. The expected output will show available pool or subpool information.

Examples

The following are examples of the QUERY POOL command:

Example 1 for QUERY POOL command

TSO SPOC input:

```
QUERY POOL TYPE(FPBP64) SHOW(ALL)
```

TSO SPOC output:

```
 Response for: QUERY POOL TYPE(FPBP64) SHOW(ALL)                  More:   +>
 Subpool  MbrName   CC    Size Type Status T_id Tot_Buf Buf_T Buf_Use Buf_U Buf_Avl Buf_A %Use %Ext Qui_Buf Buf_Q
 DBF_MAXB SYS3
 DBF_TOTB SYS3                 G              744             7             737                      136
 DBFC0001 SYS3          512 Tot          10  608             0             608                        0
 DBFC0001 SYS3     0         Base        15          32              0             32    0   300            0
 DBFC0001 SYS3     0         Ext         20          96              0             96    0                  0
 DBFC0001 SYS3     0         Ext         20          96              0             96    0                  0
 DBFC0001 SYS3     0         Ext         20          96              0             96    0                  0
 DBFC0001 SYS3     0         Ext         20          64              0             64    0                  0
 DBFC0001 SYS3     0         Ext         20          64              0             64    0                  0
 DBFC0001 SYS3     0         Ext         20          64              0             64    0                  0
 DBFC0001 SYS3     0         Ext         20          32              0             32    0                  0
 DBFC0001 SYS3     0         Ext         20          32              0             32    0                  0
 DBFC0001 SYS3     0         Ext         20          32              0             32    0                  0
 DBFC0003 SYS3         2048 Tot          10   16             0              16                        0
 DBFC0003 SYS3     0         Base        15          16              0             16    0   100            0
 DBFC0005 SYS3         4096 Tot          10   64             0              64                        0
 DBFC0005 SYS3     0         Base        15          16              0             16    0   100            0
 DBFC0005 SYS3     0         Ext         20          16              0             16    0                  0
 DBFC0005 SYS3     0         Ext         20          16              0             16    0                  0
 DBFC0005 SYS3     0         Ext         20          16              0             16    0                  0
 DBFC0006 SYS3         1024 Tot          10   16             0              16                        0
 DBFC0006 SYS3     0         Base        15          16              0             16    0   100            0
 DBFS0001 SYS3          512 Tot          10                  0               0                       32
 DBFS0001 SYS3     0         Base  Qsc   55           0              0              0  N/A    50           32
 DBFS0003 SYS3         2048 Tot          10    8             0               8                        0
 DBFS0003 SYS3     0         Base        15           8              0              8    0   100            0
 DBFS0004 SYS3         4096 Tot          10    8             0               8                        0
 DBFS0004 SYS3     0         Base        15           8              0              8    0   100            0
 DBFS0005 SYS3         1024 Tot          10    8             0               8                        0
 DBFS0005 SYS3     0         Base        15           8              0              8    0   100            0
 DBFS0006 SYS3          512 Tot          10   16             7               9                        0
 DBFS0006 SYS3     0         Base        15          16              7              9   43   100            0

 (Scrolled right to screen 2)
```

```
Subpool    Size Type  EPVT_Tot EPVT_T ECSA_Tot ECSA_Buf ECSA_B ECSA_Oth ECSA_O 64b_Tot 64b_Buf TimeCreate
DBF_MAXB                                                                                2048M
DBF_TOTB        G      4K              510K     108K            397K            804K
DBFC0001   512  Tot    2K                                       266K            304K
DBFC0001        Base           1K                                      18K              16K 2010.148 15:05:04.95
DBFC0001        Ext            156                                     42K              48K 2010.148 15:25:02.58
DBFC0001        Ext            156                                     42K              48K 2010.148 15:25:02.51
DBFC0001        Ext            156                                     42K              48K 2010.148 15:25:02.41
DBFC0001        Ext            156                                     28K              32K 2010.148 15:25:02.36
DBFC0001        Ext            156                                     28K              32K 2010.148 15:24:55.49
DBFC0001        Ext            156                                     28K              32K 2010.148 15:17:10.70
DBFC0001        Ext            156                                     14K              16K 2010.148 15:17:10.29
DBFC0001        Ext            156                                     14K              16K 2010.148 15:17:09.72
DBFC0001        Ext            156                                     14K              16K 2010.148 15:17:09.16
DBFC0003   2048 Tot    156                                      7K              32K
DBFC0003        Base           156                                     7K               32K 2010.148 15:05:04.95
DBFC0005   4096 Tot    936                                      28K             256K
DBFC0005        Base           468                                     8K               64K 2010.148 15:09:05.45
DBFC0005        Ext            156                                     7K               64K 2010.148 15:24:43.27
DBFC0005        Ext            156                                     7K               64K 2010.148 15:24:43.24
DBFC0005        Ext            156                                     7K               64K 2010.148 15:24:43.29
DBFC0006   1024 Tot    0                                        7K              16K
DBFC0006        Base           0                                      7K                16K 2010.148 15:09:05.46
DBFS0001   512  Tot    156                       30K           14K
DBFS0001        Base           156                     30K           14K              2010.148 15:05:04.95
DBFS0003   2048 Tot    156                       20K           3K
DBFS0003        Base           156                     19K           4K               2010.148 15:05:04.95
DBFS0004   4096 Tot    156                       36K           3K
DBFS0004        Base           156                     35K           4K               2010.148 15:09:05.46
DBFS0005   1024 Tot    0                         12K           3K
DBFS0005        Base           0                      11K           4K               2010.148 15:09:05.46
DBFS0006   512  Tot    0                         15K           7K
DBFS0006        Base           0                      15K           7K               2010.148 15:09:05.46
```

OM API input:

```
CMD (QUERY POOL TYPE(FPBP64) SHOW(ALL))
```

OM API output:

```
<cmdrsphdr>
<hdr slbl="POOL" llbl="Subpool" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="right" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
 len="4" dtype="CHAR" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="SIZE" llbl="Size" scope="LCL" sort="n" key="0" scroll="no"
 len="6" dtype="CHAR" align="right" />
<hdr slbl="TYPE" llbl="Type" scope="LCL" sort="a" key="3" scroll="no"
 len="1" dtype="CHAR" align="center" />
<hdr slbl="TOTBUF" llbl="Tot_Buf" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="USE" llbl="Buf_Use" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="ABUF" llbl="Buf_Avl" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="PUSE" llbl="%Use" scope="LCL" sort="n" key="0" scroll="yes"
 len="3" dtype="CHAR" align="right" />
<hdr slbl="EXTPER" llbl="%Ext" scope="LCL" sort="n" key="yes"
 scroll="yes" len="3" dtype="CHAR" align="right" />
<hdr slbl="ECSAT" llbl="ECSA_Tot" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="ECSA" llbl="ECSA_Buf" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="ECSAO" llbl="ECSA_Oth" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="CHAR" align="right" />
<hdr slbl="B64BT" llbl="64b_Tot" scope="LCL" sort="n" key="0"
 scroll="n" len="5" dtype="CHAR" align="right" />
```

```
<hdr slbl="B64B" llbl="64b_Buf" scope="LCL" sort="n" key="0" scroll="n"
 len="5" dtype="CHAR" align="right" />
<hdr slbl="TMCR" llbl="TimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left"  skipb="yes"  />
</cmdrsphdr>
<cmdrspdata>
<rsp>POOL(DBF_MAXB)  MBR(SYS3   )              B64BT(  100M)</rsp>
<rsp>POOL(DBF_TOTB)  MBR(SYS3   )              TYPE(G) TOTBUF( 219) USE(
    0) ABUF( 219) ECSAT(62680) ECSA(  92K) ECSAO(   4K) B64B( 280K)
 B64BT( 280K)</rsp>
<rsp>POOL(DBFS0004)  MBR(SYS3   )              SIZE( 4096) TYPE(A)
 TOTBUF(   8) USE(   ) ABUF(   8) ECSAT(  32K)</rsp>
<rsp>POOL(DBFS0004)  MBR(SYS3   ) CC(  0)  TYPE(B) TOTBUF(   8) USE(
    0) ABUF(   8) PUSE(  0) EXTPER(100) ECSA(  32K) ECSAO(   3K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFC0004)  MBR(SYS3   )              SIZE( 4096) TYPE(A)
 TOTBUF(  47) USE(   ) ABUF(  47) ECSAT(   0) B64BT( 188K)</rsp>
<rsp>POOL(DBFC0004)  MBR(SYS3   ) CC(  0)  TYPE(B) TOTBUF(  47) USE(
    0) ABUF(  47) PUSE(  0) EXTPER( 34) B64B( 188K) ECSAO(  20K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFS0003)  MBR(SYS3   )              SIZE( 2048) TYPE(A)
 TOTBUF(   8) USE(   ) ABUF(   8) ECSAT(  16K)</rsp>
<rsp>POOL(DBFS0003)  MBR(SYS3   ) CC(  0)  TYPE(B) TOTBUF(   8) USE(
    0) ABUF(   8) PUSE(  0) EXTPER(100) ECSA(  16K) ECSAO(   3K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFC0003)  MBR(SYS3   )              SIZE( 2048) TYPE(A)
 TOTBUF(  16) USE(   ) ABUF(  16) ECSAT(   0) B64BT(  32K)</rsp>
<rsp>POOL(DBFC0003)  MBR(SYS3   ) CC(  0)  TYPE(B) TOTBUF(  16) USE(
    0) ABUF(  16) PUSE(  0) EXTPER(100) B64B(  32K) ECSAO(   7K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFS0002)  MBR(SYS3   )              SIZE( 1024) TYPE(A)
 TOTBUF(  28) USE(   ) ABUF(  28) ECSAT(  28K)</rsp>
<rsp>POOL(DBFS0002)  MBR(SYS3   ) CC(  0)  TYPE(B) TOTBUF(  28) USE(
    0) ABUF(  28) PUSE(  0) EXTPER( 28) ECSA(  28K) ECSAO(  12K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFC0002)  MBR(SYS3   )              SIZE( 1024) TYPE(A)
 TOTBUF(  40) USE(   ) ABUF(  40) ECSAT(   0) B64BT(  40K)</rsp>
<rsp>POOL(DBFC0002)  MBR(SYS3   ) CC(  0)  TYPE(B) TOTBUF(  40) USE(
    0) ABUF(  40) PUSE(  0) EXTPER( 40) B64B(  40K) ECSAO(  17K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFS0001)  MBR(SYS3   )              SIZE(  512) TYPE(A)
 TOTBUF(  32) USE(   ) ABUF(  32) ECSAT(  16K)</rsp>
<rsp>POOL(DBFS0001)  MBR(SYS3   ) CC(  0)  TYPE(B) TOTBUF(  32) USE(
    0) ABUF(  32) PUSE(  0) EXTPER( 50) ECSA(  16K) ECSAO(  14K)
 TMCR(2010.088 14:26:14.83)</rsp>
<rsp>POOL(DBFC0001)  MBR(SYS3   )              SIZE(  512) TYPE(A)
 TOTBUF(  40) USE(   ) ABUF(  40) ECSAT(   0) B64BT(  20K)</rsp>
<rsp>POOL(DBFC0001)  MBR(SYS3   ) CC(  0)  TYPE(B) TOTBUF(  40) USE(
    0) ABUF(  40) PUSE(  0) EXTPER( 80) B64B(  20K) ECSAO(  17K)
 TMCR(2010.088 14:26:14.83)</rsp>
</cmdrspdata>
```

Explanation: In this example, information about the Fast Path 64-bit DEDB buffer
pool is displayed. Each subpool is shown on a different output line.

Example 2 for QUERY POOL command

TSO SPOC input:
```
QRY POOL TYPE(ACBIN64)
```

TSO SPOC output:
```
PoolName Type      MbrName    CC
ACBIN64  Cache64   IMS1        0
```

OM API input:

```
CMD (QRY POOL TYPE(ACBIN64))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.090 17:04:30.525184</statime>
<stotime>2010.090 17:04:30.525588</stotime>
<staseq>C5C2C060F8700392</staseq>
<stoseq>C5C2C060F8894DD2</stoseq>
<rqsttkn1>USRT011 10100430</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>POOL           </kwd>
<input>QRY POOL TYPE(ACBIN64) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="POOLNM" llbl="PoolName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TYPE" llbl="Type" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBRNAME" llbl="MbrName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="right" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>POOLNM(ACBIN64 ) TYPE(Cache64 ) MBRNAME(SYS3     ) CC(    0)
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: In this example, information about the 64-bit cache pools is displayed.

Example 3 for QUERY POOL command

TSO SPOC input:

```
QRY POOL TYPE(ACBIN64) SHOW(ALL)
```

TSO SPOC output:

```
PoolName Type     CC Size  Mbrs  Used  Free  Overflow   Gets   Hit   Miss
ACBIN64  Cache64  0   2    3700   25    75          5  10000  9603   397

  Isrt    Del  Lmbr     Ltype  Lsize   Smbr   Stype  Ssize
   300     20  PAYROLL  PSB     2000   DEBIT  INT      100
```

OM API input:

```
CMD (QRY POOL TYPE(ACBIN64) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
```

<omvsn>1.5.0</omvsn>
<xmlvsn>20 </xmlvsn>
<statime>2010.090 18:17:16.097203</statime>
<stotime>2010.090 18:17:16.097784</stotime>
<staseq>C5C2D0A44DEB3E15</staseq>
<stoseq>C5C2D0A44E0F8CD5</stoseq>
<rqsttkn1>USRT011 10111716</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3   </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>POOL   </kwd>
<input>QRY POOL TYPE(ACBIN64) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="POOLNM" llbl="PoolName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TYPE" llbl="Type" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBRNAME" llbl="MbrName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="POOLSZ" llbl="Size" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="MBRS" llbl="Mbrs" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="INUSE" llbl="Used" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="FREE" llbl="Free" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="OVERFLOW" llbl="Overflow" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="FINDS" llbl="Gets" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="HITS" llbl="Hit" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="MISSES" llbl="Miss" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="ADDS" llbl="Isrt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="DELETES" llbl="Del" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="LGBFNM" llbl="Lmbr" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LGBFTP" llbl="Ltype" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LGBFSZ" llbl="Lsize" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="SMBFNM" llbl="Smbr" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SMBFTP" llbl="Stype" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SMBFSZ" llbl="Ssize" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>POOLNM(ACBIN64) TYPE(Cache64) MBRNAME(SYS3   ) CC(0)
 POOLSZ(5120) MBRS(20) INUSE(0) FREE(100)
 OVERFLOW(0) FINDS(0) HITS(0) MISSES(0)
 ADDS(14) DELETES(0) LGBFNM(BMP255) LGBFTP(JCB   )

<omvsn>1.5.0</omvsn>
<xmlvsn>20 </xmlvsn>
<statime>2010.090 18:17:16.097203</statime>
<stotime>2010.090 18:17:16.097784</stotime>
<staseq>C5C2D0A44DEB3E15</staseq>
<stoseq>C5C2D0A44E0F8CD5</stoseq>
<rqsttkn1>USRT011 10111716</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3   </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>POOL   </kwd>
<input>QRY POOL TYPE(ACBIN64) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="POOLNM" llbl="PoolName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="TYPE" llbl="Type" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBRNAME" llbl="MbrName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="POOLSZ" llbl="Size" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="MBRS" llbl="Mbrs" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="INUSE" llbl="Used" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="FREE" llbl="Free" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="OVERFLOW" llbl="Overflow" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="FINDS" llbl="Gets" scope="LCL" sort="a" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="HITS" llbl="Hit" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="MISSES" llbl="Miss" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="ADDS" llbl="Isrt" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="DELETES" llbl="Del" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="LGBFNM" llbl="Lmbr" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LGBFTP" llbl="Ltype" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="LGBFSZ" llbl="Lsize" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
<hdr slbl="SMBFNM" llbl="Smbr" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SMBFTP" llbl="Stype" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="SMBFSZ" llbl="Ssize" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="right" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>POOLNM(ACBIN64) TYPE(Cache64) MBRNAME(SYS3   ) CC(0)
 POOLSZ(5120) MBRS(20) INUSE(0) FREE(100)
 OVERFLOW(0) FINDS(0) HITS(0) MISSES(0)
 ADDS(14) DELETES(0) LGBFNM(BMP255) LGBFTP(JCB   )

```
          LGBFSZ(   45056) SMBFNM(DX41SK01) SMBFTP(DMB     ) SMBFSZ(      512)
    </rsp>
    </cmdrspdata>
    </imsout>
```

Explanation: In this example, detailed information about the 64-bit cache pools is displayed.

Example 4 for QUERY POOL command

TSO SPOC input:

```
QUERY POOL TYPE(FPBP64) SHOW(STATISTICS)
```

TSO SPOC output:

```
Response for: QUERY POOL TYPE(FPBP64) SHOW(STATISTICS)                    More:    >
Buf_Size MbrName    CC SPT Tot_Buf Buf_Use Buf_Avl %Use   HWM EPVT_Tot ECSA_Tot 64b_Tot
Total    SYS3                224       0     224    0             2K     269K    344K
   512   SYS3       0 C       32       0      32    0     0     157      15K     16K
  1024   SYS3       0 C       40       0      40    0     0     156      18K     40K
  2048   SYS3       0 C       16       0      16    0     0     157       8K     32K
  4096   SYS3       0 C       32       0      32    0     0     157      15K    128K
  8192   SYS3       0 C       16       0      16    0     0     157       8K    128K
   512   SYS3       0 S       32       0      32    0     0     157      31K       0
  1024   SYS3       0 S       32       0      32    0     0     157      47K       0
  2048   SYS3       0 S        8       0       8    0     0     156      20K       0
  4096   SYS3       0 S        8       0       8    0     0     156      36K       0
  8192   SYS3       0 S        8       0       8    0     0     156      68K       0
```

OM API input:

```
CMD (QUERY POOL TYPE(FPBP64) SHOW(STATISTICS))
```

OM API output:

```
<hdr slbl="BSIZE" llbl="Buf_Size" sort="a" key="3" scroll="no" len="5"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="MBR" llbl="MbrName " sort="a" key="1" scroll="no" len="8"
 dtype="CHAR" align="right" scope="LCL" />
<hdr slbl="CC" llbl="CC" sort="n" key="0" scroll="no " len="4"
 dtype="CHA" align="right" skipb="no" scope="LCL" />
<hdr slbl="CCTXT" llbl="CCText" sort="n" key="0" scroll="yes" len="*"
 dtype="CHAR" align="left" skipb="yes" scope="LCL" />
<hdr slbl="SPT" llbl="SPT" sort="a" key="2" scroll="no" len="1"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="TOTBUF" llbl="Tot_Buf" sort="n" key="0" scroll="yes" len="5"
 dtype="CHAR" align="right" scope="LCL" />
<hdr slbl="USE" llbl="Buf_Use" sort="n" key="0" scroll="yes" len="5"
 dtype="CHAR" align="right" scope="LCL" />
<hdr slbl="ABUF" llbl="Buf_Avl" sort="n" key="0" scroll="yes" len="5"
 dtype="CHAR" align="right" scope="LCL" />
<hdr slbl="PUSE" llbl="%Use" sort="n" key="0" scroll="yes" len="3"
 dtype="CHAR" align="right" scope="LCL" />
<hdr slbl="HWM" llbl="  HWM" sort="n" key="0" scroll="yes" len="5"
 dtype="CHAR" align="left" scope="LCL" />
<hdr slbl="EPVTTOT" llbl="EPVT_Tot" sort="n" key="0" scroll="yes"
 len="5" dtype="CHAR" align="right" scope="LCL" />
<hdr slbl="ECSAT" llbl="ECSA_Tot" sort="n" key="0" scroll="yes" len="5"
 dtype="CHAR" align="right" scope="LCL" />
<hdr slbl="B64BTOT" llbl="64b_Tot" sort="n" key="0" scroll="yes"
 len="5" dtype="CHAR" align="right" scope="LCL" />
</cmdrsphdr>
<cmdrspdata>
<rsp>                  MBR(SYS3    )              BSIZE(Total) TOTBUF(  224)
 USE(   0) HWM(    ) ABUF(  224) PUSE(  0) ECSAT(  269K) EPVTTOT(
 2K) B64BTOT(  344K)</rsp>
```

```
<rsp>              MBR(SYS3    ) CC(   0) SPT(S) BSIZE(  512)
  TOTBUF(   32) USE(    0) ABUF(   32) PUSE( 0) HWM(    0) ECSAT(
31K) EPVTTOT( 157) B64BTOT(    0)</rsp>
<rsp>              MBR(SYS3    ) CC(   0) SPT(S) BSIZE( 1024)
  TOTBUF(   32) USE(    0) ABUF(   32) PUSE( 0) HWM(    0) ECSAT(
47K) EPVTTOT( 157) B64BTOT(    0)</rsp>
<rsp>              MBR(SYS3    ) CC(   0) SPT(S) BSIZE( 2048)
  TOTBUF(    8) USE(    0) ABUF(    8) PUSE( 0) HWM(    0) ECSAT(
20K) EPVTTOT( 156) B64BTOT(    0)</rsp>
<rsp>              MBR(SYS3    ) CC(   0) SPT(S) BSIZE( 4096)
  TOTBUF(    8) USE(    0) ABUF(    8) PUSE( 0) HWM(    0) ECSAT(
36K) EPVTTOT( 156) B64BTOT(    0)</rsp>
<rsp>              MBR(SYS3    ) CC(   0) SPT(S) BSIZE( 8192)
  TOTBUF(    8) USE(    0) ABUF(    8) PUSE( 0) HWM(    0) ECSAT(
68K) EPVTTOT( 156) B64BTOT(    0)</rsp>
<rsp>              MBR(SYS3    ) CC(   0) SPT(C) BSIZE(  512)
  TOTBUF(   32) USE(    0) ABUF(   32) PUSE( 0) HWM(    0) ECSAT(
15K) EPVTTOT( 157) B64BTOT(  16K)</rsp>
<rsp>              MBR(SYS3    ) CC(   0) SPT(C) BSIZE( 1024)
  TOTBUF(   40) USE(    0) ABUF(   40) PUSE( 0) HWM(    0) ECSAT(
18K) EPVTTOT( 156) B64BTOT(  40K)</rsp>
<rsp>              MBR(SYS3    ) CC(   0) SPT(C) BSIZE( 2048)
  TOTBUF(   16) USE(    0) ABUF(   16) PUSE( 0) HWM(    0) ECSAT(
8K) EPVTTOT( 157) B64BTOT(  32K)</rsp>
<rsp>              MBR(SYS3    ) CC(   0) SPT(C) BSIZE( 4096)
  TOTBUF(   32) USE(    0) ABUF(   32) PUSE( 0) HWM(    0) ECSAT(
15K) EPVTTOT( 157) B64BTOT( 128K)</rsp>
<rsp>              MBR(SYS3    ) CC(   0) SPT(C) BSIZE( 8192)
  TOTBUF(   16) USE(    0) ABUF(   16) PUSE( 0) HWM(    0) ECSAT(
8K) EPVTTOT( 157) B64BTOT( 128K)</rsp>
</cmdrspdata>
```

Example 5 for QUERY POOL command

TSO SPOC input:

```
QUERY POOL TYPE(DBAS) SUBTYPE(OSAM) SHOW(STATISTICS)
```

TSO SPOC output:

```
Subpool  MbrName CC BufSize PoolId   NBuf  FixOpt LctReq/RRba NewBlk/RKey
-------- ------- -- ------- ----     ----- ------ ----------- -----------
OSAM     IMS1    0     512            100 N/N              0           0
OSAM     IMS1    0    1024 OSM1      1000 N/N              0           0
OSAM     IMS1    0    2048 OSM2      7832 Y/N              0           0
OSAM     IMS1    0    4096 OSM3     32767 N/Y              0           0
OSAM     IMS1    0    6144 ABC1      8000 N/N              0           0
OSAM     IMS1    0    8192 ABC2      8000 N/N              0           0
OSAM     IMS1    0   10240 ABC3      8000 N/N              0           0
OSAM     IMS1    0   12288 ABC4      8000 N/N              0           0
OSAM     IMS1    0   14336 ABC5      8000 N/N              0           0
OSAM     IMS1    0   16384 ABC6      8000 N/N              0           0
OSAM     IMS1    0   32768 OSM9         4 N/N              0           0
OSAM     IMS1    0   32768 OSM4        10 N/N              0           0

(scrolled right to screen 2)
AltReq/BfAlt PurgRq/NRec FndIpl/SyncPt BfSrch/VRds RdReq/Found BfStlW/VWts
------------ ----------- ------------- ----------- ----------- -----------
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
```

```
                      0            0            0            0            0            0
                      0            0            0            0            0            0
                      0            0            0            0            0            0
```

(scrolled right to screen 3)
```
PurgWr/HSR-S WBsyId/HSW-S WBsyWr/HSNBuf WBsyRd/HS-R-F WRlseO/HS-W-F WNoBfr
------------ ----------- ------------- ------------- ------------- ------
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
           0           0             0             0             0      0
```

(scrolled right to screen 4)
```
   NumErrors CacheOption
------------- -----------
   0/     0          NO
   0/     0          NO
   0/     0          NO
   0/     0          NO
   0/     0          NO
   0/     0          NO
   0/     0          NO
   0/     0          NO
   0/     0          NO
   0/     0          NO
   0/     0         CHG
   0/     0         ALL
```

Explanation: In this example, there is a successful invocation of the QUERY POOL command for statistical information about OSAM subpools.

Example 6 for QUERY POOL command

TSO SPOC input:
```
QUERY POOL TYPE(DBAS) SUBTYPE(VSAM) SHOW(STATISTICS)
```

TSO SPOC output:
```
Subpool MbrName CC BufSize PoolId  NBuf  FixOpt LctReq/RRba NewBlk/RKey
------- ------- -- ------- ----   ----- ------ ----------- -----------
VSAM-D  IMS1     0     512 VSM1      10 N/N/N            0           0
VSAM-D  IMS1     0    1024 VSM1    1000 N/N/N            0           0
VSAM-I  IMS1     0    1024 VSM1    1000 N/N/N            0           0
VSAM-D  IMS1     0    4096 VSM1    5000 N/N/N            0           0
VSAM-I  IMS1     0    8192 XYZ1    5000 N/N/N            0           0
VSAM-D  IMS1     0   12288 XYZ3    5000 N/N/N            0           0
VSAM-I  IMS1     0   12288 XYZ3    5000 N/N/N            0           0
VSAM-D  IMS1     0   32768 VSM1   32767 N/N/N            0           0
```

(scrolled right to screen 2)
```
AltReq/BfAlt PurgRq/NRec FndIpl/SyncPt BfSrch/VRds RdReq/Found BfStlW/VWts
------------ ----------- ------------- ----------- ----------- -----------
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
           0           0             0           0           0           0
```

| 0   0   0   0   0   0
| 0   0   0   0   0   0

(scrolled right to screen 3)
PurgWr/HSR-S WBsyId/HSW-S WBsyWr/HSNBuf WBsyRd/HS-R-F WRlseO/HS-W-F WNoBfr
------------ ------------ -------------- ------------- ------------- ------
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0

(scrolled right to screen 4)
     NumErrors

    0/   0
    0/   0
    0/   0
    0/   0
    0/   0
    0/   0
    0/   0
    0/   0

Explanation: In this example, there is a successful invocation of the QUERY POOL command for statistical information about VSAM subpools.

Example 7 for QUERY POOL command

TSO SPOC input:

```
QUERY POOL TYPE(DBAS) SUBTYPE(OSAM,VSAM) SHOW(STATISTICS) SIZE(1024)
```

TSO SPOC output:

```
Subpool  MbrName CC BufSize PoolId  NBuf  FixOpt LctReq/RRba NewBlk/RKey
-------- ------- -- ------- ----    ----- ------ ----------- -----------
OSAM     IMS1    0     1024 OSM1     1000 N/N              0           0
VSAM-D   IMS1    0     1024 VSM2     1000 N/N/N            0           0
VSAM-I   IMS1    0     1024 VSM2     1000 N/N/N            0           0
OSAM     IMS1    0    32768 OSM4       10 N/N              0           0
```

(scrolled right to screen 2)
AltReq/BfAlt PurgRq/NRec FndIpl/SyncPt BfSrch/VRds RdReq/Found BfStlW/VWts
------------ ----------- ------------- ----------- ----------- -----------
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0

(scrolled right to screen 3)
PurgWr/HSR-S WBsyId/HSW-S WBsyWr/HSNBuf WBsyRd/HS-R-F WRlseO/HS-W-F WNoBfr
------------ ------------ -------------- ------------- ------------- ------
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0
             0   0   0   0   0   0
(scrolled right to screen 4)
     NumErrors CacheOption
------------- -----------
    0/   0   NO
    0/   0
    0/   0
    0/   0   ALL

Explanation: In this example, there is a successful invocation of the QUERY POOL command for statistical information about OSAM and VSAM subpools filtered by buffer sizes of 1KB.

Example 8 for QUERY POOL command

TSO SPOC input:
```
QUERY POOL TYPE(DBAS) SUBTYPE(OSAM,VSAM) SHOW(STATISTICS) POOLID(OSM2)
```

TSO SPOC output:

Subpool	MbrName	CC	BufSize	PoolId	NBuf	FixOpt	LctReq/RRba	NewBlk/RKey
OSAM	IMS1	0	2048	OSM2	7832	Y/N	0	0
OSAM	IMS1	0	32768	OSM4	10	Y/N	0	0

(scrolled right to screen 2)

AltReq/BfAlt	PurgRq/NRec	FndIpl/SyncPt	BfSrch/VRds	RdReq/Found	BfStlW/VWts
0	0	0	0	0	0
0	0	0	0	0	0

(scrolled right to screen 3)

PurgWr/HSR-S	WBsyId/HSW-S	WBsyWr/HSNBuf	WBsyRd/HS-R-F	WRlseO/HS-W-F	WNoBfr
0	0	0	0	0	0
0	0	0	0	0	0

(scrolled right to screen 4)

NumErrors	CacheOption
0/ 0	NO
0/ 0	ALL

Explanation: In this example, there is a successful invocation of the QUERY POOL command for statistical information about OSAM and VSAM subpools filtered by pool IDs of OSM2.

Example 9 for QUERY POOL command

TSO SPOC input:
```
QUERY POOL TYPE(DBAS) SUBTYPE(OSAM,VSAM) SHOW(MEMBER)
```

TSO SPOC output:

Subpool	MbrName	CC	BufSize	PoolId	NBuf	ProcMbr	Section
OSAM	IMS1	0	512		100	DFSDFXYZ	OSAM003
OSAM	IMS1	0	1024	OSM1	2000	DFSDFXYZ	OSAM003
OSAM	IMS1	0	2048	OSM2	7832	DFSDFXYZ	OSAM003
OSAM	IMS1	0	4096	OSM3	32787	DFSDFXYZ	OSAM003
OSAM	IMS1	0	6144	ABC1	8000	DFSVSM00	
OSAM	IMS1	0	8192	ABC2	8000	DFSVSM00	
OSAM	IMS1	0	10240	ABC3	8000	DFSVSM00	
OSAM	IMS1	0	12288	ABC4	8000	DFSVSM00	
OSAM	IMS1	0	14336	ABC5	8000	DFSVSM00	
OSAM	IMS1	0	16384	ABC6	8000	DFSVSM00	
OSAM	IMS1	0	32768	OSM9	4	DFSDFXYZ	OSAM003
VSAM-D	IMS1	0	512	VSM1	10	DFSDFXYZ	VSAM001
VSAM-D	IMS1	0	1024	VSM2	1000	DFSDFXYZ	VSAM001
VSAM-I	IMS1	0	1024	VSM2	1000	DFSDFXYZ	VSAM001
VSAM-D	IMS1	0	4096	VSM4	5000	DFSDFXYZ	VSAM001
VSAM-I	IMS1	0	8192	XYZ1	5000	DFSVSM00	

```
VSAM-D  IMS1   0  12288 XYZ3      5000  DFSVSM00
VSAM-I  IMS1   0  12288 XYZ3      5000  DFSVSM00
VSAM-D  IMS1   0  32768 VSM9     32767  DFSDFXYZ VSAM001
```

Explanation: In this example, there is a successful invocation of the QUERY POOL command for member information about OSAM and VSAM subpools.

Example 10 for QUERY POOL command

TSO SPOC input:
```
QUERY POOL TYPE(DBAS) SUBTYPE(OSAM,VSAM) SHOW(ALL)
```

TSO SPOC output:
```
Subpool  MbrName CC BufSize PoolId   NBuf   ProcMbr  Section
-------- ------- -- ------- ------   -----  -------- -------
OSAM     IMS1     0    512            100   DFSDFXYZ OSAM003
OSAM     IMS1     0   1024 OSM1      2000   DFSDFXYZ OSAM003
OSAM     IMS1     0   2048 OSM2      7832   DFSDFXYZ OSAM003
OSAM     IMS1     0   4096 OSM3     32787   DFSDFXYZ OSAM003
VSAM-D   IMS1     0    512 VSM1        10   DFSDFXYZ VSAM001
VSAM-D   IMS1     0   1024 VSM2      1000   DFSDFXYZ VSAM001
VSAM-I   IMS1     0   1024 VSM2      1000   DFSDFXYZ VSAM001
OSAM     IMS1     0  32768 OSM4        10   DFSDFXYZ OSAM003
```

```
(scrolled right to screen 2)
FixOpt LctReq/RRba NewBlk/RKey AltReq/BfAlt PurgRq/NRec FndIpl/SyncPt BfSrch/VRds
------ ----------- ----------- ------------ ----------- ------------- -----------
N/N              0           0            0           0             0           0
N/N              0           0            0           0             0           0
N/N              0           0            0           0             0           0
N/N              0           0            0           0             0           0
N/N/N            0           0            0           0             0           0
N/N/N            0           0            0           0             0           0
N/N/N            0           0            0           0             0           0
N/N              0           0            0           0             0           0
```

```
(scrolled right to screen 3)
RdReq/Found BfStlW/VWts PurgWr/HSR-S WBsyId/HSW-S WBsyWr/HSNBuf WBsyRd/HS-R-F
----------- ----------- ------------ ------------ ------------- -------------
          0           0            0            0             0             0
          0           0            0            0             0             0
          0           0            0            0             0             0
          0           0            0            0             0             0
          0           0            0            0             0             0
          0           0            0            0             0             0
          0           0            0            0             0             0
          0           0            0            0             0             0
```

```
(scrolled right to screen 4)
WRlseO/HS-W-F WNoBfr    NumErrors CacheOption
------------- ------ ------------- -----------
            0      0   0/        0          NO
            0      0   0/        0          NO
            0      0   0/        0          NO
            0      0   0/        0          NO
            0     NA   0/        0
            0     NA   0/        0
            0     NA   0/        0
            0      0   0/        0         ALL
```

Explanation: In this example, there is a successful invocation of the QUERY POOL command for both OSAM and VSAM pool information when TYPE(DBAS) and SHOW(ALL) are specified.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming APIs)

➡ Adjusting OSAM and VSAM database buffers (Database Administration)

➡ Monitoring VSAM buffers (Database Administration)

Related tasks:

➡ OSAM buffers (Database Administration)

Related reference:

➡ /DISPLAY POOL command (Commands)

➡ Command keywords and their synonyms (Commands)

QUERY RM command

Use the QUERY RM command to view status and attributes information about a Resource Manager (RM) address space. This command returns information about IMSRSC repositories managed by RM.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 573
- "Output fields" on page 573
- "Return, reason, and completion codes" on page 575
- "Examples" on page 575

Environment

The QUERY RM command is processed by the Common Service Layer (CSL) Resource Manager (RM) address space.

Syntax

```
►►──┬─QUERY─┬──RM──TYPE(REPO)──SHOW(──┬─ALL────┬──)────────────────────►◄
    └─QRY───┘                         ├─ATTRIB─┤
                                      └─STATUS─┘
```
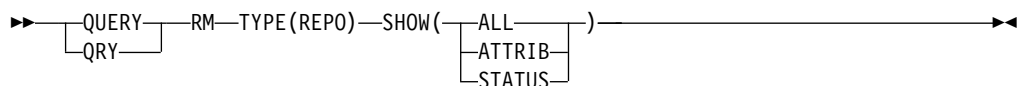
Keywords

The following keywords are valid for the QUERY RM command:

TYPE
    Specifies the type of data or information to be processed.

    REPO
        Specifies that information about all IMSRSC repositories defined to RM is to be returned.

SHOW
    Specifies the information to be returned on the QUERY command.

ALL
> Specifies that all the status and attribute information is to be returned.

ATTRIB
> Specifies that the repository attributes in RM is to be returned. The attributes that are returned are AUDITACCESS settings in RM.

STATUS
> Specifies that the status information in RM is to be returned. For possible status values that can be returned, see "Output fields."

Usage notes

This command can be specified only through the Operations Manager (OM) API.

The QUERY RM command is defined to OM as ROUTE=ALL. The command is processed by each RM that receives the command.

The command syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

The following table shows the QUERY RM output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 183. Output fields for QUERY RM command

Short label	Long label	Keyword	Meaning
AUDACC	AuditAccess	ATTRIB, ALL	AUDITACCESS attribute in RM.
CC	CC	N/A	Completion code for the line of output. Completion code is always returned.
CCTXT	CCText	N/A	Completion code text that briefly explains the meaning of the nonzero completion code.
MBR	MbrName	N/A	Name of the RM that processes the command.
REPOGRP	RepositoryGroup	N/A	z/OS cross-system coupling facility (XCF) group that is associated to the Repository Server.
REPONM	RepositoryName	N/A	Name of the repository that RM is connected to.
REPOTYP	RepositoryType	N/A	Type of the repository that RM is connected to.

Table 183. Output fields for QUERY RM command (continued)

Short label	Long label	Keyword	Meaning
STT	Status	STATUS, ALL	Status. Valid status values that can be returned are:

CONNECTED
> Indicates that RM is connected to the IMSRSC repository and that the repository is available for use.

CONNECT-INCOMPLETE
> Indicates that RM successfully connected to the repository, but RM either failed to correctly update the repository global entry in the resource structure or did not complete the initialization of the AFMT or IMBT in the IMSRSC repository. A CSL2518E error message is issued for any errors in initializing the AFMT or IMBT. In addition, other RM systems in the IMSplex might not have been notified with the directive to connect to the repository.

> To complete the connect processing, the operator must issue the UPDATE RM TYPE(REPO) SET(REPO(Y)) command and route the command to the RM that has the CONNECT-INCOMPLETE status.

DISCONNECT-INCOMPLETE
> Indicates that RM successfully disconnected from the repository, but RM failed to correctly update the repository global entry in the resource structure. In addition, other RM systems in the IMSplex might not have been notified to disconnect from the repository.

> The operator must issue the UPDATE RM TYPE(REPO) SET(REPO(N)) command to complete the disconnect processing.

NOTAVAIL
> Indicates that the repository is not available. RM reconnects to the repository when the Repository Server indicates to RM that the repository is available.

RS-NOTAVAIL
> Indicates that there is no master Repository Server that is available. This status is displayed only when RM has successfully registered with the Repository Server, but the master Repository Server is down and there are no subordinate Repository Servers available to take over. The repository is not available for use until a Repository Server is available.

SPARERECOV
> Indicates that the repository spare recovery process is in progress. The repository is not available during this process.

SPARERCVERR
> Indicates that the repository spare recovery process resulted in an error and that the repository is not available for use. See the FRP messages issued by the Repository Server during the spare recovery process, and fix the error. The repository must be started after the error is fixed to make it available for use.

UPDATE-AUDACC-INCOMPLETE
> Indicates that RM successfully updated the audit access setting in the repository, but RM failed to correctly update the repository global entry in the resource structure. In addition, other RM systems in the IMSplex might not have been notified to update the audit access value from the repository global entry.

> The operator must issue the UPDATE RM TYPE(REPO) SET(REPO(AUDITACCESS)) command to complete the update of the audit access setting.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY RM command. The OM return and reason codes that might be returned as a result of the QUERY RM command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 184. Return and reason codes for the QUERY RM command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY RM command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'03000008'	X'0000203C'	Invalid keyword parameter value.
X'0300000C'	X'00003000'	At least one request was successful.
X'0300000C'	X'00003004'	No requests were successful.
X'03000010'	X'00004504'	RM is not registered to the repository.
X'03000014'	X'00005030'	Command output response allocation failed.

The following table includes an explanation of the completion code.

Table 185. Completion code for the QUERY RM command

Completion code	Meaning
0	The QUERY RM command completed successfully.

Examples

The following are examples of the QUERY RM command:

Example 1 for QUERY RM command

TSO SPOC input:
```
QUERY RM TYPE(REPO) SHOW(ALL)
```

TSO SPOC output:
```
(Screen 1)
RepositoryType MbrName    CC Status      AuditAccess
IMSRSC         RM1RM       0 CONNECTED DEFAULT
IMSRSC         RM2RM       0 CONNECTED DEFAULT
IMSRSC         RM3RM       0 CONNECTED DEFAULT

(scrolled right to screen 2)
RepositoryType MbrName   RepositoryName
IMSRSC         RM1RM     IMSRSC_REPOSITORY
IMSRSC         RM2RM     IMSRSC_REPOSITORY
IMSRSC         RM3RM     IMSRSC_REPOSITORY

(scrolled right to screen 3)
RepositoryType MbrName   RepositoryGroup
IMSRSC         RM1RM     FRPGRUP1
IMSRSC         RM2RM     FRPGRUP1
IMSRSC         RM3RM     FRPGRUP1
```

OM API input:

```
CMD(QUERY RM TYPE(REPO) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.186 22:43:56.527517</statime>
<stotime>2011.186 22:43:56.540575</stotime>
<staseq>C806A80CFEF9D40C</staseq>
<stoseq>C806A80D0229FDC8</stoseq>
<rqsttkn1>USRT005 10154356</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>RM3RM    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>RM              </kwd>
<input>QUERY RM TYPE(REPO) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="REPOTYP" llbl="RepositoryType" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="STT" llbl="Status" scope="LCL" sort="n" key="0" scroll="yes"
len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AUDACC" llbl="AuditAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="REPONM" llbl="RepositoryName" scope="LCL" sort="n" key="0"
 scroll="yes" len="44" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="REPOGRP" llbl="RepositoryGroup" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>REPOTYP(IMSRSC ) MBR(RM3RM) CC(   0) REPONM(IMSRSC_REPOSITORY
                      ) REPOGRP(FRPGRUP1) STT(CONNECTED)
 AUDACC(DEFAULT   ) </rsp>
<rsp>REPOTYP(IMSRSC ) MBR(RM2RM) CC(   0) REPONM(IMSRSC_REPOSITORY
                      ) REPOGRP(FRPGRUP1) STT(CONNECTED)
 AUDACC(DEFAULT   ) </rsp>
<rsp>REPOTYP(IMSRSC ) MBR(RM1RM) CC(   0) REPONM(IMSRSC_REPOSITORY
                      ) REPOGRP(FRPGRUP1) STT(CONNECTED)
 AUDACC(DEFAULT   ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The repository names, the attributes, and status information for all the repositories enabled at RM are returned by the QUERY RM TYPE(REPO) SHOW(ALL) command. In the example, the repository is enabled at all RMs in the IMSplex.

Example 2 for QUERY RM command

TSO SPOC input:

```
QUERY RM TYPE(REPO) SHOW(ATTRIB)
```

TSO SPOC output:

(Screen 1)
```
RepositoryType MbrName    CC AuditAccess RepositoryName>
IMSRSC         RM1RM        0 UPDATE      IMSRSC_REPOSITORY
IMSRSC         RM2RM        0 UPDATE      IMSRSC_REPOSITORY
IMSRSC         RM3RM        0 UPDATE      IMSRSC_REPOSITORY
```

(scrolled right to screen 2)
```
RepositoryType MbrName   ryName RepositoryGroup
IMSRSC         RM1RM             FRPGRUP1
IMSRSC         RM2RM             FRPGRUP1
IMSRSC         RM3RM             FRPGRUP1
```

OM API input:

```
CMD(QUERY RM TYPE(REPO) SHOW(ATRIB)
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.187 17:07:32.104787</statime>
<stotime>2011.187 17:07:32.118744</stotime>
<staseq>C8079EB917653708</staseq>
<stoseq>C8079EB91ACD82D7</stoseq>
<rqsttkn1>USRT005 10100732</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>RM3RM    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>RM             </kwd>
<input>QRY RM TYPE(REPO) SHOW(ATTRIB) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="REPOTYP" llbl="RepositoryType" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="AUDACC" llbl="AuditAccess" scope="LCL" sort="n" key="0"
scroll="yes" len="*" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="REPONM" llbl="RepositoryName" scope="LCL" sort="n" key="0"
 scroll="yes" len="44" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="REPOGRP" llbl="RepositoryGroup" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
<rsp>REPOTYP(IMSRSC  ) MBR(RM3RM) CC(   0) REPONM(IMSRSC_REPOSITORY
                     ) REPOGRP(FRPGRUP1) AUDACC(UPDATE   ) </rsp>
<rsp>REPOTYP(IMSRSC  ) MBR(RM2RM) CC(   0) REPONM(IMSRSC_REPOSITORY
                     ) REPOGRP(FRPGRUP1) AUDACC(UPDATE   ) </rsp>
<rsp>REPOTYP(IMSRSC  ) MBR(RM1RM) CC(   0) REPONM(IMSRSC_REPOSITORY
                     ) REPOGRP(FRPGRUP1) AUDACC(UPDATE   ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY RM SHOW(ATTRIB) information returns the attribute information in RM. This command was issued after the audit access value was changed to UPDATE.

Example 3 for QUERY RM command

TSO SPOC input:

```
QUERY RM TYPE(REPO) SHOW(ALL)
```

TSO SPOC output:

```
Log for . . : QRY RM TYPE(REPO) SHOW(ALL)

IMSplex . . . . . : PLEX1
Routing . . . . . :
Start time. . . . : 2011.187 10:28:53.20
Stop time . . . . : 2011.187 10:28:53.22
Return code . . . : 0200000C
Reason code . . . : 00003004
Reason text . . . : No requests were successful.
Command master. . : RM3RM

            Return    Reason
MbrName     Code      Code      Reason text
--------    --------  --------  --------------
RM3RM       03000010  00004504  Repository is not defined to RM
RM2RM       03000010  00004504  Repository is not defined to RM
RM1RM       03000010  00004504  Repository is not defined to RM
```

OM API input:

```
CMD(QUERY RM TYPE(REPO) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.187 17:28:53.214617</statime>
<stotime>2011.187 17:28:53.225942</stotime>
<staseq>C807A37EDA599405</staseq>
<stoseq>C807A37EDD1D651E</stoseq>
<rqsttkn1>USRT005 10102853</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003004</rsn>
<rsnmsg>CSLN024I</rsnmsg>
<rsntxt>No requests were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="RM3RM   ">
<typ>RM      </typ>
<styp>MULTRM  </styp>
<rc>03000010</rc>
<rsn>00004504</rsn>
<rsntxt>Repository is not defined to RM</rsntxt>
</mbr>
<mbr name="RM2RM   ">
<typ>RM      </typ>
<styp>MULTRM  </styp>
<rc>03000010</rc>
<rsn>00004504</rsn>
<rsntxt>Repository is not defined to RM</rsntxt>
</mbr>
<mbr name="RM1RM   ">
```

```
<typ>RM      </typ>
<styp>MULTRM  </styp>
<rc>03000010</rc>
<rsn>00004504</rsn>
<rsntxt>Repository is not defined to RM</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>RM3RM   </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>RM             </kwd>
<input>QRY RM TYPE(REPO) SHOW(ALL) </input>
</cmd>
</imsout>
```

Explanation: This example shows the QUERY RM SHOW(ALL) command output
when RM is not enabled to use the repository.

Related concepts:

⇨ How to interpret CSL request return and reason codes (System Programming
APIs)

⇨ CSL RM management of the IMSRSC repository (System Administration)

Related reference:

⇨ Command keywords and their synonyms (Commands)

QUERY RTC command

Use the QUERY RTC command to query Fast Path routing codes. A routing code
can be used by the Fast Path Input Edit/Routing Exit Routine (DBFHAGU0) to
route a transaction to a different application program within the same load
balancing group.

Subsections:
- "Environment"
- "Syntax" on page 580
- "Keywords" on page 580
- "Usage notes" on page 584
- "Output fields" on page 584
- "Return, reason, and completion codes" on page 588
- "Examples" on page 590

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 186. Valid environments for the QUERY RTC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY RTC	X		X
NAME	X		X
SHOW	X		X
STATUS	X		X

Syntax

```
>>--+-QUERY-+--RTC--+-NAME(*)------------------------+--------------------->
    '-QRY---'       |            ,-------------        |
                    |           V            |         |
                    '-NAME(--+--name---+-)---'         |
                             '-name*---'

                             ,------------------
                            V                   |
                    +-SHOW(----+-ALL-------+-)---+-------------------------->
                    |          +-DEFN------+     |
                    |          +-DEFNTYPE--+     |
                    |          +-GLOBAL----+     |
                    |          +-IMSID-----+     |
                    |          +-INQ-------+     |
                    |          +-LOCAL-----+     |
                    |          +-MODEL-----+     |
                    |          +-PGM-------+     |
                    |          +-STATUS----+     |
                    |          '-TIMESTAMP-'     |
                    +-SHOW(--+-REGION-+-)--------+
                    |        '-WORK---'          |
                    '-SHOW(EXPORTNEEDED)---------'

>------------------------------------------------------------------------><
    |                ,-------------        |
    |               V              |       |
    '-STATUS(--+-ACTIVE---+-)------'
               +-NOTINIT--+
               +-NOTSCHD--+
               '-STOQ-----'
```
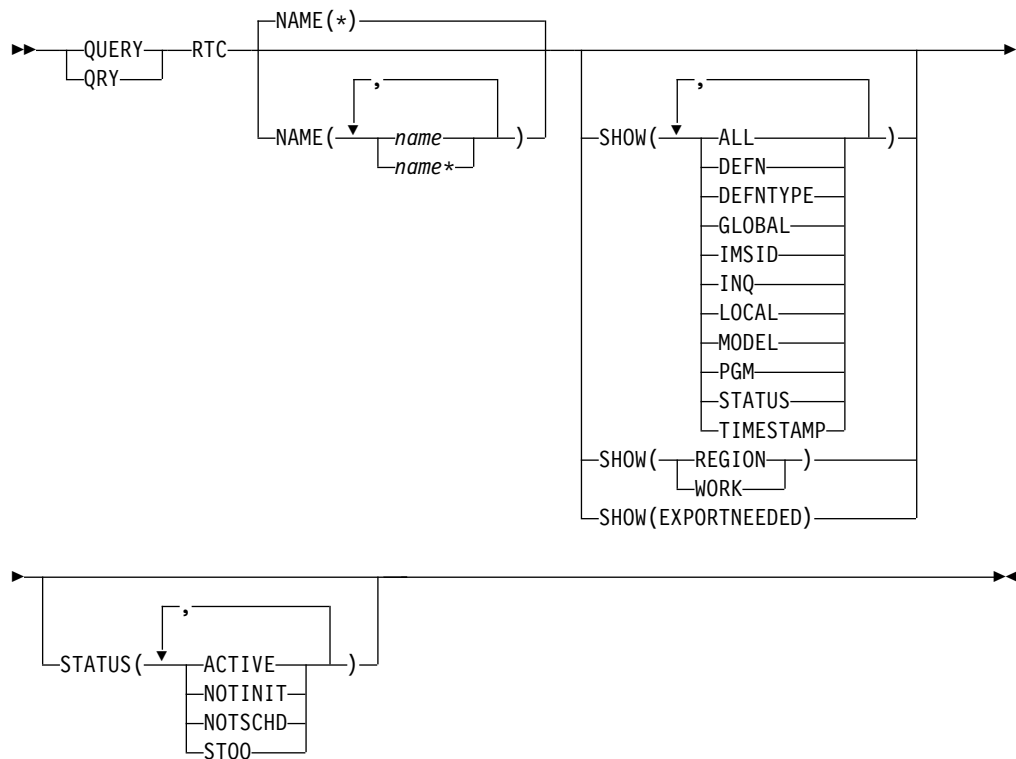
Keywords

The following keywords are valid for the QUERY RTC command:

NAME

Specifies the 1-8 character name of the routing code. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all routing code resources.

SHOW

Specifies the routing code output fields to be returned. The routing code is always returned, along with the name of the IMS that created the output and the completion code. The filters that are supported with the SHOW keyword are:

ALL

Returns all information about the routing code itself. Other SHOW keywords can be specified to return information about resources that are related to the routing code.

Restriction: The ExportNeeded column, which is returned when SHOW(EXPORTNEEDED) is specified, is not returned with SHOW(ALL).

DEFN

Specifies that the resource definitions are to be returned.

The routing code attributes that can be returned are: INQ, PGM, the repository create and update time stamps, and the IMS runtime create, update, import, and access time stamps.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository. If the command master IMS is not enabled to use the repository, SHOW(DEFN,GLOBAL) results in an error return and reason code.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE, MODEL, STATUS, REGION, or WORK.
- The LclStat, LModelName, LModelType, and LDefnType columns, which are returned on the QRY RTC SHOW(ALL) command, are not returned with SHOW(DEFN).
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).
- When querying route code information from the repository, the SHOW(DEFN) filter is not supported when used with the STATUS filter. The runtime filter of STATUS is not valid with SHOW(DEFN), SHOW(DEFN,GLOBAL), SHOW(DEFN,LOCAL), SHOW(DEFN,IMSID), SHOW(DEFN,IMSID,GLOBAL) or SHOW(DEFN,IMSID,LOCAL).

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

The repository information for SHOW(DEFN) or SHOW(DEFN,IMSID) is processed only by the command master IMS and is valid only when the command master IMS is enabled to use the repository.

DEFNTYPE
> Definition type that the resource was defined with.

EXPORTNEEDED
> Specifies that only the routing code resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command.
>
> If the NAME keyword is omitted or NAME(*) is specified, only the names of the routing codes that need to be exported to the repository are returned.
>
> If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each routing code name specified on the command. If the specified routing code does not need to be exported to the repository, an N is returned in the ExportNeeded column.
>
> SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords and is valid only when IMS is enabled with the IMSRSC repository.
>
> The QUERY RTC SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.
>
> **Tip:** For resources and descriptors that have not been exported to the IMSRSC repository, you can issue the EXPORT DEFN TARGET(REPO) OPTION(CHANGESONLY) command to write the resource and descriptor definitions to the repository so that they are available for the next IMS cold start.

GLOBAL
> Specifies that the stored resource definitions from the repository are to be returned. If SHOW(GLOBAL,DEFN) is specified, the global resource definitions from the repository are returned by the command master IMS. SHOW(GLOBAL,DEFN) is valid only when the command master IMS is enabled to use the repository.

IMSID
> Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only when the command master IMS is enabled to use the repository.
>
> When SHOW(IMSID) is specified with the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.
>
> When SHOW(IMSID) is specified without the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.
>
> SHOW(IMSID) cannot be specified with any other SHOW filters other than DEFN and GLOBAL. If SHOW(IMSID,GLOBAL) is specified, GLOBAL is ignored; that is, SHOW(IMSID,GLOBAL) is treated as SHOW(IMSID). SHOW(DEFN,IMSID,LOCAL) is treated as SHOW(DEFN,LOCAL).

INQ
> Inquiry option for transaction messages associated with this routing code.

LOCAL

Specifies that the runtime resource definitions from the IMS system are to be returned.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

MODEL

The model name and model type used to create this resource. If the resource is created with all of the attributes defined and no model specified, the model name and model type are blank. The CREATE command specified without the LIKE keyword creates a resource using the default descriptor as a model. The default descriptor is either the IMS descriptor DBFDSRT1 or user-defined. The CREATE command specified with the LIKE keyword creates a resource using a model. The resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

PGM

Program associated with this routing code.

REGION

Regions where the routing code is active.

Note: You cannot specify this filter with other SHOW filters; you must specify SHOW(REGION) individually.

STATUS

Local routing code status. For a description of the possible routing code status returned, see the STATUS keyword in the "Output fields for the QUERY RTC command" under "Output fields" on page 584.

TIMESTAMP

The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMECREATE) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:
- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

WORK

Work in progress for the routing code specified on NAME parameter and its associated resources. The QRY RTC SHOW(WORK) command can be issued before a DELETE, IMPORT or UPDATE command to check for any work in progress for the specified routing code and any of its associated

resources. Any work in progress might cause the subsequent DELETE, IMPORT or UPDATE commands to fail. The QRY RTC SHOW(WORK) command returns the resource name, resource type, and work status for the work in progress for the routing code specified on the NAME parameter or work in progress for an associated resource. If no work is in progress for the specified resource as response line is returned with a work status of blanks.

SHOW(WORK) specified with NAME(*) might take a long time.

Notes:
1. You cannot specify this filter with other SHOW filters; you must specify SHOW(WORK) individually.
2. The QRY RTC SHOW(WORK) command is not valid on an XRF alternate.

STATUS()
Selects routing codes for display that match the NAME parameter and possess at least one of the specified routing code status values.

ACTIVE
Sets the STATUS() filter to return information about active routing codes.

NOTINIT
Sets the STATUS() filter to return information about routing codes that are not initialized and cannot be used.

NOTSCHD
Sets the STATUS() filter to return information about routing codes that are not scheduled.

STOQ
Sets the STATUS() filter to return information about routing codes that are stopped. Transactions that are associated with this routing code are not processed.

Usage notes

This command can be issued only through the Operations Manager API. Fast Path must be installed on the system. This command applies to DB/DC and DCCTL systems.

This command is allowed on XRF alternate and RSR tracker systems.

If you want to display information about resource definitions, specify SHOW(DEFN). If you want to know which IMS systems have the resource defined and also know the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource defined, specify SHOW(IMSID).

Output fields

The following table shows the QUERY RTC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
> Provides a brief description of the output field.

Table 187. Output fields for the QUERY RTC command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	LCL	Completion code.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DFNT	LDefnType	DEFNTYPE	LCL	Definition type, which can be one of the following:
				CREATE Defined by a CREATE command.
				IMPORT Defined by an IMPORT command.
				IMS Defined by IMS. RTCDUMMY is an IMS-defined routing code created by the system definition process if no routing codes are defined.
				MODBLKS Defined by system definition in the MODBLKS data set.
				UPDATE Defined by system definition in the MODBLKS data set, but changed into a dynamic resource by an UPDATE command.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the routing code has been exported to the IMSRSC repository. The value can be Y or N.
IMSID	IMSid	IMSID	GBL	Returns the IMSIDs that have resource defined from the repository.
INQ	LInq	INQ, DEFN	LCL	Inquiry transaction associated with routing code (Y) or not (N). The value is obtained from the local IMS.
				N The inquiry option is not enabled.
				Y The inquiry option is enabled. The transaction messages associated with this routing code do not cause a change to a database. Programs are prohibited from issuing Insert, Delete, or Replace calls to a database.

Table 187. Output fields for the QUERY RTC command (continued)

Short label	Long label	Keyword	Scope	Meaning
LSTT	LcLStat	STATUS	LCL	Local routing code status.
				ACTIVE Routing code is active.
				NOTINIT-*xx-reason* Routing code is not initialized and cannot be used. The NOTINIT status is displayed in the format NOTINIT-*xx-reason*, to identify where an error was detected and the reason the routing code was not initialized. *xx* is the reason code that identifies the unique location in one module where this reason code is set. DBFRCTE MACRO defines each reason code that might be set in the routing code bad reason code (field RCTEBADR) and identifies the module that sets it. NOTINIT-00 indicates that the reason is unknown. Call IBM Software Support (Action: 1). *reason* explains the reason code *xx* in abbreviated text format up to 13 characters. For possible reason values and their descriptions, see Table 188 on page 587.
				NOTSCHD Routing code is not scheduled or stopped.
				STOQ Routing code is stopped for queuing of input message.
MBR	MbrName	N/A	LCL	IMSplex member that built the output line.
MDLN	LModelName	MODEL	LCL	Model name. Name of the resource used as a model to create this resource. DBFDSRT1 is the IMS descriptor name for routing codes.
MDLT	LModelType	MODEL	LCL	Model type, either RSC or DESC. RSC means that the resource was created using another resource as a model. DESC means that the resource was created using a descriptor as a model.
PGM	PgmName	PGM	LCL	Program name.
REG	Region	REGION	LCL	Region where the routing code is active.
REPO	Repo	DEFN	GBL	Indicates whether the output line contains the stored resource definitions. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
RINQ	Inq	INQ, DEFN	GBL	Inquiry transaction associated with routing code (Y) or not (N). The value is obtained from the repository.
RPGM	PgmName	DEFN	GBL	Program name. The value is obtained from the repository.

Table 187. Output fields for the QUERY RTC command (continued)

Short label	Long label	Keyword	Scope	Meaning
RTC	Rtcode	RTC	LCL	Routing code name.
RTMCR	TimeCreate	DEFN	GBL	Create time from the repository. This is the time the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN	GBL	Update time from the repository. This is the time the resource was last updated in the repository.
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the resource was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. The value is returned from the local IMS. For a routing code resource, the following actions update the last access time: • When a message is queued to the balancing group using the specified routing code. • CREATE command references the resource as a model.
TMCR	LTimeCreate	TIMESTAMP	LCL	The time that the routing code was created. This is the result of a CREATE RTC command, IMPORT command that creates the routing code, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT. The value is returned from the local IMS.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the resource was last imported, if applicable. The import time is retained across warm start and emergency restart. The value is returned from the local IMS.
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated as a result of the UPDATE RTC command or the IMPORT command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
WRK	Work	WORK	LCL	Work is in progress for the routing code or one of its associated resources. The work in progress can be one of the following: **ACTIVE** Routing code is active. **ANOTHER CMD IN PROGRESS** Another command (such as DELETE or UPDATE) to delete or update the routing code is already in progress.

Table 188. Reason information for NOTINIT-xx-reason status

Reason	Meaning
NOPGM	The program that this routing code references is not defined. No program PDIR control block exists. Action: 2.

Table 188. Reason information for NOTINIT-xx-reason status (continued)

Reason	Meaning
REPOCHGLIST	The routing code name exists in the change list in the IMSRSC repository for this IMS, and the change list is being processed or the change list processing failed because of an error. This routing code cannot be used, scheduled, updated, exported, or referenced as a model until it is successfully imported so that the runtime routing code definition and the stored definition in the repository are synchronized. Action: 3.

Note: Actions that can be taken to initialize the routing code are:
1. Call IBM Software Support.
2. Issue the CREATE PGM command to create the program. Issue the UPDATE RTC START(Q) command to complete routing code initialization.
3. If IMS change list processing failed and message DFS4411E was issued, correct the problem and take one of the following actions:
   - Shut down IMS and restart it in order to try processing the change list again.
   - Issue the IMPORT DEFN SOURCE(REPO) NAME(*rtcname*) TYPE(RTC) command to import the routing code, where *rtcname* is the routing code name.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 189. Return and reason codes for the QUERY RTC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000004'	X'00001010'	No resources were found to be returned. The resource names specified might be invalid, or: • There were no resources that match the filter specified. • There were no resources that had work to display for the SHOW(WORK) specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'00004004'	No CQS address space.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.

Table 189. Return and reason codes for the QUERY RTC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004018'	No resource structure, or resource structure is not available.
X'00000010'	X'00004024'	No Fast Path defined.
X'00000010'	X'00004100'	Resource structure is full.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS OLC, or MODBLKS not defined).
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Repository access is denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server is being shut down.
X'00000010'	X'0000450B'	Repository Server is not available.
X'00000010'	X'0000450C'	Repository Server is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields related to the IMSRSC repository.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'0000501C'	IMODULE GETMAIN error.
X'00000014'	X'00005100'	RM request error.
X'00000014'	X'00005104'	CQS error.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY RTC command is rejected because the resource list is unusable. Contact IBM Software Support. Send in the output from the command in error, and a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the IMSRSC repository RID and RMD data sets.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 190. Completion codes for the QUERY RTC command

Completion code	Completion code text	Meaning
0		Command completed successfully for routing code.
10	NO RESOURCES FOUND	Routing code name is invalid, or the wildcard parameter specified does not match any resource names.

Examples

The following are examples of the QUERY RTC command:

Example 1 for QUERY RTC command

TSO SPOC input:
```
QUERY RTC NAME(%%%%,TXCDJN11,TXBANKI2,RTC0*) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
Rtcode    MbrName    CC LPgmName LInq LclStat LModelName LModelType
GFP1      IMS1        0 DCGPSBFP N    NOTSCHD
GFP2      IMS1        0 DCGPSBF2 N    NOTSCHD
RTC00001  IMS1        0 DCGPSBF2 N    NOTSCHD GFP2       RSC
TXBANKI2  IMS1        0 BANKIFP  N    NOTSCHD
TXCDJN11  IMS1        0 DDLTJN11 Y    NOTSCHD
(scrolled to the right screen 2)
Rtcode    MbrName    CC LTimeCreate        LTimeUpdate
GFP1      IMS1        0 2011.180 12:37:38.07
GFP2      IMS1        0 2011.180 12:37:38.07
RTC00001  IMS1        0 2011.180 12:40:05.33
TXBANKI2  IMS1        0 2011.180 12:37:38.07
TXCDJN11  IMS1        0 2011.180 12:37:38.07
(scrolled to the right screen 3)
Rtcode    MbrName    CC LTimeAccess        LTimeImport        LDefnType
GFP1      IMS1        0                                       MODBLKS
GFP2      IMS1        0 2011.180 13:18:54.79                  MODBLKS
RTC00001  IMS1        0                                       CREATE
TXBANKI2  IMS1        0                                       MODBLKS
TXCDJN11  IMS1        0                                       MODBLKS
```

OM API input:
```
CMD(QUERY RTC NAME(%%%%,TXCDJN11,TXBANKI2,RTC0*) SHOW(ALL))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.180 21:00:03.888063</statime>
<stotime>2011.180 21:00:03.888992</stotime>
<staseq>C7FF05A4527BF4CA</staseq>
<stoseq>C7FF05A452B602CA</stoseq>
<rqsttkn1>USRT005 10140003</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
```

```
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>RTC            </kwd>
<input>QUERY RTC NAME(%%%%,TXCDJN11,TXBANKI2,RTC0*) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="RTC" llbl="Rtcode" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
 len="4" dtype="CHAR" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="PGM" llbl="LPgmName" scope="LCL" sort="n" key="0"
scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="INQ" llbl="LInq" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
<hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" />
<hdr slbl="MDLN" llbl="LModelName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MDLT" llbl="LModelType" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RTC(GFP1    ) MBR(IMS1    ) CC(   0)  PGM(DCGPSBFP) INQ(N)
 LSTT(NOTSCHD) TMCR(2011.180 12:37:38.07) TMUP(                    )
TMAC(                    ) TMIM(                    ) DFNT(MODBLKS )
</rsp>
<rsp>RTC(GFP2    ) MBR(IMS1    ) CC(   0)  PGM(DCGPSBF2) INQ(N)
 LSTT(NOTSCHD) TMCR(2011.180 12:37:38.07) TMUP(                    )
 TMAC(2011.180 13:18:54.79) TMIM(                    ) DFNT(MODBLKS )
</rsp>
<rsp>RTC(TXCDJN11) MBR(IMS1    ) CC(   0)  PGM(DDLTJN11) INQ(Y)
 LSTT(NOTSCHD) TMCR(2011.180 12:37:38.07) TMUP(                    )
 TMAC(                    ) TMIM(                    ) DFNT(MODBLKS )
</rsp>
<rsp>RTC(TXBANKI2) MBR(IMS1    ) CC(   0)  PGM(BANKIFP ) INQ(N)
 LSTT(NOTSCHD) TMCR(2011.180 12:37:38.07) TMUP(                    )
 TMAC(                    ) TMIM(                    ) DFNT(MODBLKS )
</rsp>
<rsp>RTC(RTC00001) MBR(IMS1    ) CC(   0)  PGM(DCGPSBF2) INQ(N)
 LSTT(NOTSCHD) MDLN(GFP2    ) MDLT(RSC) TMCR(2011.180 12:40:05.33)
 TMUP(                    ) TMAC(                    ) TMIM(
        ) DFNT(CREATE  )</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY RTC command is specified with SHOW(ALL), so all
output fields are returned for the specified routing codes. All of the routing code
output fields do not fit on one screen, so the user must scroll to the right for
additional output fields. The routing code name, the member name that built the
line of output, and the completion code are displayed on every screen. Routing
code wildcard parameter %%%% causes all routing codes with 4-character names

to be displayed. The routing codes that were generated in stage 1 MODBLKS generation are created when IMS cold starts and defined with a definition type of MODBLKS. Routing codes RTC00001 and RTC00002 were created dynamically using CREATE RTC commands and display a definition type of CREATE. RTC00001 was created using routing code GFP2 as the model and RTC00002 was created using default descriptor DBFDSRT1. Routing code GFP2 displays the last access time stamp, from when it was referred to by the CREATE RTC command that created routing code RTC00001, using routing code GFP2 as the model. Routing code RTC00002 shows the last update time, because its program was changed by an UPDATE RTC command since it was created.

Example 2 for QUERY RTC command

TSO SPOC input:
```
QUERY RTC NAME(RTC00001) SHOW(DEFN,PGM,INQ)
```

TSO SPOC output:
```
Rtcode    MbrName     CC Repo IMSid PgmName  LPgmName Inq LInq
RTC00001 IMS1         0 Y          DCGPSBF2          N
RTC00001 IMS1         0      IMS1            DCGPSBF2      N
RTC00001 IMS2         0      IMS2            DCGPSBF2      N
RTC00001 IMS3         0      IMS3            DCGPSBF2      N
```

OM API input:
```
CMD(QUERY RTC NAME(RTC00001) SHOW(DEFN,PGM,INQ))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.180 20:50:14.242034</statime>
<stotime>2011.180 20:50:14.300210</stotime>
<staseq>C7FF0371FDEF225C</staseq>
<stoseq>C7FF03720C232D8C</stoseq>
<rqsttkn1>USRT005 10135014</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>RTC            </kwd>
<input>QUERY RTC NAME(RTC00001) SHOW(DEFN,PGM,INQ) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="RTC" llbl="Rtcode" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
 len="4" dtype="CHAR" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RPGM" llbl="PgmName" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
```

```
<hdr slbl="PGM" llbl="LPgmName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="RINQ" llbl="Inq" scope="GBL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
<hdr slbl="INQ" llbl="LInq" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RTC(RTC00001)  MBR(IMS3   ) CC(   0)  PGM(DCGPSBF2) INQ(N)
 IMSID(IMS3)</rsp>
<rsp>RTC(RTC00001) MBR(IMS1    ) CC(   0) REPO(Y) RINQ(N)
 RPGM(DCGPSBF2) </rsp>
<rsp>RTC(RTC00001)  MBR(IMS1   ) CC(   0)  PGM(DCGPSBF2) INQ(N)
 IMSID(IMS1)</rsp>
<rsp>RTC(RTC00001)  MBR(IMS2   ) CC(   0)  PGM(DCGPSBF2) INQ(N)
 IMSID(IMS2)</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The stored resource definitions and the runtime resource definitions
for the specified resources are returned. Only the program name and the INQ
information are returned because the SHOW(PGM,INQ) option is specified.

Example 3 for QUERY RTC command

TSO SPOC input:

```
QUERY RTC NAME(E*) SHOW(DEFN)
```

TSO SPOC output:

```
(screen 1)
Rtcode    MbrName    CC Repo IMSid PgmName   LPgmName Inq LInq TimeCreate>
EMHTX2    IMS1       0 Y          EMHPSB2            N        2011.180 12:37:33.37
EMHTX2    IMS1       0    IMS1             EMHPSB2       N
EMHTX2    IMS2       0    IMS2             EMHPSB2       N
EMHTX22   IMS1       0 Y          EMHPSB2            N        2011.180 12:37:33.37
EMHTX22   IMS1       0    IMS1             EMHPSB2       N
EMHTX22   IMS2       0    IMS2             EMHPSB2       N
EMHTX3    IMS1       0 Y          EMHPSB2            N        2011.180 12:37:33.37
EMHTX3    IMS1       0    IMS1             EMHPSB2       N
EMHTX3    IMS2       0    IMS2             EMHPSB2       N
EMHTX32   IMS1       0 Y          EMHPSB2            N        2011.180 12:37:33.37
EMHTX32   IMS1       0    IMS1             EMHPSB2       N
EMHTX32   IMS2       0    IMS2             EMHPSB2       N

(scrolled right to screen 2)
Rtcode    MbrName    CC Repo LTimeCreate           TimeUpdate           LTimeUpdate
EMHTX2    IMS1       0 Y
EMHTX2    IMS1       0    2011.180 12:37:38.07
EMHTX2    IMS2       0    2011.180 12:37:38.08
EMHTX22   IMS1       0 Y
EMHTX22   IMS1       0    2011.180 12:37:38.07
EMHTX22   IMS2       0    2011.180 12:37:38.08
EMHTX3    IMS1       0 Y
EMHTX3    IMS1       0    2011.180 12:37:38.07
EMHTX3    IMS2       0    2011.180 12:37:38.08
EMHTX32   IMS1       0 Y
EMHTX32   IMS1       0    2011.180 12:37:38.07
EMHTX32   IMS2       0    2011.180 12:37:38.08
(scrolled right to screen 3)
Rtcode    MbrName    CC Repo LTimeAccess           LTimeImport
EMHTX2    IMS1       0 Y
EMHTX2    IMS1       0
EMHTX2    IMS2       0
EMHTX22   IMS1       0 Y
EMHTX22   IMS1       0
EMHTX22   IMS2       0
EMHTX3    IMS1       0 Y
EMHTX3    IMS1       0
```

```
EMHTX3   IMS2        0
EMHTX32  IMS1        0 Y
EMHTX32  IMS1        0
EMHTX32  IMS2        0
```

OM API input:

```
CMD(QUERY RTC NAME(E*) SHOW(DEFN))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.180 21:11:32.267556</statime>
<stotime>2011.180 21:11:32.351185</stotime>
<staseq>C7FF0834CFE246DE</staseq>
<stoseq>C7FF0834E44D1F9A</stoseq>
<rqsttkn1>USRT005 10141132</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>RTC             </kwd>
<input>QUERY RTC NAME(E*) SHOW(DEFN) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="RTC" llbl="Rtcode" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
 len="4" dtype="CHAR" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RPGM" llbl="PgmName" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="PGM" llbl="LPgmName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="RINQ" llbl="Inq" scope="GBL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
<hdr slbl="INQ" llbl="LInq" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
<hdr slbl="RTMCR" llbl="TimeCreate" scope="GBL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="RTMUP" llbl="TimeUpdate" scope="GBL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
<hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
 scroll="yes" len="20" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RTC(EMHTX2  ) MBR(IMS1    ) CC(   0) PGM(EMHPSB2 ) INQ(N)
 IMSID(IMS1) TMCR(2011.180 12:37:38.07) TMUP(                    )
```

```
   TMAC(                   ) TMIM(                    )</rsp>
<rsp>RTC(EMHTX22 )  MBR(IMS1    ) CC(   0)  PGM(EMHPSB2 ) INQ(N)
 IMSID(IMS1) TMCR(2011.180 12:37:38.07) TMUP(                   )
   TMAC(                   ) TMIM(                    )</rsp>
<rsp>RTC(EMHTX3  )  MBR(IMS1    ) CC(   0)  PGM(EMHPSB2 ) INQ(N)
 IMSID(IMS1) TMCR(2011.180 12:37:38.07) TMUP(                   )
   TMAC(                   ) TMIM(                    )</rsp>
<rsp>RTC(EMHTX32 )  MBR(IMS1    ) CC(   0)  PGM(EMHPSB2 ) INQ(N)
 IMSID(IMS1) TMCR(2011.180 12:37:38.07) TMUP(                   )
   TMAC(                   ) TMIM(                    )</rsp>
<rsp>RTC(EMHTX2  )  MBR(IMS1    ) CC(   0) REPO(Y) RINQ(N) RPGM(EMHPSB2
 ) RTMCR(2011.180 12:37:33.37) </rsp>
<rsp>RTC(EMHTX22 )  MBR(IMS1    ) CC(   0) REPO(Y) RINQ(N) RPGM(EMHPSB2
 ) RTMCR(2011.180 12:37:33.37) </rsp>
<rsp>RTC(EMHTX3  )  MBR(IMS1    ) CC(   0) REPO(Y) RINQ(N) RPGM(EMHPSB2
 ) RTMCR(2011.180 12:37:33.37) </rsp>
<rsp>RTC(EMHTX32 )  MBR(IMS1    ) CC(   0) REPO(Y) RINQ(N) RPGM(EMHPSB2
 ) RTMCR(2011.180 12:37:33.37) </rsp>
<rsp>RTC(EMHTX2  )  MBR(IMS2    ) CC(   0)  PGM(EMHPSB2 ) INQ(N)
IMSID(IMS2) TMCR(2011.180 12:37:38.08) TMUP(                   )
   TMAC(                   ) TMIM(                    )</rsp>
<rsp>RTC(EMHTX22 )  MBR(IMS2    ) CC(   0)  PGM(EMHPSB2 ) INQ(N)
 IMSID(IMS2) TMCR(2011.180 12:37:38.08) TMUP(                   )
   TMAC(                   ) TMIM(                    )</rsp>
<rsp>RTC(EMHTX3  )  MBR(IMS2    ) CC(   0)  PGM(EMHPSB2 ) INQ(N)
 IMSID(IMS2) TMCR(2011.180 12:37:38.08) TMUP(                   )
   TMAC(                   ) TMIM(                    )</rsp>
<rsp>RTC(EMHTX32 )  MBR(IMS2    ) CC(   0)  PGM(EMHPSB2 ) INQ(N)
 IMSID(IMS2) TMCR(2011.180 12:37:38.08) TMUP(                   )
   TMAC(                   ) TMIM(                    )</rsp>
</cmdrspdata>
</imsout>
```

Explanation: A line is returned for each resource that matches the wildcard name.
The resource definitions from each IMS that has the resource defined and the
global repository definition are returned. The repository information is returned by
the command master IMS. There are no IMS-specific sections in the repository for
each resource name that matches the wildcard name.

Example 4 for QUERY RTC command

TSO SPOC input:
```
QUERY RTC NAME(RTC00001) SHOW(EXPORTNEEDED)
```

TSO SPOC output:
```
Rtcode    MbrName    CC ExportNeeded
RTC00001 IMS1       0 N
```

OM API input:
```
CMD(QUERY RTC SHOW(EXPORTNEEDED))
```

Related concepts:

➡️ How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

➡️ /DISPLAY MODIFY command (Commands)

➡️ /DISPLAY RTCODE command (Commands)

➡️ /DISPLAY STATUS command (Commands)

QUERY RTCDESC command

Use the QUERY RTCDESC command to query Fast Path routing descriptors. A descriptor is a model that can be used to create descriptors.

Subsections:

- "Environment"
- "Syntax"
- "Keywords" on page 597
- "Usage notes" on page 600
- "Output fields" on page 600
- "Return, reason, and completion codes" on page 603
- "Example" on page 605

Environment

The following table lists the environments (DB/BC, DBCTL and DCCTL) in which you can use the commands and keywords.

Table 191. Valid environments for the QUERY RTCDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY RTCDESC	X		X
NAME	X		X
SHOW	X		X
DEFAULT	X		X

Syntax

```
                                                                          DEFAULT(──N──)──┐
                       ┌────,────┐                                    └─┘      └─Y─┘        ►◄
         ──SHOW(──▼──┬─ALL──────┬──)──
                     ├─DEFAULT───┤
                     ├─DEFN──────┤
                     ├─DEFNTYPE──┤
                     ├─GLOBAL────┤
                     ├─IMSID─────┤
                     ├─INQ───────┤
                     ├─LOCAL─────┤
                     ├─MODEL─────┤
                     ├─PGM───────┤
                     └─TIMESTAMP─┘
         ──SHOW(EXPORTNEEDED)────
```
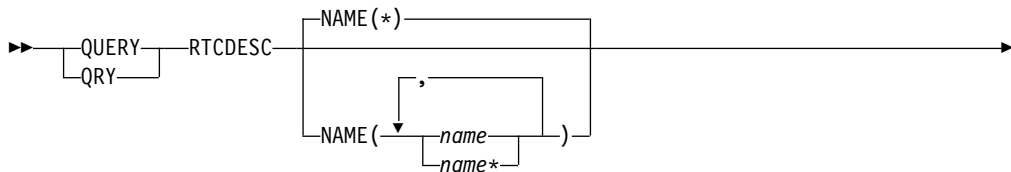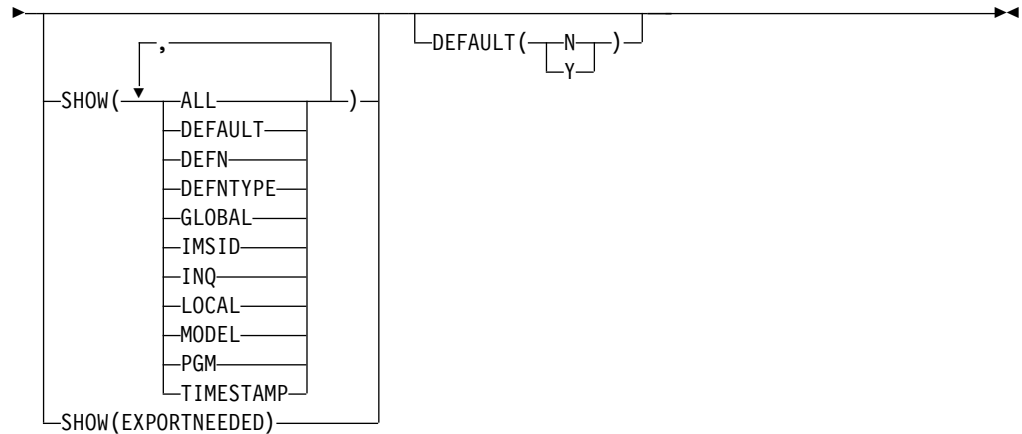
Keywords

The following keywords are valid for the QUERY RTCDESC command:

NAME

Specifies the 1-8 character name of the routing code descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all routing code descriptor resources.

DEFAULT

Selects routing code descriptors for display that possess the default value specified. DEFAULT(Y) displays the one and only default descriptor. DEFAULT(N) displays all the other descriptors that are not the default.

SHOW

Specifies the routing code descriptor output fields to be returned. The routing code descriptor name is always returned, along with the name of the IMS that created the output and the completion code. The filters supported with the SHOW keyword are:

ALL

Returns all information about the routing code descriptor itself. Other SHOW keywords can be specified to return information about resources related to the routing code descriptor.

Restriction: The ExportNeeded column, which is returned when SHOW(EXPORTNEEDED) is specified, is not returned with SHOW(ALL).

DEFAULT

Default descriptor option.

DEFN

Specifies that the resource definitions are to be returned.

The routing code descriptor attributes that can be returned are: INQ, PGM, DEFAULT, the repository create and update time stamps, and the IMS runtime create, update, import and access time stamps.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes, including those defined globally in the repository and those defined locally in the IMS system, are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the

repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE or MODEL.
- The LModelName, LModelType, and LDefnType columns, which are returned on the QRY RTCDESC SHOW(ALL) command, are not returned with SHOW(DEFN).
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).

When querying route code descriptor information from the repository, resource definitions stored in the repository are used to determine the response lines with the repository information, and the runtime resource definitions are used to determine the response lines with the IMS runtime resource information. The response lines are returned for each stored resource or runtime resource definition that matches the specified filter. If SHOW(DEFN,GLOBAL) is specified, only the stored resource definitions that match the specified filter are returned. If SHOW(DEFN,LOCAL) is specified, only the runtime resource definitions that match the specified filter are returned.

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

DEFNTYPE
Definition type that the descriptor was defined with.

EXPORTNEEDED
Specifies that only the routing code descriptor resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the routing code descriptors that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each routing code descriptor name specified on the command. If the specified routing code descriptor does not need to be exported to the repository, an `N` is returned in the ExportNeeded column.

SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords and is valid only when IMS is enabled with the IMSRSC repository.

The QUERY RTCDESC SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources and descriptors that have not been exported to the IMSRSC repository, you can issue the EXPORT DEFN TARGET(REPO) OPTION(CHANGESONLY) command to write the resource and descriptor definitions to the repository so that they are available for the next IMS cold start.

GLOBAL
Specifies that the stored resource definitions from repository are to be returned. If SHOW(GLOBAL,DEFN) is specified, the global resource definitions from the repository are returned by the command master IMS. SHOW(GLOBAL,DEFN) is valid only when the command master IMS is enabled to use the repository.

IMSID
Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only if the command master IMS is enabled to use the repository.

When SHOW(IMSID) is specified with the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

SHOW(IMSID) cannot be specified with any other SHOW filters other than DEFN and GLOBAL. If SHOW(IMSID,GLOBAL) is specified, GLOBAL is ignored; that is, SHOW(IMSID,GLOBAL) is treated as SHOW(IMSID). SHOW(DEFN,IMSID,LOCAL) is treated as SHOW(DEFN,LOCAL).

INQ
Inquiry option for transaction messages that are associated with this routing code descriptor.

LOCAL
Specifies that the runtime resource definitions from the IMS system are to be returned.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

MODEL
The model name and model type used to create this descriptor. If the

descriptor is created with all of the attributes defined and no model specified, the model name and model type are blank. The CREATE command specified without the LIKE keyword creates a descriptor using the default descriptor as a model. The default descriptor is either the IMS descriptor DBFDSRT1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor by using a model. The descriptor is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

PGM
Program that is associated with this routing code.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMECREATE) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

Usage notes

This command can be issued only through the Operations Manager API. Fast Path must be installed on the system. This command applies to DB/DC and DCCTL systems.

This command is allowed on XRF alternate and RSR tracker systems.

The QUERY RTCDESC command is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

If you want to display information about resource definitions, specify SHOW(DEFN). If you want to know which IMS systems have the resource defined and also know the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource defined, specify SHOW(IMSID).

Output fields

The following table shows the QUERY RTCDESC output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
> Provides a brief description of the output field.

Table 192. Output fields for the QUERY RTCDESC command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	LCL	Completion code.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the non-zero completion code.
DESC	DescName	RTCDESC	LCL	Routing code descriptor name.
DFNT	LDefnType	DEFNTYPE	LCL	Definition type, which can be one of the following: **CREATE** Defined by a CREATE command. **IMPORT** Defined by an IMPORT command. **IMS** Defined by IMS. DBFDSRT1 is an IMS-defined routing code descriptor containing the default routing code values.
DFLT	LDflt	DEFAULT	LCL	Default descriptor (Y) or not (N). **N** The descriptor is not the default. **Y** The descriptor is the default. When a descriptor is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a default routing code descriptor called DBFDSRT1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the routing code descriptor has been exported to the IMSRSC repository. The value can be Y or N.
IMSID	IMSid	IMSID	GBL	Returns the IMSIDs that have the resource defined. The output value is obtained from the repository.

Table 192. Output fields for the QUERY RTCDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
INQ	LInq	INQ, DEFN	LCL	Inquiry transaction associated with routing code (Y) or not (N). The output value is obtained from the local IMS. **N** The inquiry option is not enabled. **Y** The inquiry option is enabled. The transaction messages associated with this routing code do not cause a change to a database. Programs are prohibited from issuing Insert, Delete, or Replace calls to a database.
MBR	MbrName	N/A	LCL	IMSplex member that built the output line.
MDLN	LModelName	MODEL	LCL	Model name. Name of the descriptor used as a model to create this resource. DBFDSRT1 is the IMS descriptor name for routing codes.
MDLT	LModelType	MODEL	LCL	Model type, either RSC or DESC. RSC means that the resource was created using another resource as a model. DESC means that the resource was created using a descriptor as a model.
PGM	PgmName	PGM	LCL	Program name.
RDFLT	Dflt	DEFN	GBL	Default descriptor (Y) or not (N). The value is obtained from the repository.
REPO	Repo	DEFN	GBL	Indicates whether the output line contains the stored resource definitions. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
RINQ	Inq	INQ, DEFN	GBL	Inquiry transaction associated with routing code (Y) or not (N). The output value is obtained from the repository.
RPGM	PgmName	DEFN	GBL	Program name. The value is obtained from the repository.
RTMCR	TimeCreate	DEFN	GBL	Create time from the repository. This is the time the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN	GBL	Update time from the repository. This is the time the resource was last updated in the repository.
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the routing code descriptor was last accessed. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. The output value is obtained from the local IMS. For a routing code descriptor, when the CREATE command references the descriptor as a model, the last access time is updated.

Table 192. Output fields for the QUERY RTCDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
TMCR	LTimeCreate	TIMESTAMP	LCL	The time that the routing code descriptor was created. The output value is obtained from the local IMS. This value is the result of a CREATE RTCDESC command, IMPORT command that creates the routing code, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the routing code descriptor was last imported, if applicable. The import time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated as a result of the UPDATE RTCDESC command or the IMPORT command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 193. Return and reason codes for the QUERY RTCDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000004'	X'00001010'	No resources were found to be returned. The resource names specified might be invalid, or: • There were no resources that match the filter specified • There were no resources that had work to display for the SHOW(WORK) specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'00004004'	No CQS address space.

Table 193. Return and reason codes for the QUERY RTCDESC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	No resource structure exists, or resource structure is not available.
X'00000010'	X'00004024'	No Fast Path defined.
X'00000010'	X'00004100'	Resource structure is full.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS OLC, or MODBLKS not defined).
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Repository access is denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server is being shut down.
X'00000010'	X'0000450B'	Repository Server is not available.
X'00000010'	X'0000450C'	Repository Server is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields related to the IMSRSC repository.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'0000501C'	IMODULE GETMAIN error.
X'00000014'	X'00005100'	RM request error.
X'00000014'	X'00005104'	CQS error.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY RTCDESC command is rejected because the resource list is unusable. Contact IBM Software Support. Send in the output from the command in error, and a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the IMSRSC repository RID and RMD data sets.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 194. Completion codes for the QUERY RTCDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for routing code or routing code descriptor.
10	NO RESOURCES FOUND	Routing code or routing code descriptor name is invalid, or the wildcard parameter specified does not match any resource names.

Example

The following is an example of the QUERY RTCDESC command:

Example 1 for QUERY RTCDESC command

TSO SPOC input:

```
QRY RTCDESC SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
DescName MbrName    CC LPgmName LInq LModelName LModelType LDflt
DBFDSRT1 IMS1        0          N                          Y

(scrolled to the right screen 2)
DescName MbrName    CC LTimeCreate         LTimeUpdate
DBFDSRT1 IMS1        0 2011.180 12:37:38.07

(scrolled to the right screen 3)
DescName MbrName    CC LTimeAccess         LTimeImport         LDefnType
DBFDSRT1 IMS1        0 2011.180 13:18:55.75                    IMS
```

OM API input:

```
CMD(QUERY RTCDESC SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.180 21:20:16.670548</statime>
<stotime>2011.180 21:20:16.671350</stotime>
<staseq>C7FF0A28EBF54657</staseq>
<stoseq>C7FF0A28EC276557</stoseq>
<rqsttkn1>USRT005 10142016</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>RTCDESC          </kwd>
<input>QRY RTCDESC SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
```

```
         scroll="no" len="8" dtype="CHAR" align="left" />
 <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
  len="8" dtype="CHAR" align="left" />
 <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
  len="4" dtype="CHAR" align="right" />
 <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
  scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
 <hdr slbl="PGM" llbl="LPgmName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
 <hdr slbl="INQ" llbl="LInq" scope="LCL" sort="n" key="0" scroll="yes"
  len="1" dtype="INT" align="left" />
 <hdr slbl="MDLN" llbl="LModelName" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 <hdr slbl="MDLT" llbl="LModelType" scope="LCL" sort="n" key="0"
  scroll="yes" len="4" dtype="CHAR" align="left" />
 <hdr slbl="DFLT" llbl="LDflt" scope="LCL" sort="n" key="0" scroll="yes"
  len="1" dtype="INT" align="left" />
 <hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
  scroll="yes" len="20" dtype="CHAR" align="left" />
 <hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 </cmdrsphdr>
 <cmdrspdata>
 <rsp>DESC(DBFDSRT1) MBR(IMS1    ) CC(   0)  PGM(         ) INQ(N)
  DFLT(Y) TMCR(2011.180 12:37:38.07) TMUP(                 )
 TMAC(2011.180 13:18:55.75) TMIM(                 ) DFNT(IMS     )
 </rsp>
 </cmdrspdata>
 </imsout>
```

Explanation: All routing code descriptors are returned with all output fields. All of
the routing code descriptor output fields do not fit on one screen, so the user must
scroll to the right for additional output fields. The routing code descriptor name,
the member name that built the line of output, and the completion code are
displayed on every screen. IMS-defined descriptor DBFDSRT1 contains the IMS
default routing code values.

Example 2 for QUERY RTCDESC command

TSO SPOC input:

```
QUERY RTCDESC NAME(*) SHOW(DEFN,PGM,INQ))
```

TSO SPOC output:

```
DescName MbrName   CC CCText                     Repo IMSid
*        IMS1      1D3 REPOSITORY MEMBER NOT FOUND Y
DBFDSRT1 IMS1       0                                  IMS1
DBFDSRT1 IMS2       0                                  IMS2
DBFDSRT1 IMS3       0                                  IMS3
```

```
(scrolled to the right screen 2)
DescName MbrName   CC Repo IMSid PgmName  LPgmName Inq LInq
*        IMS1      1D3 Y
DBFDSRT1 IMS1       0       IMS1                        N
DBFDSRT1 IMS2       0       IMS2                        N
DBFDSRT1 IMS3       0       IMS3                        N
```

OM API input:

```
CMD(QRY RTCDESC NAME(*) SHOW(DEFN,PGM,INQ))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.180 21:29:29.633544</statime>
<stotime>2011.180 21:29:29.744831</stotime>
<staseq>C7FF0C3844B08F1C</staseq>
<stoseq>C7FF0C385FDBFD40</stoseq>
<rqsttkn1>USRT005 10142929</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
<rsnmsg>CSLN023I</rsnmsg>
<rsntxt>At least one request was successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1     ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>RTCDESC         </kwd>
<input>QRY RTCDESC NAME(*) SHOW(DEFN,PGM,INQ) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="4" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
 len="4" dtype="CHAR" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RPGM" llbl="PgmName" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="PGM" llbl="LPgmName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="RINQ" llbl="Inq" scope="GBL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
<hdr slbl="INQ" llbl="LInq" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="INT" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DBFDSRT1) MBR(IMS3    ) CC(   0) PGM(        ) INQ(N)
 IMSID(IMS3)</rsp>
<rsp>DESC(*       ) MBR(IMS1    ) CC( 1D3) CCTXT(REPOSITORY MEMBER NOT
 FOUND) REPO(Y) </rsp>
<rsp>DESC(DBFDSRT1) MBR(IMS1    ) CC(   0) PGM(        ) INQ(N)
 IMSID(IMS1)</rsp>
```

```
<rsp>DESC(DBFDSRT1) MBR(IMS2   ) CC(  0) PGM(         ) INQ(N)
 IMSID(IMS2)</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The stored resource definitions and the runtime resource definitions for the specified resources are returned. There are no stored resource definitions in the repository for the routing code descriptors, so an error completion code is returned for the repository information.

Example 3 for QUERY RTCDESC command

TSO SPOC input:

```
QUERY RTCDESC NAME(RTCD*,NOTFN*) SHOW(EXPORTNEEDED)
```

TSO SPOC output:

```
DescName MbrName   CC CCText            ExportNeeded
NOTFN*   IMS1      10 NO RESOURCES FOUND
RTCD0001 IMS1       0                   N
RTCD0002 IMS1       0                   Y
```

OM API input:

```
CMD(QUERY RTCDESC SHOW(EXPORTNEEDED))
```

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

QUERY STRUCTURE command

Use the QUERY STRUCTURE command to display information about IMS coupling facility structures used by members of an IMSplex.

Subsections:
- "Environment"
- "Syntax" on page 609
- "Keywords" on page 609
- "Usage notes" on page 609
- "Output fields" on page 609
- "Return, reason, and completion codes" on page 610
- "Examples" on page 612

Environment

There are no environment indicators for the QUERY STRUCTURE command itself, because it does not run in the address space of any IMS control or dependent region. QUERY STRUCTURE is processed in an RM command processing environment.

Syntax

```
>>--+-QUERY-+--STRUCTURE--+-NAME(*)--------------------------+-->
    +-QRY---+             |         ,                        |
                         |        v                         |
                         +-NAME(--+-structurename--+--)-----+
                                  +-structurename*-+

>----+-----------------------------------------+------------><
     |            ,                            |
     |           v                            |
     +-SHOW(--+-ALL--------+--)---------------+
              +-STATISTICS-+
              +-TYPE-------+
```
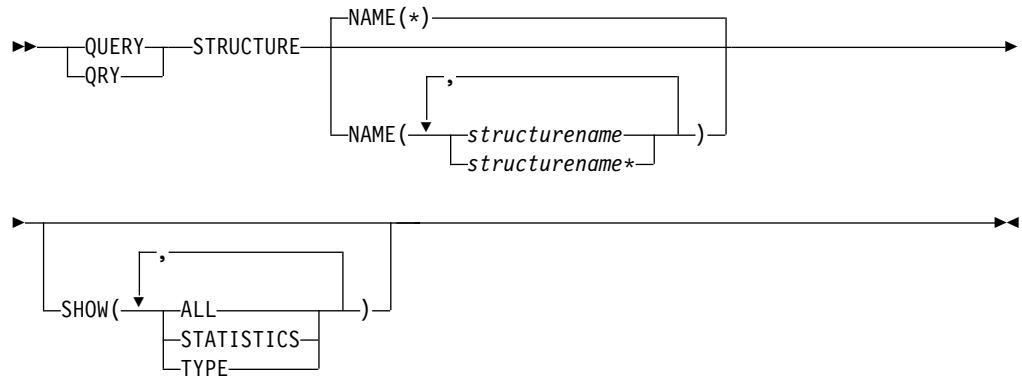
Keywords

The following keywords are valid for the QUERY STRUCTURE command:

NAME()
> Specifies the names of the structures for which information is to be returned. The structure name can be a generic parameter, to enable easy specification of a group of structures whose names match a generic parameter mask.

SHOW()
> Specifies the output fields to be returned. If SHOW is not specified, only the structure names, IMSplex member that builds the output line, and completion codes are returned. This provides a method for a system management application to obtain a list of all structure names. This can be used to determine the resource structure that is managed by RM. The parameters supported with the SHOW keyword are as follows:

> **ALL**
> > Returns all the output fields.

> **STATISTICS**
> > Displays statistics information for the structures that match the specification in the NAME() parameter.

> **TYPE**
> > Returns the type of the specified structure. For example, the type might be RSRC, which identifies a resource structure.

Usage notes

This command is supported only by Resource Manager (RM) to return information about a resource structure.

The command syntax for this command is defined in Extensible Markup Language (XML) and is available to automation programs which communicate with Operations Manager (OM).

Output fields

The following table contains information about the output fields for QUERY STRUCTURE. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 195. Output fields for the QUERY STRUCTURE command

Short label	Long label	Keyword	Meaning
STRNM	StructureName	N/A	Resource structure name. The structure name is always returned.
MBR	MbrName	N/A	IMSplex member that built the output line. The RM identifier of the RM that built the output line.
CC	CC	N/A	Completion code for the line of output. The completion code is always returned.
CCTXT	CCText	N/A	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
TYP	Type	TYPE	Structure type. RSRC indicates the RM resource structure.
LEA	LeAlloc	STATISTICS	Number of list entries that are allocated in the structure.
LEI	LeInuse	STATISTICS	Number of list entries in use in the structure.
ELMA	ElmAlloc	STATISTICS	Number of data elements that are allocated in the structure.
ELMI	ElmInuse	STATISTICS	Number of data elements in use in the structure.
RATIO	LE/EL	STATISTICS	Entry to element ratio. It is in the format of list entries/data elements.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY STRUCTURE command are standard for all commands entered through the OM API.

The following table contains the return and reason codes that can be returned to OM from a QUERY STRUCTURE command.

Table 196. Return and reason codes for the QUERY STRUCTURE command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY STRUCTURE command completed successfully.
X'00000010'	X'00000534'	Command did not complete because of a CQSQUERY buffer allocation failure.

Table 196. Return and reason codes for the QUERY STRUCTURE command (continued)

Return code	Reason code	Meaning
X'0300000C'	X'00003000'	The QUERY STRUCTURE command is successful for at least one resource name. The QUERY STRUCTURE command is not successful for one or more resource names. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the QUERY STRUCTURE command are listed in the QUERY STRUCTURE completion code table.
X'0300000C'	X'00003004'	No requests were successful. The resource names specified might be invalid, or there were no resources that match the filter specified.
X'03000014'	X'0000502C'	The QUERY STRUCTURE command processing terminated. RM was unable to obtain storage for the command output header.
X'03000014'	X'00005030'	The QUERY STRUCTURE command processing terminated. RM was unable to obtain storage for the command output response.
X'03000014'	X'00005200'	The QUERY STRUCTURE command processing terminated because of an unexpected Common Queue Server (CQS) error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains the completion codes that can be returned on a QUERY STRUCTURE command.

Table 197. Completion codes for the QUERY STRUCTURE command

Completion code	Completion code text	Meaning
0		The QUERY STRUCTURE command completed successfully for the resource.
4	NO RESOURCES FOUND	The structure name is unknown to the client that is processing the request. The structure name might have been typed in error, or the structure might not be defined or allocated at this time. If this is a wildcard request there were no matches for the name. Confirm the correct spelling of the structure name is specified on the command.
30	INVALID CHARACTER, RESOURCE NAME	Incorrect or unsupported characters are included in the structure name.
38	CQS UNEXPECTED ERROR	Command failed because of a CQS error.

Examples

The following is an example of the QUERY STRUCTURE command:

Example 1 for QUERY STRUCTURE command

This command displays all of the resource structures in the IMSplex and their statistics. Only one resource structure is defined, IMSRSRC01. The number of list entries allocated on the resource structure is 3577, the number of list entries in use on the structure is 676. The number of data elements allocated is 3574, the number of data elements in use is 24. The list entry to data element ratio on the resource structure is one to one. Not many list entries or data elements are in use on the resource structure, so the resource structure is not approaching full.

TSO SPOC input:
```
QRY STRUCTURE SHOW(STATISTICS)
```

TSO SPOC output:
```
StructureName   MbrName    CC LeAlloc LeInuse ElmAlloc ElmInuse LE/EL
IMSRSRC01       RM1RM       0    3577     676     3574       24 0001/0001
```

OM API input:
```
CMD (QRY STRUCTURE SHOW(STATISTICS))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1</xmlvsn>
<statime>2002.16314:31:34.901057</statime>
<stotime>2002.16314:31:34.941134</stotime>
<staseq>B7C49C9433D410C1</staseq>
<stoseq>B7C49C943D9CEC44</stoseq>
<rqsttkn1>USRT011 10073134</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>RM1RM   </master>
<userid>USRT011</userid>
<verb>QRY </verb>
<kwd>STRUCTURE</kwd>
<input>QUERY STRUCTURE SHOW(ALL)</input>
</cmd>
<cmdrsphdr>
<hdr slbl="STRNM" llbl="StructureName" scope="LCL" sort="A" key="1" scroll="NO"
len="16" dtype="CHAR" align="left"/>
<hdr slbl="MBR"   llbl="MbrName"  scope="LCL" sort="N" key="0" scroll="NO" len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"    llbl="CC"       scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="TYP"   llbl="Type"     scope="LCL" sort="N" key="0" scroll="YES" len="8"
dtype="CHAR" align="left" />
<hdr slbl="LEA"   llbl="LeAlloc"  scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="LEI"   llbl="LeInuse"  scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="ELMA"  llbl="ElmAlloc" scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT"  align="right" />
<hdr slbl="ELMI"  llbl="ElmInuse" scope="LCL" sort="N" key="0" scroll="YES" len="4"
```

```
dtype="INT"  align="right" />
<hdr slbl="RATIO" llbl="LE/EL"   scope="LCL" sort="N" key="0" scroll="YES" len="9"
dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>STRNM(IMSRSRC01) MBR(RM1RM) CC(   0) TYP(RSRC) LEA(3577) LEI( 676) ELMA(3574)
ELMI(  24) RATIO(0001/0001)</rsp>
</cmdrspdata>
</imsout>
```

Related concepts:

⬏ How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

⬏ Command keywords and their synonyms (Commands)

QUERY TRACE command

Use the QUERY TRACE command to display information about type-2 traces.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 614
- "Usage notes" on page 615
- "Output fields" on page 615
- "Return, reason, and completion codes" on page 616
- "Example for QUERY TRACE command" on page 616

Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which
you can use the QUERY TRACE commands and keywords.

Table 198. Valid environments for the QUERY TRACE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY TRACE	X	X	X
NAME	X	X	X
SHOW	X	X	X
TYPE	X	X	X

Syntax

Chapter 5. QUERY commands **613**
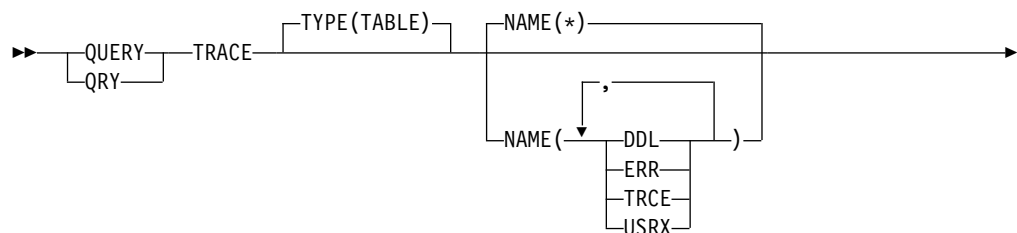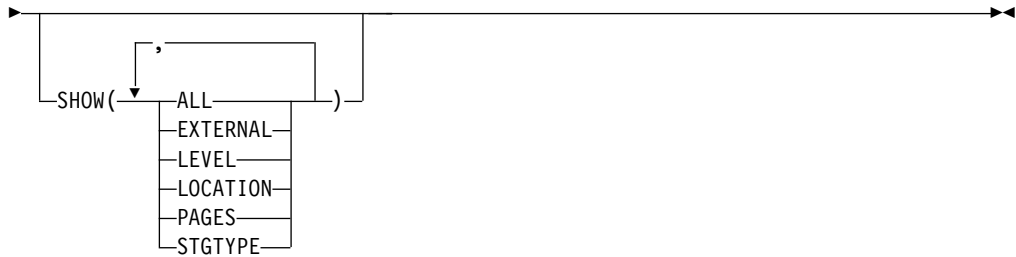
Keywords

The following keywords are valid for the QUERY TRACE command:

Type()
Specifies the trace type.

> **TABLE**
> Specifies a type-2 trace. This is the default value.

NAME()
Specifies the 1-4 character type-2 trace name. Wildcards can be specified in the name. The name is a repeatable parameter.

The following values are valid for the NAME keyword:

*   All traces are returned. This is the default value.

DDL   Data Definition Language (DDL) trace.

ERR   Error trace.

TRCE   Trace services trace.

USRX   User exit services trace.

SHOW()
Specifies the trace output fields to be returned. The trace name and trace type are always returned along with the name of the IMS system that created the output and the completion code. If SHOW is not specified, only the trace name, type, the name of the IMS that created the output, and the completion code are returned. You can specify the following parameters with the SHOW keyword, and you can specify the parameters in any order:

> **ALL**
> Returns all information for the trace that you specified.

> **EXTERNAL**
> Returns the state of external trace for the trace that you specified. Y indicates that external trace is active for the trace. N indicates that external trace is not active.

> **LEVEL**
> Returns the trace volume level for the trace that you specified.

> **LOCATION**
> Returns the location of the trace data. 31 indicates that the trace data is in 31-bit storage. 64 indicates that the trace data is in 64-bit storage.

> **PAGES**
> Returns the number of 4K pages that are allocated for the trace. Each page has a header and the space for the trace entries.

STGTYPE

Returns the storage type, either common or private, of the trace.

Usage notes

This command can be issued only through the Operations Manager API. This command applies to DB/DC, DBCTL and DCCTL systems.

The command output for this command is defined in Extensible Markup Language (XML) and is available to automation programs that communicate with Operations Manager (OM).

This command is routed to all IMS systems in an IMSplex as its default routing.

Output fields

The following table contains information about the output fields for QUERY TRACE. The columns in the table are as follows:

Short label

Contains the short label that is generated in the XML output.

Long label

Contains the long label that is generated in the XML output.

Keyword

Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning

Provides a brief description of the output field.

Table 199. Output fields for the QUERY TRACE command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	N/A	Completion code for the line of output. The completion code is always returned.
CCTXT	CCText	*error*	N/A	Completion code text that briefly explains the meaning of the nonzero completion code.
EXT	External	EXTERNAL	LCL	Indicates whether the trace is written to the external trace data set.
LEVEL	Level	LEVEL	LCL	Current volume level of the trace.
LOC	Location	LOCATION	LCL	Location of the trace.
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. Member name is always returned.
NAME	TraceName	NAME	LCL	Trace name.
PAGES	Pages	PAGES	LCL	Number of pages that are allocated for the trace.
STG	StgType	STGTYPE	LCL	Storage type of the trace.
TYPE	TraceType	TYPE	LCL	Type of the trace.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY TRACE command are standard for all commands that are entered through the OM API.

The following table contains the return and reason codes that can be returned to OM from a QUERY TRACE command.

Table 200. Return and reason codes for the QUERY TRACE command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY TRACE command completed successfully.
X'0000000C'	X'00003000'	The QUERY TRACE command was successful for at least one trace. The command failed for one or more traces. The completion code indicates the reason for the error. The completion codes that can be returned by the QUERY TRACE command are listed in the QUERY TRACE completion code table.
X'0000000C'	X'00003004'	The QUERY TRACE command failed for all traces that are specified. The completion code indicates the reason for the error. The completion codes that can be returned by the QUERY TRACE command are listed in the QUERY TRACE completion code table.
X'00000014'	X'00005004'	The QUERY TRACE command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The QUERY TRACE command processing terminated due to an internal error.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains the completion codes that can be returned on a QUERY TRACE command.

Table 201. Completion codes for the QUERY TRACE command

Completion code	Meaning
0	The QUERY TRACE command completed successfully for the trace.
10	No resources found. In this case, the resource is the trace name. None of the traces specified are known to the IMS that processed the command. The trace name might have been typed in error. Confirm that the correct spelling of the trace name is specified on the command.

Example for QUERY TRACE command

TSO SPOC input:

```
QRY TRACE TYPE(TABLE) NAME(DDL,USRX) SHOW(ALL)
```

TSO SPOC output:

```
Response for: QRY TRACE TYPE(TABLE) NAME(DDL,USRX) SHOW(ALL)
TraceName MbrName    CC TraceType Level  Pages Location StgType External
DDL       IMS1        0 Table     None      10       31 Common  N
```

```
DDL       IMS2      0 Table    None       8      31 Common  N
USRX      IMS1      0 Table    High       6      31 Common  N
USRX      IMS2      0 Table    None       6      31 Common  N
```

OM API input:

```
CMD(QRY TRACE TYPE(TABLE) NAME(DDL,USRX) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.7.0</omvsn>
<statime>2017.094 22:36:58.340816</statime>
<stotime>2017.094 22:36:58.354479</stotime>
<staseq>D256F32A269D0222</staseq>
<stoseq>D256F32A29F2F522</stoseq>
<rqsttkn1>USRT011 10153658</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>TRACE           </kwd>
<input>QRY TRACE TYPE(TABLE) NAME(DDL,USRX) SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="NAME" llbl="TraceName" scope="LCL" sort="a" key="1"
 scroll="no" len="4" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYPE" llbl="TraceType" scope="LCL" sort="n" key="0"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="LEVEL" llbl="Level" scope="LCL" key="0" len="6"
 dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="PAGES" llbl="Pages" scope="LCL" key="0" len="5"
 dtype="INT" align="right" skipb="yes" />
<hdr slbl="LOC" llbl="Location" scope="LCL" key="0" len="2"
 dtype="INT" align="right" skipb="yes" />
<hdr slbl="STG" llbl="StgType" scope="LCL" key="0" len="7"
 dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="EXT" llbl="External" scope="LCL" key="0" len="1"
 dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>NAME(USRX) MBR(IMS2) CC(   0) TYPE(Table   ) LEVEL(None  ) PAGES(
    6) LOC(   31) STG(Common ) EXT(N) </rsp>
<rsp>NAME(DDL ) MBR(IMS2) CC(   0) TYPE(Table   ) LEVEL(None  ) PAGES(
    8) LOC(   31) STG(Common ) EXT(N) </rsp>
<rsp>NAME(USRX) MBR(IMS1) CC(   0) TYPE(Table   ) LEVEL(High  )
 PAGES(    6) LOC(   31) STG(Common ) EXT(N) </rsp>
<rsp>NAME(DDL ) MBR(IMS1) CC(   0) TYPE(Table   ) LEVEL(None  ) PAGES(
   10) LOC(   31) STG(Common ) EXT(N) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The QUERY TRACE command was routed to IMS1 and IMS2. IMS1
and IMS2 both processed the command and returned information about the DDL
table trace and the USRX table trace.

QUERY TRAN command

Use the QUERY TRAN command to display information about transactions (for example, class, status, queue count, and others) across the IMSplex. This command can be specified only through the OM API and is valid on an XRF alternate.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 620
- "Usage notes" on page 631
- "Equivalent IMS type-1 commands" on page 632
- "Output fields" on page 632
- "Return, reason, and completion codes" on page 652
- "Examples" on page 655

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY TRAN command and keywords can be issued.

Table 202. Valid environments for the QUERY TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY TRAN	X		X
CLASS	X		X
CONV	X		X
FP	X		X
NAME	X		X
QCNT	X		X
REMOTE	X		X
RESP	X		X
SHOW	X		X
STATUS	X		X

Syntax

```
►───────────────────────────────────────────────────────────────►

         ┌─────────,─────────┐
   ┌─CLASS(─▼──class number──┴──)─────────────┐
   ├─QCNT(──┬─LT─┬──,number)───────────────────┤
   │        ├─LE─┤                              │
   │        ├─GT─┤                              │
   │        ├─GE─┤                              │
   │        ├─EQ─┤                              │
   │        └─NE─┘                              │
   │                        ┌──────,───────┐   │
   │                        ▼              │   │
   │         ┌─,─┐       ┌─CONV(──┬─N─┬──)─┤   │
   └─STATUS(─▼─AFFIN──┬──)┤       └─Y─┘    │   │
             ├─BAL────┤   ├─FP(──┬─E─┬──)──┤   │
             ├─CONV───┤   │      ├─N─┤     │   │
             ├─CPIC───┤   │      └─P─┘     │   │
             ├─DYN────┤   ├─REMOTE(──┬─N─┬──)─┤ │
             ├─IOPREV─┤   │          └─Y─┘    │ │
             ├─LCK────┤   └─RESP(──┬─N─┬──)───┘ │
             ├─NOTINIT┤            └─Y─┘        │
             ├─QERR───┤
             ├─QSTP───┤
             ├─SUSPEND┤
             ├─STOQ───┤
             ├─STOSCHD┤
             ├─TRACE──┤
             └─USTO───┘
```

```
                                    ┌─────,─────────┐
►─────────────────SHOW(──▼──┬─AFFIN─────┬──)──────────────────────────────────────►◄
                            ├─ALL───────┤
                            ├─AOCMD─────┤
                            ├─CLASS─────┤
                            ├─CMTMODE───┤
                            ├─CONV──────┤
                            ├─CPRI──────┤
                            ├─DCLWA─────┤
                            ├─DEFN──────┤
                            ├─DEFNTYPE──┤
                            ├─DIRROUTE──┤
                            ├─EDITRTN───┤
                            ├─EDITUC────┤
                            ├─EMHBSZ────┤
                            ├─EXPRTIME──┤
                            ├─FP────────┤
                            ├─GLOBAL────┤
                            ├─IMSID─────┤
                            ├─INQ───────┤
                            ├─LCT───────┤
                            ├─LOCAL─────┤
                            ├─LPRI──────┤
                            ├─MAXRGN────┤
                            ├─MODEL─────┤
                            ├─MSGTYPE───┤
                            ├─MSNAME────┤
                            ├─NPRI──────┤
                            ├─PARLIM────┤
                            ├─PGM───────┤
                            ├─PLCT──────┤
                            ├─PLCTTIME──┤
                            ├─PSB───────┤
                            ├─QCNT──────┤
                            ├─RECOVER───┤
                            ├─REMOTE────┤
                            ├─RESP──────┤
                            ├─RGC───────┤
                            ├─SEGNO─────┤
                            ├─SEGSZ─────┤
                            ├─SERIAL────┤
                            ├─SPASZ─────┤
                            ├─SPATRUNC──┤
                            ├─STATUS────┤
                            ├─TIMESTAMP─┤
                            ├─TRANSTAT──┤
                            └─WFI───────┘
     ─SHOW(WORK)────────────────────
     ─SHOW(EXPORTNEEDED)────────────
```

Keywords

The following keywords are valid for the QUERY TRAN command:

CLASS()
Displays transactions that possess at least one of the specified classes. This

keyword allows for additional filtering by CLASS value. If no filter is specified (such as STATUS, CLASS or QCNT), all the transactions matching the transaction name are returned.

The output returned when the CLASS filter is specified includes the class value of the transaction that caused the transaction name to be displayed even if the SHOW(CLASS) option is not specified.

CONV()
Selects transactions for display that possess the conversational attributes specified.

FP()
Selects transactions for display that possess the Fast Path option specified. If more than one FP option is specified, selects transactions for display that possess at least one of the Fast Path options specified.

NAME()
Specifies the 1-8 character name of the transaction. Wildcards can be specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all transactions. If no filter is specified (such as STATUS, CLASS or QCNT), all the transactions matching the transaction name are returned.

QCNT()
Selects transactions that have a queue count less than (LT), less than or equal to (LE), greater than (GT), greater than or equal to (GE), equal to (EQ), or not equal to (NE) the specified numbers. The specified number cannot be a 1 when LT is specified. This keyword allows for additional filtering by QCNT value.

Transactions with a queue count of 0 are not returned when the QCNT filter is specified. When a filter of QCNT(LT,n) is specified, transactions with a queue count greater than 0 and less than n are returned. If no filter is specified (such as STATUS, CLASS or QCNT), all the transactions matching the transaction name are returned.

The QCNT filter is valid in both a shared-queues environment and a non-shared-queues environment.

In a shared-queues environment, if QCNT is specified, the performance implication is that the shared queues are read. In this environment, the QUERY TRAN QCNT command is processed only by the master IMS as the queues are global. The command master returns all the transactions on the shared queues that match the queue count filter specified. If QCNT is specified with a wildcard transaction name, the performance implication is that all the shared-queues transaction messages on the Coupling Facility must be read.

In a non-shared-queues environment, the local queue count values are used to determine the transactions to be displayed. In this environment, the QUERY TRAN QCNT command is processed by each IMS the command is routed to as the queues are local. Each IMS returns all the transactions it found locally that match the queue count filter specified.

The output returned when the QCNT filter is specified includes the queue count of the transaction that caused the transaction name to be displayed even if the SHOW(QCNT) option is not specified.

REMOTE()
Selects transactions for display that possess the remote option specified.

RESP()
Selects transactions for display that possess the response mode option specified.

SHOW()

Specifies the transaction output fields to be returned. The transaction name is always returned along with the name of the IMS that created the output and the completion code. If SHOW is not specified, only the transaction names are returned if the QCNT, CLASS, or STATUS filter is not specified. This keyword provides a method for a system management application to obtain a list of transactions that match the transaction names that are currently known in the IMSplex.

Important: The only SHOW option supported when the QCNT() filter is specified is the AFFIN option. No other SHOW options are supported with the QCNT() filter because of performance reasons.

AFFIN

Transaction has local affinity to the IMS.

If the AFFIN option is used as a status filter, IMS shows all the transactions that have affinity status.

If the SHOW(ALL) or SHOW(STATUS) keyword is specified, and if a transaction has affinity status, AFFIN is displayed in the LclStat column.

If the SHOW(AFFIN) keyword is used with the QCNT filter, messages on the shared queue are displayed with their AFFIN value.

The affinity that is shown when the SHOW(AFFIN) keyword is specified is valid only with the QCNT filter and is ignored for the other variations of the QUERY TRAN command.

The AFFIN option is valid only in a shared-queues environment and is ignored in a non-shared-queues environment.

ALL

Returns all information about the transaction itself. Other SHOW keywords can be specified to return information about resources that are related to the transaction.

Global values are returned only for those status fields and attributes for which global information is kept for the IMSplex.

Restriction: The ExportNeeded column, which is returned when SHOW(EXPORTNEEDED) is specified, is not returned with SHOW(ALL).

AOCMD

Specifies that you want the AOI option returned which indicates whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call.

CLASS

Scheduling class used to determine which message regions can process the transaction locally on a particular IMS.

CMTMODE

Specifies when database updates and non-express output messages are committed. This operand affects emergency restart.

CONV

Conversation option.

CPRI

The current priority. The current priority is the normal priority, when the

transaction queue count is less than the limit count. The current priority is raised to the limit priority if the transaction queue count is equal to, or exceeds, the limit cover.

DCLWA

Log write-ahead option.

DEFN

Specifies that the resource definitions are to be returned.

The transaction attributes that can be returned are: AOCMD, CLASS, CMTMODE, CONV, DCLWA, DIRROUTE, EDITUC, EDITRTN, EMHBSZ, EXPRTIME, FP, INQ, LCT, LPRI, MAXRGN, MSGTYPE, NPRI, PLCT, PLCTTIME, PARLIM, PGM, RECOVER, REMOTE, RESP, SERIAL, SIDR, SIDL, SEGNO, SEGSZ, SPASZ, SPATRUNC, TRANSTAT, WFI, the repository create and update time stamps, the IMS runtime create, update, import and access time stamps, and the IMS runtime MSNAME.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition that is obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE, MODEL, QCNT, STATUS, or WORK.
- The LclStat, LModelName, LModelType, and LDefnType columns, which are returned on the QRY TRAN SHOW(ALL) command, are not returned with SHOW(DEFN).
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).
- When querying transaction information from the repository, the SHOW(DEFN) filter is not supported when used with either the QCNT

or STATUS filter. The runtime filters of QCNT and STATUS are not valid with SHOW(DEFN), SHOW(DEFN,GLOBAL), SHOW(DEFN,LOCAL), SHOW(DEFN,IMSID), SHOW(DEFN,IMSID,GLOBAL) or SHOW(DEFN,IMSID,LOCAL).

Resource definitions that are stored in the repository are used to determine the response lines with the repository information, and the runtime resource definitions are used to determine the response lines with the IMS runtime resource information. The response lines are returned for each stored resource or runtime resource definition that matches the specified filter. If SHOW(DEFN,GLOBAL) is specified, only the stored resource definitions that match the specified filter are returned. If SHOW(DEFN,LOCAL) is specified, only the runtime resource definitions that match the specified filter are returned.

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

DEFNTYPE
Definition type, which describes how the resource was defined.

DIRROUTE
MSC directed routing option.

EDITRTN
Input edit routine that edits messages before the program receives the message.

EDITUC
Uppercase translation option of the input data.

EMHBSZ
EMH buffer size that is required to run the Fast Path transaction.

EXPORTNEEDED
Specifies that only the transaction resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command.

If the NAME keyword is omitted or NAME(*) is specified, only the names of the transactions that need to be exported to the repository are returned.

If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each transaction name specified on the command. If the specified transaction does not need to be exported to the repository, an N is returned in the ExportNeeded column.

Restrictions:
- SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords.
- SHOW(EXPORTNEEDED) is valid only when IMS is enabled with the IMSRSC repository.
- The QUERY TRAN SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

Tip: For resources and descriptors that have not been exported to the IMSRSC repository, you can issue the EXPORT DEFN TARGET(REPO) OPTION(CHANGESONLY) command to write the resource and descriptor definitions to the repository so that they are available for the next IMS cold start.

EXPRTIME
Transaction expiration time in seconds.

FP   Fast Path option.

GLOBAL
For output fields that have both local and global values, this option returns only the global values. If used with another SHOW keyword to request a specific output field, this OPTION requests that only the global value of the specified output field is returned. IMS retrieves global information from the source that maintains the data. For example, RM maintains some global information, and CQS maintains other global information. Global output is returned only by the command master. The QUERY TRAN command can return global information from the RM resource structure if global status is maintained for transactions.

The QUERY TRAN command returns global information if you specified that the IMSplex maintain global transaction status in RM. You can specify this during IMS initialization in either the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSTRAN(Y)). You can also change it dynamically using the UPD IMS SET(PLEXPARM(GSTSTRAN(Y)) command. If you do not specify that global database status is to be maintained, the GLOBAL keyword is processed, and the global status is not updated.

If SHOW(GLOBAL) is specified without SHOW(DEFN), the repository information is not returned. SHOW(GLOBAL) returns any global status information from the resource structure if GSTSTRAN=Y is enabled. It also returns the global queue count from IMS shared queues. If SHOW(GLOBAL,DEFN) is specified, the global resource definitions from the repository are returned by the command master IMS. SHOW(GLOBAL,DEFN) is valid only when the command master IMS is enabled to use the repository.

IMSID
Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only when the command master IMS is enabled to use the repository.

When SHOW(IMSID) is specified with the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

SHOW(IMSID) cannot be specified with any other SHOW filters other than DEFN and GLOBAL. If SHOW(IMSID,GLOBAL) is specified, GLOBAL is ignored; that is, SHOW(IMSID,GLOBAL) is treated as SHOW(IMSID). SHOW(DEFN,IMSID,LOCAL) is treated as SHOW(DEFN,LOCAL).

INQ
Inquiry option.

LCT

Limit count. Specifies the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction. The limit count value can range from 1 through 65 535.

The limit count value is ignored for a transaction that is processed by a BMP.

The limit count value is ignored in a shared-queues environment.

LOCAL

For output fields that have both local and global values, this option returns only the local values. If used with another SHOW keyword to request a specific output field, this option requests that only the local value of the specified output field is returned. Local output is returned by each IMS that processes the command.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

LPRI

Limit priority. The scheduling priority to which this transaction is raised when the numbers of input transactions enqueued and waiting to be processes is equal to or greater than the limit count value. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class.

The limit priority value is ignored for a transaction that is processed by a BMP.

MAXRGN

Maximum region count. This limits the number of message processing program (MPP) regions that can be concurrently schedule to process a transaction. When the number of MPP regions is not limited, one transaction might monopolize all available regions. MAXRGN(0) means that no limit is imposed.

MODEL

The model name and model type used to create this resource. The model name and model type are blank for IMS-defined resources and descriptors and queue-only transactions created by the DFSINSX0 exit. The CREATE command specified without the LIKE keyword creates a resource using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSTR1 or user-defined. The CREATE command specified with the LIKE keyword creates a resource using a model. The resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource. The model name is blank if the resource is IMS-defined.

MSGTYPE

Message segment type (single or multiple segment). It specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing.

If MSC-directed routing is used in a multiple IMS system configuration, IMS does not ensure that both the message and the transaction destined to process that message are either single segment or multiple segments.

MSNAME

The logical link path name, remote system ID, and local system ID are returned.

Logical link path name in a multiple IMS system configuration (MSC). A logical link path is a path between any two IMS systems. The IMS systems are identified by the remote system ID and the local system ID associated with the logical link path. The remote system ID identifies the system in which messages using this path are to be processed. The local system ID identifies this system.

The remote system ID (SIDR) identifies the IMS system on which the application program runs. A value of 0 means MSC is not enabled on this system. The local system ID and remote system ID are the same for local transactions.

The local system ID (SIDL) identifies the originating system to which responses are returned. A value of 0 means MSC is not enabled on this system. The local system ID and remote system ID are the same for local transactions.

NPRI

Normal scheduling priority. The scheduling priority is an attribute that is used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The normal priority is assigned to the transaction as the scheduling priority when the number of input transactions that are enqueued and waiting to be processed is less than the limit count value.

The normal priority value is ignored for a transaction that is processed by a BMP.

PARLIM

Parallel processing limit count. This value is the maximum number of messages that can currently be queued, but not yet processed, by each active message region that is currently scheduled for this transaction. This value is the threshold value to be used when the associated program is defined with a scheduling type of parallel. An additional region is scheduled whenever the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions that are currently scheduled for this transaction.

PARLIM(0) indicates that any input message can cause a new region to be scheduled. PARLIM(65535) indicates that parallel processing is disabled and IMS allows the transaction to be scheduled in only one region at a time.

PGM

The name of the program that is associated with this transaction. This name matches the PSB name in ACBLIB. SHOW(PSB) is an alias of SHOW(PGM).

PLCT

Processing limit count. This value is the maximum number of messages that are sent to the application program by the IMS for processing without reloading the application program.

PLCT(0) means that a maximum of one message is sent to the application program at a single program load.

PLCT(65535) means that no limit is placed on the number of messages that are processed at a single program load.

PLCTTIME

Processing limit count time. This value is the amount of time (in hundredths of seconds) allowable to process a single transaction (or message). The number specifies the maximum CPU time allowed for each message to be processed in the message processing region.

For the Fast Path potential transactions that are defined in the system definition TRANSACT macro, the value shown is the CPU-time-per-transaction number, which is specified on the PROCLIM keyword, multiplied by 100.

PLCTTIME(6553500) means that no time limit is placed on the application program.

PSB

The name of the program that is associated with this transaction. This name matches the PSB name in ACBLIB. SHOW(PSB) is an alias of SHOW(PGM).

QCNT

Transaction message queue count.

In a non-shared-queues environment, the local transaction message queue count is returned. This count represents the number of messages on the IMS system where the command is issued.

In a shared-queues environment, both the local transaction message queue count and the global transaction message queue count are returned.

- The local queue counts value that is returned in this environment represents the messages being processed by the IMS system where this command is issued. Do not use local queue counts for queue evaluation purposes.
- The global transaction message queue count represents the current message queue count on the shared queues structure. This count represents the number of messages that can be processed by the IMS system where the command is issued. This count includes messages that can be processed by any IMS system (messages with no affinity), plus messages that can be processed by the IMS system where the command is issued (messages with an affinity to the IMS system that issued the command).

RECOVER

Recovery option.

REMOTE

Remote option.

RESP

Response option.

RGC

Number of regions the transaction is currently scheduled in the local IMS.

SEGNO

Segment number. This value is the maximum number of application program output segments that are allowed into the message queues per Get Unique (GU) call from the application program.

SEGNO(0) means that the number of segments is not checked by the online system at the execution time.

SEGSZ

Segment size. This value is the maximum number of bytes allowed in any one output segment.

SEGSZ(0) means that the segment size is not checked by the online system at execution time.

SERIAL

Serial option.

SPASZ

Scratchpad area size for a conversational transaction.

SPATRUNC

The SPA data truncation option indicates whether the SPA data should be truncated or preserved across a program switch to a transaction that is defined with a smaller SPA.

When a conversation initially starts, and when a program is switched, the SPATRUNC option is checked and set or reset as specified. When the option is set, it remains set for the life of the conversation, or until a program switch occurs to a transaction that specifies the option is to be reset.

When a program switch occurs, the truncated data option for the new transaction is first checked, and that specification is set for the conversation and is used for the SPA inserted into the output message. If the option is not specified for the new transaction, the option currently in effect for the conversation is used.

STATUS

Local transaction status. For a description of the possible transaction status returned, see the STATUS keyword in the "Output fields for the QUERY TRAN command" table under "Output fields" on page 632.

TIMESTAMP

The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:

- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

TRANSTAT

Transaction level statistics option.

WFI
Wait-for-input option.

WORK
Work in progress for the transaction specified on the NAME parameter and its associated resources. The QUERY TRAN SHOW(WORK) command can be issued before a DELETE, IMPORT or UPDATE command to check for any work in progress for the specified transaction and any of its associated resources. Any work in progress might cause the subsequent DELETE, IMPORT or UPDATE commands to fail. If no work is in progress for the specified resource, a response line is returned with a work status of blanks.

Notes:
1. SHOW(WORK) specified with NAME(*) might have a performance impact on the processing of the command.
2. You cannot specify this filter with other SHOW filters; you must specify SHOW(WORK) individually.
3. The QRY TRAN SHOW(WORK) command is not valid on an XRF alternate.

STATUS()
Selects transactions for display that possess at least one of the specified transaction statuses. This keyword allows for additional filtering by transaction status. If a STATUS, CLASS, or QCNT filter is not specified, all the transactions matching the transaction name are returned.

The output returned when the STATUS filter is specified includes the status of the transaction that caused the transaction name to be displayed even if the SHOW(STATUS) option is not specified.

Filtering is based only on local status, even if RM maintains global status.

AFFIN
Transaction is registered with local affinity with IMS.

BAL
Transaction is eligible for load balancing (for example, with parallel limits specified).

CPIC
This CPI-C transaction was built dynamically on this IMS system and can process only on this IMS system.

DYN
Transaction was built in a shared-queues environment, is not defined to this IMS, and therefore, cannot be scheduled in this IMS subsystem.

IOPREV
Indicates that a BMP program containing GSAM cannot complete scheduling because I/O prevention has not completed. Further I/O requests to data sets are inhibited.

LCK
Transaction is locked by a /LOCK TRAN command.

NOTINIT
Transaction is not initialized and cannot be used.

QERR
I/O error has occurred on this queue for this MSC remote transaction.

QSTP
Transaction queuing is stopped by online change because the transaction is affected by the online change. Online change might be changing or deleting the transaction, or changing or deleting a program, PSB, database, or DMB referenced by the transaction. Transaction queuing is stopped until the online change is committed or aborted.

SUSPEND
Transaction has messages on the suspend queue.

STOQ
Transaction is stopped for queuing and can no longer be queued globally. This status might be caused by a previous UPDATE TRAN, /PURGE TRAN or /STO TRAN command.

STOSCHD
Transaction is stopped for scheduling and can no longer be scheduled globally. This status might be caused by a previous UPDATE TRAN, /PSTOP TRAN or /STO TRAN command or an application abend.

TRACE
Transaction is being traced.

USTO
Transaction scheduling stopped because of unavailable data.

Usage notes

The transaction information displayed depends on whether the IMS issuing the QUERY TRAN command is running with RM services. If the QUERY TRAN command is issued by an IMS command master running without RM, all transaction information local to that IMS is returned. If the QUERY TRAN command is issued by an IMS command master running with RM, the IMS command master retrieves global information from CQS or RM as specified.

If the QUERY TRAN command is routed for global information to all IMS systems in an environment where some IMS systems use RM services and other IMS systems do not, the command results will vary because any of the IMS systems can be the command master. The RM environment of the IMS command master affects the type of transaction information that is displayed. You might want to route QUERY TRAN to specific IMS systems if some IMS systems are using RM. Here are two examples of why you might receive different command results:

1. The IMSplex has non-cloned systems and the transaction is only defined by an IMS that has RMENV=N. The IMS command master is running with RM services, but because the transaction is not defined to RM, no global information is obtained. The results are two response lines:
   - The IMS command master returns the transaction name as invalid.
   - The IMS with RMENV=N returns its local information.
2. The IMSplex has non-cloned systems. If global queue counts are requested, and the IMS command master does not have RM running, no global queue counts are returned for transactions that are not defined locally at the command master. All other IMS systems return only their local information.

In a shared-queues environment, a QUERY TRAN NAME(*tranname*) QCNT(GT,0) command returns the total count of messages on the shared queue structure for transaction *tranname*. This count includes messages that can be processed by any IMS (messages with no affinity), plus messages with affinity to an IMS. A QUERY

TRAN NAME(*tranname*) QCNT(GT,0) SHOW(AFFIN) command returns one line
for the transaction name with messages that can be processed by any IMS
(messages with no affinity). Another line is returned for the transaction that has
messages queued with affinity to an IMS. This line shows the number of messages
that are queued with affinity, as well as the IMSID of the IMS to which they have
affinity. A QUERY TRAN NAME(*tranname*) QCNT(GT,0) or QUERY TRAN
NAME(*tranname*) QCNT(GT,0) SHOW(AFFIN) command is processed only by the
command master IMS because the command master IMS can obtain information
for all IMS subsystems in the IMSplex environment.

If you want to display information about resource definitions, specify
SHOW(DEFN). If you want to know which IMS systems have the resource defined
and also know the attributes or resource definitions at each IMS system, specify
SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource
defined, specify SHOW(IMSID).

Equivalent IMS type-1 commands

The following table shows variations of the QUERY TRAN command and the
type-1 IMS commands that display similar information.

Table 203. Type-1 equivalents for the QUERY TRAN command

QUERY TRAN command	Similar IMS type-1 command
QUERY TRAN NAME(tranname) SHOW(ALL)	/DISPLAY TRAN tranname
QUERY TRAN SHOW(ALL)	/DISPLAY TRAN ALL
QUERY TRAN NAME(tranname) SHOW(QCNT)	/DISPLAY TRAN tranname QCNT
QUERY TRAN NAME(tranname) STATUS(IOPREV,LCK,QERR,SUSPEND,STOQ, STOSCHD,USTO) SHOW(STATUS)	/DISPLAY STATUS TRANSACTION

Output fields

The following table shows the output fields for the QUERY TRAN command. The
columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies the keyword on the command that caused the field to be
> generated. N/A appears for output fields that are always returned. *error*
> appears for output fields that can appear for a nonzero completion code.

Scope Identifies the scope of the output field.

Meaning
> Provides a brief description of the output field.

Table 204. Output fields for the QUERY TRAN command

Short label	Long label	Keyword	Scope	Meaning
AFIN	Affinity	AFFIN	GBL	Affinity of the transaction messages on the shared queues, or affinity registration of the transactions for this IMS. For a transaction message on the shared queues (that is, the QCNT option), AFFIN displays the IMS ID or the recoverable service element (RSE) name of the IMS system that the message can be processed on. For a transaction registration, AFFIN displays the IMS ID or the RSE name appended to the transaction that was registered to CQS with local affinity.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
AOCMD	LAOCMD	AOCMD	LCL	Transaction supports AOI CMD calls (CMD, TRAN, or Y) or not (N). The output value is obtained from the local IMS.
				N Indicates that the transaction is not permitted to issue AOI type-1 CMD calls. The transaction is permitted to issue AOI type-2 ICMD calls.
				CMD Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, authorization checking is based on which transactions can issue a particular command. In this case, the commands (or first three characters of the commands) need to be defined to RACF or equivalent product as a user. The type-1 AOI transactions must be defined as profiles under the TIMS class, and for each transaction, the commands it can issue must be specified. Defining AOCMD(CMD) requires you to create fewer user IDs than you need to create for the AOCMD(TRAN) definition. However, defining AOCMD(CMD) requires you to create or modify a larger number of resource profiles.
				TRAN Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the transaction code is used for authorization. The first authorization check results in the accessor environment element (ACEE) being built. This environment is kept for use by future authorization checks. The type-1 AOI transaction must be defined to RACF or equivalent product as a user. The transactions will then be specified on RACF PERMIT statements for each command they are allowed to issue from a type-1 AOI transaction. Specifying AOI transactions as users to RACF might conflict with the name of a user already defined to RACF. If this conflict occurs, then either the transaction name or the existing user name needs to be changed.
				Y Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the user ID or program name is used for authorization. For some environments, if a Get Unique call has not yet occurred, the program name is used for authorization.
CC	CC	N/A	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned. Refer to Table 208 on page 655 for more information.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the nonzero completion code.
CMTM	LCmtMode	CMTMODE	LCL	Commit mode for the transaction: commit after a single message (SNGL) or multiple messages (MULT). The output value is obtained from the local IMS.
				MULT Database updates and non-express output messages are committed only when the application program terminates normally, when the processing limit count has been reached, or, in the case of a pseudo-WFI dependent region, when there are no more messages on the queue. For example, if five transactions are processed during a single schedule of a program, all five are committed only when the fifth one is completed and the program terminates. Until a transaction has been committed, locks for updated database records are not released and non-express output messages are not queued for output. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing.
				SNGL Database updates and non-express output messages are committed when the application program completes processing each transaction. IMS invokes commit processing either when the application program requests the next message (issues a GU to the IO-PCB), or when the application program terminates. If an application ends abnormally before committing its message, emergency restart requeues the message that was in process before the abend and makes it available for reprocessing.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
CONV	LConv	CONV	LCL	Conversation option. Transaction is conversational (Y), or not (N). The output value is obtained from the local IMS.
				N Transaction is not conversational.
				Y Transaction is conversational. The transaction message is destined for a conversational program. A conversational program processes transactions made up of several steps. A conversational program receives a message from a terminal, replies to the terminal, but saves the data from the transaction in a scratchpad area (SPA). When the person at the terminal enters more data, the program has the data it saved from the last message in the SPA, so it can continue processing the request without the person at the terminal having to enter the data again. **Restriction:** The method to filter on a conversational transaction has changed between IMS Version 9 and IMS Version 10. In IMS Version 9, the command is QUERY TRAN STATUS(CONV). In IMS Version 10 or later, the command is QUERY TRAN CONV(Y). If IMS is running in an IMSplex with IMS Version 10 or later and either IMS Version 8 or IMS Version 9 systems, and the command is to be processed on all systems, the command must be issued as QUERY TRAN STATUS(CONV) CONV(Y). The IMS Version 10 system or later processes the CONV(Y) filter, while the IMS Version 8 and IMS Version 9 systems process the STATUS(CONV) filter. If a command is issued to an IMS Version 10 system or later with STATUS(CONV) without CONV(Y), the command is rejected.
CONVID	LConvID	WORK	LCL	Conversation ID for transaction that has a conversation in progress.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
DCLW	LDCLWA	DCLWA	LCL	Perform log write-ahead for recoverable, nonresponse mode input messages and transaction output messages (Y) or not (N). The output value is obtained from the local IMS.
				N IMS does not perform log write-ahead.
				Y IMS performs log write-ahead for recoverable, nonresponse input messages and transaction output messages. If not defined for the transaction, the default is the DCLWA parameter in the IMSCTRL macro. This ensures that a nonresponse input transaction is made recoverable across IMS failures before IMS acknowledges receipt of the input.
				Database changes are made recoverable before IMS sends associated output reply messages.
				This ensures that information in the log buffers is written to the IMS log, before the associated input acknowledgment or output reply is sent to the terminal.
DFNT	LDefnType	DEFNTYPE	LCL	Definition type, which can be one of the following:
				CPIC A CPI-C transaction that was dynamically built on this IMS system.
				CREATE Defined by a CREATE command.
				DFSINSX0 Defined by user exit DFSINSX0. The resource can be exported only if the export option was set. This does not apply to descriptors.
				IMPORT Defined by the IMPORT command.
				IMS Defined by IMS.
				MODBLKS Defined by system definition in the MODBLKS data set.
				UPDATE Defined by system definition in the MODBLKS data set, but changed into a dynamic resource by the UPDATE command. This does not apply to descriptors.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
DRRT	LDirRoute	DIRROUTE	LCL	Supports MSC directed routing (Y) or not (N). The output value is obtained from the local IMS.
				N The application program processing a transaction is not informed of the system which originated the transaction. The name of the originating LTERM is placed in the I/O PCB.
				Y The application program processing a transaction is informed of the system which originated the transaction, if MSC directed routing is used in a multiple IMS system configuration. An MSNAME corresponding to a logical path back to the originating system is placed in the I/O PCB.
EDTR	LEditRtn	EDITRTN	LCL	Input edit routine name.
EDTT	LEditUC	EDITUC	LCL	Input data is to be translated to uppercase (Y) or not (N). The output value is obtained from the local IMS.
				N Input data is not translated to uppercase. It can consist of uppercase and lowercase characters as entered from the terminal.
				Y Input data is to be translated to uppercase before it is presented to the processing program. If FP(E), the transaction is to be translated to uppercase before being presented to the edit/routing exit routine.
				Specifying EDITUC(Y) for VTAM terminals prevents the transmission of embedded device control characters.
EMHBS	LEMHBSz	EMHBSZ	LCL	EMH buffer size. The output value is obtained from the local IMS.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the transaction has been exported to the IMSRSC repository. The value can be Y or N.
EXPRT	LExprTm	EXPRTIME	LCL	Transaction expiration time. The output value is obtained from the local IMS.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
FP	LFP	FP	LCL	Fast Path potential candidate (P), Fast Path exclusive (E), or FP option not enabled (N). The output value is obtained from the local IMS.
				E Fast Path exclusive transaction. Any message for this transaction is always routed to a Fast Path application program.
				N Fast Path option is not enabled. When FP(N) is specified, any attempt to use Fast Path resources or commands will yield unpredictable results.
				P Fast Path potential transaction. Any message for this transaction can potentially be routed to a Fast Path application program. **Restriction:** The method to filter on a Fast Path transaction has changed between IMS Version 9 and IMS Version 10. In Version 9, the command is QUERY TRAN STATUS(FPE), QUERY TRAN STATUS(FPP) or QUERY TRAN STATUS(FPE,FPP). In IMS Version 10 or later, the command is QUERY TRAN FP(E), QUERY TRAN FP(P), or QUERY TRAN FP(E,P). If IMS is running in an IMSplex with IMS Version 10 or later and either IMS Version 8 or IMS Version 9 systems, and the command is to be processed on all systems, the command must be issued as QUERY TRAN STATUS(FPE) FP(E), QUERY TRAN STATUS(FPP) FP(P), or QUERY TRAN STATUS(FPE,FPP) FP(E,P). The IMS Version 10 system or later processes the FP() filter, while the IMS Version 8 and IMS Version 9 systems process the STATUS() filter. If a command is issued to an IMS Version 10 system or later with STATUS(FPE), STATUS(FPP), or STATUS(FPE,FPP) without FP(), the command is rejected.
IMSID	IMSid	IMSID	GBL	Returns the IMSIDs that have the resource defined. The output value is obtained from the repository.
INQ	LInq	INQ	LCL	Inquiry transaction (Y) or not (N). The output value is obtained from the local IMS.
				N Inquiry option is disabled.
				Y Inquiry option is enabled. This is an inquiry transaction that, when entered, does not cause a change in any database. Programs are prohibited from issuing ISRT, DLET, or REPL calls to a database when scheduled to process a transaction defined as INQ(Y). An application program cannot do an SQL INSERT, DELETE, or UPDATE when the IMS transaction is defined with INQ(Y).
LCLS	LCls	CLASS	LCL	Scheduling class used to determine which message regions can process the transaction locally on a particular IMS.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
LCP	LCPRI	CPRI	LCL	Local current scheduling priority. The current scheduling priority is used to calculate which transaction is selected for scheduling.
LLCT	LLCT	LCT	LCL	Limit count in the local IMS. The limit count is the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction.
LLP	LLPRI	LPRI	LCL	Local limit scheduling priority. The limit scheduling priority is the priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value.
LMRG	LMaxRgn	MAXRGN	LCL	Local maximum region count. The maximum region count is the maximum number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction that is eligible for parallel scheduling.
LNP	LNPRI	NPRI	LCL	Local normal scheduling priority. The normal scheduling priority is the priority assigned to this transaction when the number of input transactions enqueued and waiting to be processed is less than the limit count value.
LPLCT	LPLCT	PLCT	LCL	Local processing limit count. The processing limit count is the number of transaction messages a program can process in a single scheduling.
LPLM	LParLim	PARLIM	LCL	Local parallel processing limit count. The parallel limit count is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. An additional message region is scheduled whenever the transaction queue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.
LQ	LQCnt	QCNT	LCL	Local transaction message queue count.
LSNO	LSegNo	SEGNO	LCL	Local application program output segment limit allowed in message queues for each GU call.
LSSZ	LSegSz	SEGSZ	LCL	Local application program output segment size limit allowed in the message queues for each GU call.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
LSTT	LclStat	STATUS	LCL	Local transaction status.

	BAL	Transaction is eligible for load balancing (for example, with parallel limits specified).
	CPIC	This CPI-C transaction was built dynamically on this IMS system and can process only on this IMS system.
	DYN	Transaction was built in a shared-queues environment, is not defined to this IMS, and therefore, cannot be scheduled in this IMS subsystem.
	IOPREV	Indicates that a BMP program containing GSAM cannot complete scheduling because I/O prevention has not completed. Further I/O requests to data sets are inhibited.
	LOCK	Transaction is locked by a /LOCK TRAN or UPD TRAN SET(LOCK(ON)) command.
	NOTINIT-*xx-reason*	Transaction is not initialized and cannot be used. NOTINIT status is displayed in the format NOTINIT-*xx-reason*, where *xx* is the reason code that identifies the unique location in one module where this reason code is set.
		NOTINIT-00 indicates that the reason is unknown. Action: 1.
		IAPS MACRO defines each reason code that might be set in the transaction bad reason code (field SMBBADR) and identifies the module that sets it. *reason* explains the reason code *xx* in abbreviated text format up to 13 characters.
		For possible *reason* values and their descriptions, see Table 205 on page 650.
	QERR	I/O error has occurred on this queue for this MSC remote transaction.
	QSTP	Transaction queuing is stopped by online change because the transaction is affected by the online change. Online change might be changing or deleting the transaction, or changing or deleting a program, PSB, database, or DMB referenced by the transaction. Transaction queuing is stopped until the online change is committed or aborted.
	SUSPEND	Transaction has messages on the suspend queue.

(to be continued)

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning	
LSTT *(continued)*	LclStat	STATUS	LCL	**STOQ**	Transaction is stopped for queuing, and messages can no longer be queued to it. This status might be caused by a previous UPDATE TRAN, /PURGE TRAN, or /STO TRAN command.
				STOSCHD	Transaction is stopped for scheduling and can no longer be scheduled. This status might be caused by a previous UPDATE TRAN, /PSTOP TRAN, or /STO TRAN command or an application abend.
				TRA	Transaction is being traced.
				USTO	Transaction scheduling stopped because of unavailable data.
LU	LUName	WORK	LCL	APPC LU name that initiated conversation.	
MDLN	LModelName	MODEL	LCL	Model name. Name of the resource used as a model to create this resource. DFSDSTR1 is the IMS descriptor name for transactions.	
MDLT	LModelType	MODEL	LCL	Model type, either RSC or DESC. RSC means that the resource was created using another resource as a model. DESC means that the resource was created using a descriptor as a model.	
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. IMS identifier of IMS that built the output. The IMS identifier is always returned.	
MSGT	LMsgType	MSGTYPE	LCL	Message type of single segment (SNGLSEG) or multiple segment (MULTSEG). The output value is obtained from the local IMS.	
				MULTSEG	Specifies that the incoming message can be more than one segment in length. It is not eligible for scheduling to an application program until an end-of-message indication is received, or a complete message is created by MFS.
				SNGLSEG	Specifies that the incoming message is one segment in length. It becomes eligible for scheduling when the terminal operator indicates end-of-segment.
MSN	LMSName	MSNAME	LCL	Logical link path name. The output value is obtained from the local IMS.	
NODE	NodeName	WORK	LCL	Node name that initiated conversation.	
PLCTT	LPLCTTime	PLCTTIME	LCL	Processing limit count time.	
PSB	LPSBName	PSB	LCL	PSB name associated with the transaction. The output value is obtained from the local IMS.	
Q	Qcnt	QCNT, GLOBAL	GBL	Global transaction message queue count on the shared queues. Q is displayed only if shared queues are used.	

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
RAOCMD	LAOCMD	DEFN, AOCMD	GBL	Transaction supports AOI CMD calls (CMD, TRAN, or Y) or not (N). The output value is obtained from the repository.
RCLS	Cls	DEFN, CLASS	GBL	Class value in the repository.
RCMTM	CmtMode	DEFN, CMTMODE	GBL	Commit mode for the transaction: commit after a single message (SNGL) or multiple messages (MULT). The output value is obtained from the repository.
RCONV	Conv	DEFN, CONV	GBL	Conversation ID if a conversation is in progress in the repository.
RCV	LRecover	RECOVER	LCL	Recovered during an IMS emergency or normal restart (Y) or not (N). The output value is obtained from the local IMS. N Recovery option is disabled. The transaction is not recovered. Y Recovery option is enabled. The transaction is recovered during IMS emergency or normal restart.
RDCLW	DCLWA	DEFN, DCLWA	GBL	Perform log write-ahead for recoverable, nonresponse mode input messages and transaction output messages (Y) or not (N). The output value is obtained from the repository.
RDRRT	DirRoute	DEFN, DIRROUTE	GBL	Supports MSC directed routing (Y) or not (N). The output value is obtained from the repository.
REDTR	Editrtn	DEFN, EDITRTN	GBL	Edit routine value obtained from the repository.
REDTT	EditUC	DEFN, EDITUC	GBL	Input data is to be translated to uppercase (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see "LEditUC" in this table.
REMHBS	EMHBSz	DEFN, EMHBSZ	GBL	EMH buffer size. The output value is obtained from the repository.
REPO	Repo	DEFN	GBL	Indicates whether the output line contains the stored resource definitions. Y Indicates repository definitions. **(blank)** Indicates local definitions.
REXPRT	ExprTm	DEFN, EXPRTIME	GBL	Transaction expiration time. The output value is obtained from the repository.
RFP	FP	DEFN, FP	GBL	Fast Path potential candidate (P), Fast Path exclusive (E), or FP option not enabled (N). The output value is obtained from the repository. For the values to be returned, see the description for "LFP" in this table.
RGC	LRegCnt	RGC	LCL	Number of regions the transaction is currently scheduled in the local IMS. The output value is obtained from the local IMS.
RINQ	Inq	DEFN, INQ	GBL	Inquiry transaction (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LInq" in this table.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
RLCT	Lct	DEFN, LCT	GBL	Limit count value in the repository. The limit count is the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction.
RLP	LPRI	DEFN, LPRI	GBL	Local limit scheduling priority value in the repository. The limit scheduling priority is the priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value.
RMRG	RMaxRgn	DEFN, MAXRGN	GBL	Maximum region count obtained from the repository. The maximum region count is the maximum number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction that is eligible for parallel scheduling.
RMSGT	MsgType	DEFN, MSGTYPE	GBL	Message type of single segment (SNGLSEG) or multiple segment (MULTSEG). The output value is obtained from the repository. For the values to be returned, see the description for "LMsgType" in this table.
RMT	LRemote	REMOTE	LCL	Remote transaction (Y) or not (N). The output value is obtained from the local IMS. **N** Local transaction. The transaction runs on the local system. **Y** Remote transaction. The transaction runs on a remote system. **Restriction:** The method to filter on a remote transaction changed between IMS Version 9 and IMS Version 10. In Version 9, the command is QUERY TRAN STATUS(RMT). In IMS Version 10 or later, the command is QUERY TRAN REMOTE(Y). If IMS is running in an IMSplex with IMS Version 10 or later and either IMS Version 8 or IMS Version 9 systems, and the command is to be processed on all systems, the command must be issued as QUERY TRAN STATUS(RMT) REMOTE(Y). The IMS Version 10 system or later processes the REMOTE(Y) filter, while the IMS Version 8 and IMS Version 9 systems process the STATUS(RMT) filter. If a command is issued to an IMS Version 10 system or later with STATUS(RMT) without REMOTE(Y), the command is rejected.
RNP	NPRI	DEFN, NPRI	GBL	Normal scheduling priority value obtained from the repository. The normal scheduling priority is the priority assigned to this transaction when the number of input transactions enqueued and waiting to be processed is less than the limit count value.
RPLCT	PLCT	DEFN, PLCT	GBL	Processing limit count obtained from the repository. The processing limit count is the number of transaction messages a program can process in a single scheduling.
RPLCTT	PLCTTime	DEFN, PLCTTIME	GBL	Processing limit count time value in the repository.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
RPLM	ParLim	DEFN, PARLIM	GBL	Parallel processing limit count obtained from the repository. The parallel limit count is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. An additional message region is scheduled whenever the transaction queue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.
RPSB	PsbName	DEFN, PGM	GBL	PSB name associated with the transaction. The output value is obtained from the repository.
RRCV	Recover	DEFN, RECOVER	GBL	Recovered during an IMS emergency or normal restart (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LRecover" in this table.
RRMT	Remote	DEFN, REMOTE	GBL	Remote transaction (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LRemote" in this table.
RRSP	Resp	DEFN, RESP	GBL	Response mode transaction (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LResp" in this table.
RSER	Serial	DEFN, SERIAL	GBL	Transaction is processed serially (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LSerial" in this table.
RSIDL	SIDL	DEFN, MSNAME	GBL	Local system ID. The output value is obtained from the repository.
RSIDR	SIDR	DEFN, MSNAME	GBL	Remote system ID. The output value is obtained from the repository.
RSNO	SegNo	DEFN, SEGNO	GBL	Application program output segment limit allowed in message queues for each GU call. The value is obtained from the repository.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
RSP	LResp	RESP	LCL	Response mode transaction (Y) or not (N). The output value is obtained from the local IMS.
				N Response mode option is disabled. For terminals specifying or accepting a default of OPTIONS=TRANRESP, input should not stop after this transaction is entered.
				Y Response mode option is enabled. For terminals specifying or accepting a default of OPTIONS=TRANRESP, no additional messages are to be allowed after this transaction is entered until this transaction sends a response message back to the terminal. Response mode can be forced or negated by individual terminal definition. RESP(Y) is ignored during online processing for all terminals that do no operate in response mode. **Restriction:** The method to filter on a remote transaction changed between IMS Version 9 and IMS Version 10. In Version 9, the command is QUERY TRAN STATUS(RESP). In IMS Version 10 or later, the command is QUERY TRAN RESP(Y). If IMS is running in an IMSplex with IMS Version 10 or later and either IMS Version 8 or IMS Version 9 systems, and the command is to be processed on all systems, the command must be issued as QUERY TRAN STATUS(RESP) RESP(Y). The IMS Version 10 system or later processes the RESP(Y) filter, while the IMS Version 8 and IMS Version 9 systems process the STATUS(RESP) filter. If a command is issued to an IMS Version 10 system or later with STATUS(RESP) without RESP(Y), the command is rejected.
RSPASZ	LSpaSz	DEFN, SPASZ	GBL	Conversational transaction scratchpad area size. The output value is obtained from the repository.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
RSPATR	SpaTrunc	DEFN, SPATRUNC	GBL	Conversational transaction SPA data should be truncated (R) or preserved (S) across a program switch to a transaction that is defined with a smaller SPA. The SPATRUNC value defined for the conversational transaction is stored in the repository. A QUERY TRAN command with SHOW(DEFN) returns a SPATRUNC value of R, S, or null from the repository values.
				S S is shown on the QRY TRAN command for a conversational transaction in one of the following conditions: • If the transaction is defined with SPATRUNC=S on the CREATE TRAN or CREATE TRANDESC command • If SPA=STRUNC is specified on the TRANSACT macro • If the system-wide truncated data option is set as TRUNC=Y in the DFSDCxxx member
				R R is shown on the QRY TRAN command for a conversational transaction in one of the following conditions: • If the transaction is defined with SPATRUNC=R on the CREATE TRAN or CREATE TRANDESC command • If SPA=RTRUNC is specified on the TRANSACT macro • If the system-wide truncated data option is not set as TRUNC=N in the DFSDCxxx member.
				null A null value indicates that the transaction does not have the SPATRUNC value defined and that the value is overridden with the system-wide truncated data option defined with the TRUNC= option on the DFSDCxxx member.
RSSSZ	SegSz	DEFN, SEGSZ	GBL	Application program output segment size limit allowed in the message queues for each GU call. The value is obtained from the repository.
RTLS	TranStat	DEFN, TRANSTAT	GBL	Transaction level statistics logged (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LTranStat" in this table.
RTMCR	TimeCreate	DEFN, TIMESTAMP	GBL	Create time from the repository. This is the time the resource was first created in the repository.
RTMUP	TimeUpdate	DEFN, TIMESTAMP	GBL	Update time from the repository. This is the time the resource was last updated in the repository.
RWFI	WFI	DEFN, WFI	GBL	Wait-for-input transaction (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LWFI" in this table.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
SER	LSerial	SERIAL	LCL	Transaction is processed serially (Y) or not (N). The output value is obtained from the local IMS.
				N Serial option is disabled. Messages for the transaction are not processed serially. Message processing can be processed in parallel. Messages are placed on the suspend queue after a U3303 pseudoabend. Scheduling continues until repeated failures result in the transaction being stopped with a USTOP.
				Y Serial option is enabled. Messages for the transaction are processed serially. U3303 pseudoabends do not cause the message to be placed on the suspend queue but rather on the front of the transaction message queue, and the transaction is stopped with a USTOP. The USTOP of the transaction is removed when the transaction or the class is started with a /START command.
SIDL	LSIDL	MSNAME	LCL	Local system ID. The output value is obtained from the local IMS.
SIDR	LSIDR	MSNAME	LCL	Remote system ID. The output value is obtained from the local IMS.
SPASZ	LSpaSz	SPASZ	LCL	Conversational transaction scratchpad area size. The output value is obtained from the local IMS.
SPATR	LSPATrunc	SPATRUNC	LCL	Conversational transaction SPA data should be truncated (R) or preserved (S) across a program switch to a transaction that is defined with a smaller SPA. The output value is obtained from the local IMS.
				S IMS preserves all the data in the SPA, even when a program switch is made to a transaction that is defined with a smaller SPA. The transaction with the smaller SPA does not see the truncated data, but when the transaction switches to a transaction with a larger SPA, the truncated data is used.
				R Truncated data is not preserved.
STT	Status	GLOBAL, STATUS	GBL	Global transaction status, which can be one of the following:
				STAQ Transaction has a global status of started for queuing.
				STASCHD Transaction has a global status of started for scheduling.
				STOQ Transaction has a global status of stopped for queuing.
				STOSCHD Transaction has a global status of stopped for scheduling.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
TLS	LTranStat	TRANSTAT	LCL	Transaction level statistics logged (Y) or not (N). The output value is obtained from the local IMS. **N** Transaction level statistics logging is not active. **Y** Transaction level statistics logging is active.
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the resource was last accessed. The output value is obtained from the local IMS. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. The transaction time stamp is not updated for Fast Path exclusive transactions; only the associated routing code time stamp is updated. For a transaction descriptor, the following actions update the last access time: • Message is queued to the transaction from any source (program, command, other). • Message is dequeued by an application program (MSG GU). If a DEQ command is issued to dequeue a message, that action does not update the access time. • CREATE command or DFSINSX0 exit references the resource as a model.
TMCR	LTimeCreate	TIMESTAMP	LCL	The time that the resource was created. The output value is obtained from the local IMS. This is the result of a CREATE TRAN command, IMPORT command that creates the transaction, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT.
TMEM	TMember	WORK	LCL	OTMA tmember that initiated conversation.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the resource was last imported. The import time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated as a result of the UPDATE TRAN, a type-1 command, or the IMPORT command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
TPIP	TPipe	WORK	LCL	OTMA tpipe that initiated conversation.
TRAN	Trancode	TRAN	LCL	Transaction name. A transaction defines the processing characteristics of messages destined for an application program.
USER	UserName	WORK	LCL	User that initiated conversation.

Table 204. Output fields for the QUERY TRAN command (continued)

Short label	Long label	Keyword	Scope	Meaning
WFI	LWFI	WFI	LCL	Wait-for-input transaction (Y) or not (N). The output value is obtained from the local IMS.
				N Wait-for-input option is disabled.
				Y Wait-for-input option is enabled. A message processing or batch processing application program that processes WFI transactions is scheduled and invoked normally. If the transaction to be processed is defined as WFI, the program is allowed to remain in main storage after it has processed the available input messages. The QC status code (no more messages) is returned to the program if the processing limit count is reached; a command is entered to change the status of the schedule transaction, database, program, or class; commands relating to the databases used by the transaction are entered; or IMS is terminated with a checkpoint shutdown.
WRK	Work	WORK	LCL	Work is in progress for the transaction or one of its associated resources. The work in progress can be one of the following:
				ANOTHER CMD IN PROGRESS Another command (DELETE, IMPORT, UPDATE) to delete or update the transaction is already in progress.
				IN CONVERSATION Transaction is in conversation. Additional information that uniquely identifies the conversation such as the conversation ID, username or ltermname (if conversation stored in RM), node, lterm, user, luname, tmember, or tpipe is returned as separate information with unique XML tags.
				IN USE Transaction is in use. Queuing is in progress, either terminal input or a program-to-program switch.
				QUEUEING Transaction has messages queued in a non-shared-queues environment.
				SCHEDULED Transaction is scheduled or application program referenced by transaction is scheduled.
				SUSPENDED Transaction is on the suspend queue.

Table 205. Reason information for NOTINIT-xx-reason status

Reason	Meaning
EDTRTN	The input edit routine that is referred to by the transaction could not be loaded.

Table 205. Reason information for NOTINIT-xx-reason status (continued)

Reason	Meaning
NOMSN	The MSNAME that is referred to by the transaction does not exist.
NOPGM	The program that this transaction references is not defined. No program PDIR control block exists. Action: 2.
NORTC	The Fast Path routing code that this transaction references is not defined. No Fast Path routing code RCTE control block exists. Action: 3.
NOTBL	The table that is used to manage transaction input edit routines could not be allocated.
REPOCHGLIST	The transaction name exists in the change list in the IMSRSC repository for this IMS, and the change list is being processed or the change list processing failed because of an error. This transaction cannot be used, scheduled, updated, exported, or referenced as a model until it is successfully imported so that the runtime transaction definition and the stored definition in the repository are synchronized. Action: 4.

Note: Actions that can be taken to initialize the transaction are:

1. Call IBM.
2. Issue a CREATE PGM command to create the program. Issue an UPDATE TRAN START(Q,SCHD) command to complete transaction initialization.
3. Issue a CREATE RTC command to create the routing code. Issue an UPDATE TRAN START(Q,SCHD) command to complete transaction initialization.
4. If IMS change list processing failed and message DFS4411E was issued, correct the problem and take one of the following actions:
   - Shut down IMS and restart it in order to try processing the change list again.
   - Issue the IMPORT DEFN SOURCE(REPO) NAME(*tranname*) TYPE(TRAN) command to import the transaction, where *tranname* is the transaction name.

How the SHOW keyword on QUERY TRAN determines the type of output

The following table provides some examples of how the SHOW keyword determines the type of output returned on the QUERY TRAN command.

Table 206. How the SHOW keyword on QUERY TRAN determines the type of output

Form of SHOW keyword used	Type of output returned
SHOW(LOCAL)	Only those output fields that are local to an IMS system. SHOW(ALL,LOCAL) provides the same output.
SHOW(GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM or CQS. SHOW(ALL,GLOBAL) provides the same output.
SHOW(ALL)	All the output fields for those fields that have both local and global data. Both values are returned in the output.
SHOW(QCNT,GLOBAL)	Only global QCNT values.
SHOW(QCNT,LOCAL)	Only local QCNT values.
SHOW(QCNT)	Both local and global QCNT values, because there is no LOCAL or GLOBAL qualifier.

Table 206. How the SHOW keyword on QUERY TRAN determines the type of output (continued)

Form of SHOW keyword used	Type of output returned
SHOW(ALL,GLOBAL)	Only those output fields that are globally maintained, such as data maintained by RM or CQS. SHOW(GLOBAL) provides the same output.
SHOW(ALL,LOCAL)	Only those output fields that are local to an IMS system. SHOW(LOCAL) provides the same output.

If specific output fields are listed in the SHOW keyword, you can specify them as either local or global; however, you cannot combine a list of SHOW keywords in which some are global and some are local.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY TRAN command are standard for all commands entered through the OM API.

The following table contains the return and reason codes that can be returned to OM from a QUERY TRAN command.

Table 207. Return and reason codes for the QUERY TRAN command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY TRAN command completed successfully.
X'00000004'	X'00000010'	No resources were found to be returned. The resource names specified might be invalid, or: • There were no resources that match the filter specified • There were no resources that had work to display for the SHOW(WORK) specified. • There were no resources that have been created or updated but have not been exported to the IMSRSC repository when the SHOW(EXPORTNEEDED) was specified.
X'00000004'	X'00001000'	The QUERY TRAN command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000004'	X'00001004'	The QUERY TRAN command was processed for a few resources and a partial list of resources is returned. The command terminated because the resource count to be returned exceeded the maximum number of resources that can be returned by a QUERY TRAN command. The maximum number of resources that can be returned by a QUERY TRAN command is 5000. Issue the command again with a generic mask or other filters so the number of resources to be returned is less than 5000.

Table 207. Return and reason codes for the QUERY TRAN command (continued)

Return code	Reason code	Meaning
X'00000004'	X'00001010'	No resources were found to be returned. The resource names specified might be invalid or: • There were no resources that match the filter specified. • There were no resources that had work to display for the SHOW(WORK) specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'00000008'	X'00002040'	The QUERY TRAN command has more than one filter value specified or an invalid filter value is specified for the QCNT filter. Only one of the CLASS, STATUS, or QCNT filters can be specified. A value of 0 might have been specified for QCNT with LE, GE or, EQ. Or a value of 1 might have been specified for QCNT with LT.
X'00000008'	X'0000204C'	The CLASS value specified is invalid. Confirm the correct CLASS value is specified on the command.
X'00000008'	X'00002137'	Transaction referenced as a model is a queue-only transaction with a status of DYN, which cannot be used as a model.
X'0000000C'	X'00003000'	The QUERY TRAN command was successful for some resources but failed for others. The completion code indicates the reason for the error with the resource name.
X'0000000C'	X'00003004'	The QUERY TRAN command was not successful for any of the resource names specified. The completion code indicates the reason for the error with the resource name.
X'00000010'	X'00004004'	The QUERY TRAN command processing terminated because there is no CQS address space.
X'00000010'	X'00004014'	The QUERY TRAN command processing terminated because the TRAN keyword is not valid on the RSR tracker.
X'00000010'	X'00004018'	The QUERY TRAN command processing terminated because no resource structure exists or the resource structure is not available.
X'00000010'	X'00004100'	The QUERY TRAN command processing terminated because the resource structure is full.
X'00000010'	X'00004104'	The QUERY TRAN command processing terminated because there is no RM address space.
X'00000010'	X'00004108'	The QUERY TRAN command processing terminated because there is no SCI address space.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS OLC, or MODBLKS not defined).
X'00000010'	X'00004500'	IMS is not enabled to use the repository.

Table 207. Return and reason codes for the QUERY TRAN command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Access to the repository was denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server (RS) is being shut down.
X'00000010'	X'0000450B'	RS is not available.
X'00000010'	X'0000450C'	RS is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields related to the IMSRSC repository.
X'00000014'	X'00005004'	The QUERY TRAN command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	The QUERY TRAN command processing terminated because the DFSPOOL storage could not be obtained.
X'00000014'	X'0000501C'	IMODULE GETMAIN error.
X'00000014'	X'00005100'	The QUERY TRAN command processing terminated because of an RM error.
X'00000014'	X'00005104'	CQS error.
X'00000014'	X'00005108'	The QUERY TRAN command processing terminated because of an SCI request error.
X'00000014'	X'00005110'	Repository error.
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY TRAN command is rejected because the resource list is unusable. Contact IBM Software Support. Send in the output from the command in error, and a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the IMSRSC repository RID and RMD data sets.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains completion codes that can be returned on a QUERY TRAN command.

Table 208. Completion codes for the QUERY TRAN command

Completion code	Completion code text	Meaning
0		Command completed successfully for transaction.
10	NO RESOURCES FOUND	Transaction name is invalid, or the wildcard parameter specified does not match any resource names.
50	CQS IS UNAVAILABLE	The QUERY TRAN command could not be completed for the resource as CQS is not available. Make sure CQS is available before issuing the command again.
51	NO RESOURCE STRUCTURE	The QUERY TRAN command could not be completed for the resource as there is no resource structure or it is unavailable.
52	RESOURCE STRUCTURE FULL	The QUERY TRAN command could not be completed for the resource as the resource structure is full.
90	INTERNAL ERROR	The QUERY TRAN command could not be completed for the resource because of an IMS internal error.
94	RM REQUEST ERROR	The QUERY TRAN command could not be completed for the resource because of an RM error.
98	CQS REQUEST ERROR	The QUERY TRAN command could not be completed for the resource because of a CQS error.

Examples

The following are examples of the QUERY TRAN command:

Example 1 for QUERY TRAN command

TSO SPOC input:
```
QRY TRAN NAME(PART,TRAN000%,CDEBTRN4,CONV13V0,TXBANKI4,FPACP) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
Trancode MbrName    CC LPSBname LCls    LQCnt  LLCT LPLCT LPLCTTime LCPRI LNPRI
CDEBTRN4 IMS1        0 CDEBS       1        0 65535 65535   6553500     1     1
CONV13V0 IMS1        0 CPGM1V0     1        0 65535 65535   6553500     1     1
FPACP    IMS1        0 TACP1       1        0 65535 65535   6553500     1     1
PART     IMS1        0 DFSSAM02    4        0     2 65535   6553500     7     7
TRAN0001 IMS1        0 DFSSAM02    4        0     2 65535   6553500     7     7
TRAN0002 IMS1        0 DFSSAM04    1        0 65535 65535   6553500     1     1
TXBANKI4 IMS1        0 BANKIFP     1        0     0 65535   6553500     1     1
(scrolled to the right screen 2)
Trancode MbrName    LLPRI LSegSz LSegNo LParLim LRegCnt LMaxRgn LEditRtn LFP LEMHB
CDEBTRN4 IMS1           1      0      0   65535       0       0        N
CONV13V0 IMS1           1      0      0   65535       0       0        N
FPACP    IMS1           1      0      0   65535       0       0        P
PART     IMS1          10      0      0   65535       0       0        N
TRAN0001 IMS1          10      0      0   65535       0       0        N
TRAN0002 IMS1           1      0      0   65535       0       0        N
TXBANKI4 IMS1           1      0      0   65535       0       0        E
```

```
(scrolled to the right screen 3)
Trancode MbrName   LEMHBSz LCmtMode LMsgType LSPATrunc LSPASz LSIDR LSIDL LDCLWA
CDEBTRN4 IMS1            0 SNGL     MULTSEG         S     1000    10    10 Y
CONV13V0 IMS1            0 SNGL     MULTSEG         S     1000    16    36 Y
FPACP    IMS1            0 SNGL     SNGLSEG               0       10    10 Y
PART     IMS1            0 MULT     MULTSEG               0       10    10 Y
TRAN0001 IMS1            0 MULT     MULTSEG               0       10    10 Y
TRAN0002 IMS1            0 SNGL     MULTSEG               0       10    10 Y
TXBANKI4 IMS1            0 SNGL     SNGLSEG               0       10    10 Y
(scrolled to the right screen 4)
Trancode MbrName   LDirRoute LEditUC LInq LRecover LResp LRemote LSerial LWFI
CDEBTRN4 IMS1       N         Y       N    Y        N     N       N       N
CONV13V0 IMS1       N         Y       N    Y        N     Y       N       N
FPACP    IMS1       N         Y       N    Y        Y     N       N       N
PART     IMS1       N         Y       Y    Y        N     N       N       N
TRAN0001 IMS1       N         Y       Y    Y        N     N       N       N
TRAN0002 IMS1       N         Y       N    Y        N     N       N       N
TXBANKI4 IMS1       N         Y       N    Y        Y     N       N       N
(scrolled to the right screen 5)
Trancode MbrName   LAOCMD LConv LTranStat LclStat       LModelName LModelType LMSName
CDEBTRN4 IMS1       N      Y     N
CONV13V0 IMS1       N      Y     N                                             LINK31V1
FPACP    IMS1       N      N     N
PART     IMS1       N      N     N         STOQ,STOSCHD
TRAN0001 IMS1       N      N     N                       PART       RSC
TRAN0002 IMS1       N      N     N                       DFSDSTR1   DESC
TXBANKI4 IMS1       N      N     N
(scrolled to the right screen 6)
Trancode MbrName   LMSName  LTimeCreate          LTimeUpdate
CDEBTRN4 IMS1                2011.182 11:39:21.86 2011.182 13:18:45.91
CONV13V0 IMS1      LINK31V1  2011.182 11:39:21.86 2011.182 13:18:45.91
FPACP    IMS1                2011.182 11:39:21.86 2011.182 13:19:01.81
PART     IMS1                2011.182 11:39:21.86 2011.182 13:19:09.30
TRAN0001 IMS1                2011.182 13:03:18.32 2011.182 13:19:21.67
TRAN0002 IMS1                2011.182 13:06:26.21 2011.182 13:19:27.69
TXBANKI4 IMS1                2011.182 11:39:21.87 2011.182 13:23:15.27
(scrolled to the right screen 7)
Trancode MbrName LTimeAccess          LTimeImport         LDefnType LExprTm
CDEBTRN4 IMS1                                              MODBLKS         0
CONV13V0 IMS1                                              MODBLKS         0
FPACP    IMS1                                              UPDATE        200
PART     IMS1    2011.182 13:22:50.75                      UPDATE        100
TRAN0001 IMS1                                              CREATE        450
TRAN0002 IMS1                                              CREATE        550
TXBANKI4 IMS1                                              UPDATE        300
```

OM API input:

```
CMD(QRY TRAN NAME(PART,TRAN000%,CDEBTRN4,CONV13V0,TXBANKI4,FPACP) SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.182 20:27:15.389095</statime>
<stotime>2011.182 20:27:15.404799</stotime>
<staseq>C8018209EFEA735A</staseq>
<stoseq>C8018209F3BFFB55</stoseq>
<rqsttkn1>USRT005 10132715</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
```

```
<verb>QRY </verb>
<kwd>TRAN          </kwd>
<input>QRY TRAN NAME(PART,TRAN000%,CDEBTRN4,CONV13V0,TXBANKI4,FPACP)
 SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="PSB" llbl="LPSBname" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="LCLS" llbl="LCls" scope="LCL" sort="n" key="0" scroll="yes"
 len="3" dtype="INT" align="right" />
<hdr slbl="LQ" llbl="LQCnt" scope="LCL" sort="d" key="4" scroll="yes"
 len="8" dtype="INT" align="right" />
<hdr slbl="LLCT" llbl="LLCT" scope="LCL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" />
<hdr slbl="LPLCT" llbl="LPLCT" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="PLCTT" llbl="LPLCTTime" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="INT" align="right" />
<hdr slbl="LCP" llbl="LCPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="LNP" llbl="LNPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="LLP" llbl="LLPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="LSSZ" llbl="LSegSz" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="LSNO" llbl="LSegNo" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="LPLM" llbl="LParLim" scope="LCL" sort="n" key="0"
scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="RGC" llbl="LRegCnt" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="LMRG" llbl="LMaxRgn" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="EDTR" llbl="LEditRtn" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="FP" llbl="LFP" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="EMHBS" llbl="LEMHBSz" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="CMTM" llbl="LCmtMode" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="MSGT" llbl="LMsgType" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="CHAR" align="left" />
<hdr slbl="SPATR" llbl="LSPATrunc" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="right" />
<hdr slbl="SPASZ" llbl="LSPASz" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="SIDR" llbl="LSIDR" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="SIDL" llbl="LSIDL" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="DCLW" llbl="LDCLWA" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="DRRT" llbl="LDirRoute" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="EDTT" llbl="LEditUC" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="INQ" llbl="LInq" scope="LCL" sort="n" key="0" scroll="yes"
```

```
|                                 len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RCV" llbl="LRecover" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RSP" llbl="LResp" scope="LCL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RMT" llbl="LRemote" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="SER" llbl="LSerial" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="WFI" llbl="LWFI" scope="LCL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="AOCMD" llbl="LAOCMD" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="4" dtype="CHAR" align="left" />
|                              <hdr slbl="CONV" llbl="LConv" scope="LCL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="TLS" llbl="LTranStat" scope="LCL" sort="n" key="0"
|                              scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="LSTT" llbl="LclStat" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="*" dtype="CHAR" align="left" />
|                              <hdr slbl="MDLN" llbl="LModelName" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="8" dtype="CHAR" align="left" />
|                              <hdr slbl="MDLT" llbl="LModelType" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="4" dtype="CHAR" align="left" />
|                              <hdr slbl="MSN" llbl="LMSName" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="8" dtype="CHAR" align="left" />
|                              <hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" />
|                              <hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" />
|                              <hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
|                              <hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" />
|                              <hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="8" dtype="CHAR" align="left" />
|                              <hdr slbl="EXPRT" llbl="LExprTm" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="5" dtype="INT" align="right" />
|                           </cmdrsphdr>
|                           <cmdrspdata>
|                           <rsp>TRAN(PART    ) MBR(IMS1    ) CC(   0) PSB(DFSSAM02) LCLS(  4) LQ(
|                              0) LLCT(   2) LPLCT(65535) LCP( 7) LNP( 7) LLP(10) LSSZ(    0)
|                            LSNO(   0) LPLM(65535) RGC(   0) LMRG(   0) LSTT(STOQ,STOSCHD)
|                            AOCMD(N) CMTM(MULT) CONV(N) DCLW(Y) DFNT(UPDATE) DRRT(N) EDTT(Y)
|                            EMHBS(   0) EXPRT(  100) FP(N) INQ(Y) MSGT(MULTSEG) PLCTT(6553500)
|                            RCV(Y) RMT(N) RSP(N) SER(N) SIDL(  10) SIDR(  10) SPASZ(    0)
|                            TMAC(2011.182 13:22:50.75) TMUP(2011.182 13:19:09.30) TMCR(2011.182
|                            11:39:21.86) TLS(N) WFI(N) </rsp>
|                           <rsp>TRAN(TRAN0002) MBR(IMS1    ) CC(   0) PSB(DFSSAM04) LCLS(  1) LQ(
|                              0) LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
|                            LSNO(   0) LPLM(65535) RGC(   0) LMRG(   0) AOCMD(N) CMTM(SNGL)
|                            CONV(N) DCLW(Y) DFNT(CREATE) DRRT(N) EDTT(Y) EMHBS(   0) EXPRT(  550)
|                            FP(N) INQ(N) MDLT(DESC) MDLN(DFSDSTR1) MSGT(MULTSEG) PLCTT(6553500)
|                            RCV(Y) RMT(N) RSP(N) SER(N) SIDL(  10) SIDR(  10) SPASZ(    0)
|                            TMUP(2011.182 13:19:27.69) TMCR(2011.182 13:06:26.21) TLS(N) WFI(N)
|                           </rsp>
|                           <rsp>TRAN(TRAN0001) MBR(IMS1    ) CC(   0) PSB(DFSSAM02) LCLS(  4) LQ(
|                              0) LLCT(   2) LPLCT(65535) LCP( 7) LNP( 7) LLP(10) LSSZ(    0)
|                            LSNO(   0) LPLM(65535) RGC(   0) LMRG(   0) AOCMD(N) CMTM(MULT)
|                            CONV(N) DCLW(Y) DFNT(CREATE) DRRT(N) EDTT(Y) EMHBS(   0) EXPRT(  450)
|                            FP(N) INQ(Y) MDLT(RSC) MDLN(PART    ) MSGT(MULTSEG) PLCTT(6553500)
|                            RCV(Y) RMT(N) RSP(N) SER(N) SIDL(  10) SIDR(  10) SPASZ(    0)
|                            TMUP(2011.182 13:19:21.67) TMCR(2011.182 13:03:18.32) TLS(N) WFI(N)
|                           </rsp>
|                           <rsp>TRAN(CDEBTRN4) MBR(IMS1    ) CC(   0) PSB(CDEBS   ) LCLS(  1) LQ(
|                              0) LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
|                            LSNO(   0) LPLM(65535) RGC(   0) LMRG(   0) AOCMD(N) CMTM(SNGL)
|                            CONV(Y) DCLW(Y) DFNT(UPDATE) DRRT(N) EDTT(Y) EMHBS(   0) EXPRT(   0)
```

```
          FP(N) INQ(N) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(N) SER(N)
          SIDL(  10) SIDR(  10) SPASZ( 1000) SPATR(S) TMUP(2011.182 13:18:45.91)
          TMCR(2011.182 11:39:21.86) TLS(N) WFI(N) </rsp>
          <rsp>TRAN(CONV13V0) MBR(IMS1    ) CC(  0) PSB(CPGM1V0 ) LCLS(  1) LQ(
            0) LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
          LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL)
          CONV(Y) DCLW(Y) DFNT(UPDATE) DRRT(N) EDTT(Y) EMHBS(    0) EXPRT(    0)
          FP(N) INQ(N) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(Y) RSP(N) SER(N)
          SIDL(  36) SIDR(  16) MSN(LINK31V1) SPASZ( 1000) SPATR(S)
          TMUP(2011.182 13:18:45.91) TMCR(2011.182 11:39:21.86) TLS(N) WFI(N)
          </rsp>
          <rsp>TRAN(TXBANKI4) MBR(IMS1    ) CC(  0) PSB(BANKIFP ) LCLS(  1) LQ(
            0) LLCT(    0) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
          LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL)
          CONV(N) DCLW(Y) DFNT(UPDATE) DRRT(N) EDTT(Y) EMHBS(    0) EXPRT( 300)
          FP(E) INQ(N) MSGT(SNGLSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(Y) SER(N)
          SIDL(  10) SIDR(  10) SPASZ(    0) TMUP(2011.182 13:23:15.27)
          TMCR(2011.182 11:39:21.87) TLS(N) WFI(N) </rsp>
          <rsp>TRAN(FPACP   ) MBR(IMS1    ) CC(  0) PSB(TACP1    ) LCLS(  1) LQ(
            0) LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0)
          LSNO(    0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL)
          CONV(N) DCLW(Y) DFNT(UPDATE) DRRT(N) EDTT(Y) EMHBS(    0) EXPRT( 200)
          FP(P) INQ(N) MSGT(SNGLSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(Y) SER(N)
          SIDL(  10) SIDR(  10) SPASZ(    0) TMUP(2011.182 13:19:01.81)
          TMCR(2011.182 11:39:21.86) TLS(N) WFI(N) </rsp>
          </cmdrspdata>
          </imsout>
```

Explanation: The QUERY TRAN command is specified with SHOW(ALL) to display all the output fields for the specified transactions. All the transaction output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The transaction name and the member name that built the line of output are displayed on every screen. Transaction PART has a status of stopped for scheduling and queuing. Transactions CDEBTRN4 and CONV13V0 are conversational. Transaction FPACP is Fast Path potential. Transaction CONV13V0 is remote. Transactions TRAN0001 and TRAN0002 are dynamic, they display the LModelName and LModelType used to create them, and they have a definition type of CREATE.

Example 2 for QUERY TRAN command

TSO SPOC input:
```
QUERY TRAN STATUS(AFFIN)
```

TSO SPOC output:
```
 Trancode MbrName    CC LclStat
 APOL12   IMS1        0 AFFIN
```

Explanation: This example shows the use of the AFFIN option as a filter to query transactions that have affinity status.

Example 3 for QUERY TRAN command

TSO SPOC input:
```
QUERY TRAN NAME(APOL11 APOL12) SHOW(STATUS)
```

TSO SPOC output:
```
 Trancode MbrName    CC LclStat
 APOL11   IMS1        0
 APOL12   IMS1        0 AFFIN
```

Explanation: This example queries transaction status for transactions APOL11 and APOL12. The output shows that transaction APOL12 has affinity status.

Example 4 for QUERY TRAN command

TSO SPOC input:

```
QUERY TRAN QCNT(GT,0) SHOW(AFFIN)
```

TSO SPOC output:

```
  Trancode MbrName    CC     QCnt Affinity
  APOL12   IMS1        0        1
  APOL12   IMS1        0        1 IMS1
```

Explanation: This example queries transaction messages on the shared queue and shows which messages have affinity status.

Example 5 for QUERY TRAN command

TSO SPOC input:

```
QRY TRAN NAME(PART) SHOW(DEFN,CLASS)
```

TSO SPOC output:

```
Trancode MbrName     CC Repo IMSid     Cls LCls
PART     IMS1         0 Y               1
PART     IMS1         0 Y   IMS3        5
PART     IMS1         0     IMS1            1
PART     IMS2         0     IMS2           99
PART     IMS3         0     IMS3            5
```

OM API input:

```
CMD(QRY TRAN NAME(PART) SHOW(DEFN,CLASS))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.182 21:12:56.087584</statime>
<stotime>2011.182 21:12:56.099946</stotime>
<staseq>C8018C3FABC2038C</staseq>
<stoseq>C8018C3FAEC6A010</stoseq>
<rqsttkn1>USRT005 10141256</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>TRAN            </kwd>
<input>QRY TRAN NAME(PART) SHOW(DEFN,CLASS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
```

```
                    scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
                  <hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
                  len="1" dtype="CHAR" align="left" />
                  <hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
                   scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
                  <hdr slbl="RCLS" llbl="Cls" scope="GBL" sort="n" key="0" scroll="yes"
                   len="3" dtype="INT" align="right" />
                  <hdr slbl="LCLS" llbl="LCls" scope="LCL" sort="n" key="0" scroll="yes"
                   len="3" dtype="INT" align="right" />
                  </cmdrsphdr>
                  <cmdrspdata>
                  <rsp>TRAN(PART    ) MBR(IMS1    ) CC(   0) LCLS(  1) IMSID(IMS1    )
                  </rsp>
                  <rsp>TRAN(PART    ) MBR(IMS1    ) CC(   0) REPO(Y) RCLS(  1) </rsp>
                  <rsp>TRAN(PART    ) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS3    )
                   RCLS(  5) </rsp>
                  <rsp>TRAN(PART    ) MBR(IMS3    ) CC(   0) LCLS(  5) IMSID(IMS3    )
                  </rsp>
                  <rsp>TRAN(PART    ) MBR(IMS2    ) CC(   0) LCLS( 99) IMSID(IMS2    )
                  </rsp>
                  </cmdrspdata>
                  </imsout>
```

Explanation: The stored resource definitions and the runtime resource definitions
for the specified resources are returned. PART has the generic definition in the
repository and also a specific section for IMS3 in the repository.

Example 6 for QUERY TRAN command

TSO SPOC input:

```
QRY TRAN NAME(PART*) SHOW(DEFN)
```

TSO SPOC output:

(screen 1)

Trancode	MbrName	CC	Repo	IMSid	PSBname	LPSBname	Cls	LCls	LCT	LLCT	PLC
PART	IMS1	0	Y		DFSSAM02		4		2		6553
PART	IMS1	0		IMS1		DFSSAM02		4		2	
PART	IMS2	0		IMS2		DFSSAM02		4		2	
PARTROOT	IMS1	0	Y		TPARTAPP		1		2		6553
PARTROOT	IMS1	0		IMS1		TPARTAPP		1		2	
PARTROOT	IMS2	0		IMS2		TPARTAPP		1		2	

(scrolled to the right screen 2)

Trancode	MbrName	Repo	CT	LPLCT	PLCTTime	LPLCTTime	LCPRI	NPRI	LNPRI	LPRI	LLPRI
PART	IMS1	Y	35		6553500			7		10	
PART	IMS1			65535		6553500	7		7		10
PART	IMS2			65535		6553500	7		7		10
PARTROOT	IMS1	Y	35		6553500			7		10	
PARTROOT	IMS1			65535		6553500	7		7		10
PARTROOT	IMS2			65535		6553500	7		7		10

(scrolled to the right screen 3)

Trancode	MbrName	Repo	SegSz	LSegSz	SegNo	LSegNo	ParLim	LParLim	LRegCnt	MaxRgn
PART	IMS1	Y	0		0		65535			0
PART	IMS1			0		0		65535	0	
PART	IMS2			0		0		65535	0	
PARTROOT	IMS1	Y	0		0		65535			0
PARTROOT	IMS1			0		0		65535	0	
PARTROOT	IMS2			0		0		65535	0	

(scrolled to the right screen 4)

Trancode	MbrName	Repo	LMaxRgn	EditRtn	LEditRtn	FP	LFP	EMHBSz	LEMHBSz	CmtMode
PART	IMS1	Y				N		0		MULT
PART	IMS1		0			N		0		

```
| PART     IMS2            0                    N              0
| PARTROOT IMS1    Y                            N        0        SNGL
| PARTROOT IMS1            0                    N              0
| PARTROOT IMS2            0                    N              0
|
| (scrolled to the right screen 5)
| Trancode MbrName  Repo LCmtMode MsgType LMsgType SPATrunc LSPATrunc SPASz LSPASz
| PART     IMS1     Y             MULTSEG                             0
| PART     IMS1          MULT             MULTSEG                           0
| PART     IMS2          MULT             MULTSEG                           0
| PARTROOT IMS1     Y             MULTSEG                             0
| PARTROOT IMS1          SNGL             MULTSEG                           0
| PARTROOT IMS2          SNGL             MULTSEG                           0
|
| (scrolled to the right screen 6)
| Trancode MbrName  Repo SIDR LSIDR SIDL LSIDL DCLWA LDCLWA DirRoute LDirRoute
| PART     IMS1     Y    0         0        Y          N
| PART     IMS1                             Y          N
| PART     IMS2                             Y          N
| PARTROOT IMS1     Y    0         0        Y     N
| PARTROOT IMS1                             Y          N
| PARTROOT IMS2                             Y          N
|
| (scrolled to the right screen 7)
| Trancode MbrName  Repo EditUC LEditUC Inq LInq Recover LRecover Resp LResp Remote
| PART     IMS1     Y    Y            Y        Y               N          N
| PART     IMS1          Y            Y        Y               N
| PART     IMS2          Y            Y        Y               N
| PARTROOT IMS1     Y    Y            N        Y               N          N
| PARTROOT IMS1          Y            N        Y               N
| PARTROOT IMS2          Y            N        Y               N
|
| (scrolled to the right screen 8)
| Trancode MbrName  Repo LRemote Serial LSerial WFI LWFI AOCMD LAOCMD Conv LConv
| PART     IMS1     Y           N            N        N          N
| PART     IMS1          N            N          N        N          N
| PART     IMS2          N            N          N        N          N
| PARTROOT IMS1     Y           N            N        N          N
| PARTROOT IMS1          N            N          N        N          N
| PARTROOT IMS2          N            N          N        N          N
|
| (scrolled to the right screen 9)
| Trancode MbrName  Repo TranStat LTranStat LMSName TimeCreate          LTimeCreate
| PART     IMS1     Y    N                           2011.189 19:03:17.76
| PART     IMS1               N                                          2011.189 19:03:17.76
| PART     IMS2               N                                          2011.189 19:01:55.17
| PARTROOT IMS1     Y    N                           2011.189 19:03:17.76
| PARTROOT IMS1              N                                           2011.189 19:03:17.76
| PARTROOT IMS2              N                                           2011.189 19:01:55.17
|
| (scrolled to the right screen 10)
| Trancode MbrName  Repo TimeUpdate          LTimeUpdate
| PART     IMS1     Y
| PART     IMS1                               2011.189 19:04:25.07
| PART     IMS2                               2011.189 19:11:47.65
| PARTROOT IMS1     Y
| PARTROOT IMS1
| PARTROOT IMS2
|
| (scrolled to the right screen 11)
| Trancode MbrName  Repo LTimeAccess         LTimeImport         ExprTm LExprTm
| PART     IMS1     Y                                            100
| PART     IMS1          2011.189 19:04:19.92                           100
| PART     IMS2          2011.189 19:04:19.92                           100
| PARTROOT IMS1     Y                                            0
| PARTROOT IMS1                                                         0
| PARTROOT IMS2                                                         0
```

OM API input:

```
CMD(QRY TRAN NAME(PART*) SHOW(DEFN))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.190 02:15:21.735325</statime>
<stotime>2011.190 02:15:21.827190</stotime>
<staseq>C80A9CE6EF89DD85</staseq>
<stoseq>C80A9CE705F76C0C</stoseq>
<rqsttkn1>USRT011 10191521</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>QRY </verb>
<kwd>TRAN           </kwd>
<input>QRY TRAN NAME(PART*) SHOW(DEFN) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RPSB" llbl="PSBname" scope="GBL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="PSB" llbl="LPSBname" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="STT" llbl="Status" scope="GBL" sort="d" key="3" scroll="yes"
 len="*" dtype="CHAR" align="left" />
<hdr slbl="RCLS" llbl="Cls" scope="GBL" sort="n" key="0" scroll="yes"
 len="3" dtype="INT" align="right" />
<hdr slbl="LCLS" llbl="LCls" scope="LCL" sort="n" key="0" scroll="yes"
 len="3" dtype="INT" align="right" />
<hdr slbl="RLCT" llbl="LCT" scope="GBL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" />
<hdr slbl="LLCT" llbl="LLCT" scope="LCL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" />
<hdr slbl="RPLCT" llbl="PLCT" scope="GBL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" />
<hdr slbl="LPLCT" llbl="LPLCT" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="RPLCTT" llbl="PLCTTime" scope="GBL" sort="n" key="0"
 scroll="yes" len="7" dtype="INT" align="right" />
<hdr slbl="PLCTT" llbl="LPLCTTime" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="INT" align="right" />
<hdr slbl="LCP" llbl="LCPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="RNP" llbl="NPRI" scope="GBL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="LNP" llbl="LNPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="RLP" llbl="LPRI" scope="GBL" sort="n" key="0" scroll="yes"
```

```
|                          len="2" dtype="INT" align="right" />
|                          <hdr slbl="LLP" llbl="LLPRI" scope="LCL" sort="n" key="0" scroll="yes"
|                          len="2" dtype="INT" align="right" />
|                          <hdr slbl="RSSZ" llbl="SegSz" scope="GBL" sort="n" key="0" scroll="yes"
|                          len="5" dtype="INT" align="right" />
|                          <hdr slbl="LSSZ" llbl="LSegSz" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="RSNO" llbl="SegNo" scope="GBL" sort="n" key="0" scroll="yes"
|                          len="5" dtype="INT" align="right" />
|                          <hdr slbl="LSNO" llbl="LSegNo" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="RPLM" llbl="ParLim" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="LPLM" llbl="LParLim" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="RGC" llbl="LRegCnt" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="RMRG" llbl="MaxRgn" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="LMRG" llbl="LMaxRgn" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="REDTR" llbl="EditRtn" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="8" dtype="CHAR" align="left" />
|                          <hdr slbl="EDTR" llbl="LEditRtn" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="8" dtype="CHAR" align="left" />
|                          <hdr slbl="RFP" llbl="FP" scope="GBL" sort="n" key="0" scroll="yes"
|                          len="1" dtype="CHAR" align="left" />
|                          <hdr slbl="FP" llbl="LFP" scope="LCL" sort="n" key="0" scroll="yes"
|                          len="1" dtype="CHAR" align="left" />
|                          <hdr slbl="REMHBS" llbl="EMHBSz" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="EMHBS" llbl="LEMHBSz" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="RCMTM" llbl="CmtMode" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="4" dtype="CHAR" align="left" />
|                          <hdr slbl="CMTM" llbl="LCmtMode" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="4" dtype="CHAR" align="left" />
|                          <hdr slbl="RMSGT" llbl="MsgType" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="7" dtype="CHAR" align="left" />
|                          <hdr slbl="MSGT" llbl="LMsgType" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="7" dtype="CHAR" align="left" />
|                          <hdr slbl="RSPATR" llbl="SPATrunc" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="1" dtype="CHAR" align="right" />
|                          <hdr slbl="SPATR" llbl="LSPATrunc" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="1" dtype="CHAR" align="right" />
|                          <hdr slbl="RSPASZ" llbl="SPASz" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="SPASZ" llbl="LSPASz" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="5" dtype="INT" align="right" />
|                          <hdr slbl="RSIDR" llbl="SIDR" scope="GBL" sort="n" key="0" scroll="yes"
|                          len="4" dtype="INT" align="right" />
|                          <hdr slbl="SIDR" llbl="LSIDR" scope="LCL" sort="n" key="0" scroll="yes"
|                          len="4" dtype="INT" align="right" />
|                          <hdr slbl="RSIDL" llbl="SIDL" scope="GBL" sort="n" key="0" scroll="yes"
|                          len="4" dtype="INT" align="right" />
|                          <hdr slbl="SIDL" llbl="LSIDL" scope="LCL" sort="n" key="0" scroll="yes"
|                          len="4" dtype="INT" align="right" />
|                          <hdr slbl="RDCLW" llbl="DCLWA" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="1" dtype="CHAR" align="left" />
|                          <hdr slbl="DCLW" llbl="LDCLWA" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="1" dtype="CHAR" align="left" />
|                          <hdr slbl="RDRRT" llbl="DirRoute" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="1" dtype="CHAR" align="left" />
|                          <hdr slbl="DRRT" llbl="LDirRoute" scope="LCL" sort="n" key="0"
|                          scroll="yes" len="1" dtype="CHAR" align="left" />
|                          <hdr slbl="REDTT" llbl="EditUC" scope="GBL" sort="n" key="0"
|                          scroll="yes" len="1" dtype="CHAR" align="left" />
```

```
|                              <hdr slbl="EDTT" llbl="LEditUC" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RINQ" llbl="Inq" scope="GBL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="INQ" llbl="LInq" scope="LCL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RRCV" llbl="Recover" scope="GBL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RCV" llbl="LRecover" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RRSP" llbl="Resp" scope="GBL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RSP" llbl="LResp" scope="LCL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RRMT" llbl="Remote" scope="GBL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RMT" llbl="LRemote" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RSER" llbl="Serial" scope="GBL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="SER" llbl="LSerial" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RWFI" llbl="WFI" scope="GBL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="WFI" llbl="LWFI" scope="LCL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RAOCMD" llbl="AOCMD" scope="GBL" sort="n" key="0"
|                               scroll="yes" len="4" dtype="CHAR" align="left" />
|                              <hdr slbl="AOCMD" llbl="LAOCMD" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="4" dtype="CHAR" align="left" />
|                              <hdr slbl="RCONV" llbl="Conv" scope="GBL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="CONV" llbl="LConv" scope="LCL" sort="n" key="0" scroll="yes"
|                               len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="RTLS" llbl="TranStat" scope="GBL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="TLS" llbl="LTranStat" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="1" dtype="CHAR" align="left" />
|                              <hdr slbl="MSN" llbl="LMSName" scope="LCL" sort="n" key="0"
|                              scroll="yes" len="8" dtype="CHAR" align="left" />
|                              <hdr slbl="RTMCR" llbl="TimeCreate" scope="GBL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" />
|                              <hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" />
|                              <hdr slbl="RTMUP" llbl="TimeUpdate" scope="GBL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" />
|                              <hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" />
|                              <hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
|                              <hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="20" dtype="CHAR" align="left" />
|                              <hdr slbl="REXPRT" llbl="ExprTm" scope="GBL" sort="n" key="0"
|                               scroll="yes" len="5" dtype="INT" align="right" />
|                              <hdr slbl="EXPRT" llbl="LExprTm" scope="LCL" sort="n" key="0"
|                               scroll="yes" len="5" dtype="INT" align="right" />
|                              </cmdrsphdr>
|                              <cmdrspdata>
|                              <rsp>TRAN(PARTROOT) MBR(IMS1    ) CC(   0) PSB(TPARTAPP) LCLS(  1)
|                               LLCT(    2) LPLCT(65535) LCP( 7) LNP( 7) LLP(10) LSSZ(   0) LSNO(
|                               0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N)
|                               DCLW(Y) DRRT(N) EDTT(Y) EMHBS(    0) EXPRT(    0) FP(N) IMSID(IMS1
|                              ) INQ(N) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(N) SER(N)
|                               SIDL(  10) SIDR(  10) SPASZ(    0) TMCR(2011.189 19:03:17.76) TLS(N)
|                               WFI(N) </rsp>
|                              <rsp>TRAN(PART    ) MBR(IMS1    ) CC(   0) PSB(DFSSAM02) LCLS(  4)
|                               LLCT(    2) LPLCT(65535) LCP( 7) LNP( 7) LLP(10) LSSZ(   0) LSNO(
```

```
          0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(MULT) CONV(N)
          DCLW(Y) DRRT(N) EDTT(Y) EMHBS(    0) EXPRT(  100) FP(N) IMSID(IMS1
          ) INQ(Y) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(N) SER(N)
          SIDL( 10) SIDR( 10) SPASZ(    0) TMAC(2011.189 19:04:19.92)
          TMUP(2011.189 19:04:25.07) TMCR(2011.189 19:03:17.76) TLS(N) WFI(N)
          </rsp>
          <rsp>TRAN(PART    ) MBR(IMS1    ) CC(   0) REPO(Y) RAOCMD(N   ) RCLS(
          4) RCONV(N) RCMTM(MULT) RDCLW(Y) RDRRT(N) REDTT(Y) REMHBS(     0)
          REXPRT(  100) RFP(N) RINQ(Y) RLCT(    2) RLP(10) RMRG(     0)
          RMSGT(MULTSEG) RNP( 7) RPLM(65535) RPLCT(65535) RPLCTT(6553500)
          RPSB(DFSSAM02) RRCV(Y) RRMT(N) RRSP(N) RSNO(    0) RSSZ(    0) RSER(N)
          RSIDL(    0) RSIDR(    0) RSPASZ(    0) RTMCR(2011.189 19:03:17.76)
          RTLS(N) RWFI(N) </rsp>
          <rsp>TRAN(PARTROOT) MBR(IMS1    ) CC(   0) REPO(Y) RAOCMD(N   ) RCLS(
          1) RCONV(N) RCMTM(SNGL) RDCLW(Y) RDRRT(N) REDTT(Y) REMHBS(     0)
          REXPRT(   0) RFP(N) RINQ(N) RLCT(    2) RLP(10) RMRG(     0)
          RMSGT(MULTSEG) RNP( 7) RPLM(65535) RPLCT(65535) RPLCTT(6553500)
          RPSB(TPARTAPP) RRCV(Y) RRMT(N) RRSP(N) RSNO(     0) RSSZ(    0) RSER(N)
          RSIDL(    0) RSIDR(    0) RSPASZ(    0) RTMCR(2011.189 19:03:17.76)
          RTLS(N) RWFI(N) </rsp>
          <rsp>TRAN(PART    ) MBR(IMS2    ) CC(   0) PSB(DFSSAM02) LCLS( 99)
          LLCT(    2) LPLCT(65535) LCP( 7) LNP( 7) LLP(10) LSSZ(    0) LSNO(
          0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(MULT) CONV(N)
          DCLW(Y) DRRT(N) EDTT(Y) EMHBS(    0) EXPRT(  100) FP(N) IMSID(IMS2
          ) INQ(Y) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(N) SER(N)
          SIDL( 10) SIDR( 10) SPASZ(    0) TMAC(2011.189 19:04:19.92)
          TMUP(2011.189 19:11:47.65) TMCR(2011.189 19:01:55.17) TLS(N) WFI(N)
          </rsp>
          <rsp>TRAN(PARTROOT) MBR(IMS2    ) CC(   0) PSB(TPARTAPP) LCLS( 1)
          LLCT(    2) LPLCT(65535) LCP( 7) LNP( 7) LLP(10) LSSZ(    0) LSNO(
          0) LPLM(65535) RGC(    0) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N)
          DCLW(Y) DRRT(N) EDTT(Y) EMHBS(    0) EXPRT(    0) FP(N) IMSID(IMS2
          ) INQ(N) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(N) SER(N)
          SIDL( 10) SIDR( 10) SPASZ(    0) TMCR(2011.189 19:01:55.17) TLS(N)
          WFI(N) </rsp>
          </cmdrspdata>
          </imsout>
```

Explanation: A line is returned for each resource that matches the wildcard name.
The resource definitions from each IMS that has the resource defined and the
global repository definition are returned. The repository information is returned by
the command master IMS. There are no IMS specific sections in the repository for
each resource name that matches the wildcard name.

Example 7 for QUERY TRAN command

TSO SPOC input:
```
QUERY TRAN NAME(*) SHOW(EXPORTNEEDED)
```

TSO SPOC output:
```
Trancode MbrName    CC ExportNeeded
TRN00002 IMS1        0 Y
```

OM API input:
```
CMD(QUERY TRAN SHOW(EXPORTNEEDED)
```

Related concepts:

➡️ How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

➡️ /DISPLAY MODIFY command (Commands)

QUERY TRANDESC command

Use the QUERY TRANDESC command to query information about transaction descriptors.

A descriptor is a model that can be used to create descriptors or resources.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 668
- "Usage notes" on page 675
- "Output fields" on page 676
- "Return, reason, and completion codes" on page 689
- "Examples" on page 690

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 209. Valid environments for the QUERY TRANDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY TRANDESC	X		X
NAME	X		X
SHOW	X		X
DEFAULT	X		X

Syntax

```
                     ┌─────────────<,─────────────┐                    ┌──DEFAULT(─┬─N─┬─)──┐
►►─────────────────────────────────────────────────────────────────────┴───────────┴─┴─Y─┴──┴──────►◄
                  ┌─SHOW(─▼─┬─ALL───────┬─)─┐
                  │        ├─AOCMD──────┤    │
                  │        ├─CLASS──────┤    │
                  │        ├─CMTMODE────┤    │
                  │        ├─CONV───────┤    │
                  │        ├─CPRI───────┤    │
                  │        ├─DCLWA──────┤    │
                  │        ├─DEFAULT────┤    │
                  │        ├─DEFN───────┤    │
                  │        ├─DEFNTYPE───┤    │
                  │        ├─DIRROUTE───┤    │
                  │        ├─EDITRTN────┤    │
                  │        ├─EDITUC─────┤    │
                  │        ├─EMHBSZ─────┤    │
                  │        ├─EXPRTIME───┤    │
                  │        ├─FP─────────┤    │
                  │        ├─GLOBAL─────┤    │
                  │        ├─IMSID──────┤    │
                  │        ├─INQ────────┤    │
                  │        ├─LCT────────┤    │
                  │        ├─LOCAL──────┤    │
                  │        ├─LPRI───────┤    │
                  │        ├─MAXRGN─────┤    │
                  │        ├─MODEL──────┤    │
                  │        ├─MSGTYPE────┤    │
                  │        ├─MSNAME─────┤    │
                  │        ├─NPRI───────┤    │
                  │        ├─PARLIM─────┤    │
                  │        ├─PGM────────┤    │
                  │        ├─PLCT───────┤    │
                  │        ├─PLCTTIME───┤    │
                  │        ├─PSB────────┤    │
                  │        ├─RECOVER────┤    │
                  │        ├─REMOTE─────┤    │
                  │        ├─RESP───────┤    │
                  │        ├─SEGNO──────┤    │
                  │        ├─SEGSZ──────┤    │
                  │        ├─SERIAL─────┤    │
                  │        ├─SPASZ──────┤    │
                  │        ├─SPATRUNC───┤    │
                  │        ├─TIMESTAMP──┤    │
                  │        ├─TRANSTAT───┤    │
                  │        └─WFI────────┘    │
                  └─SHOW(EXPORTNEEDED)───────┘
```
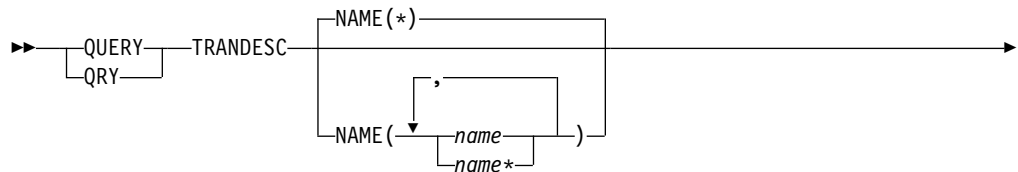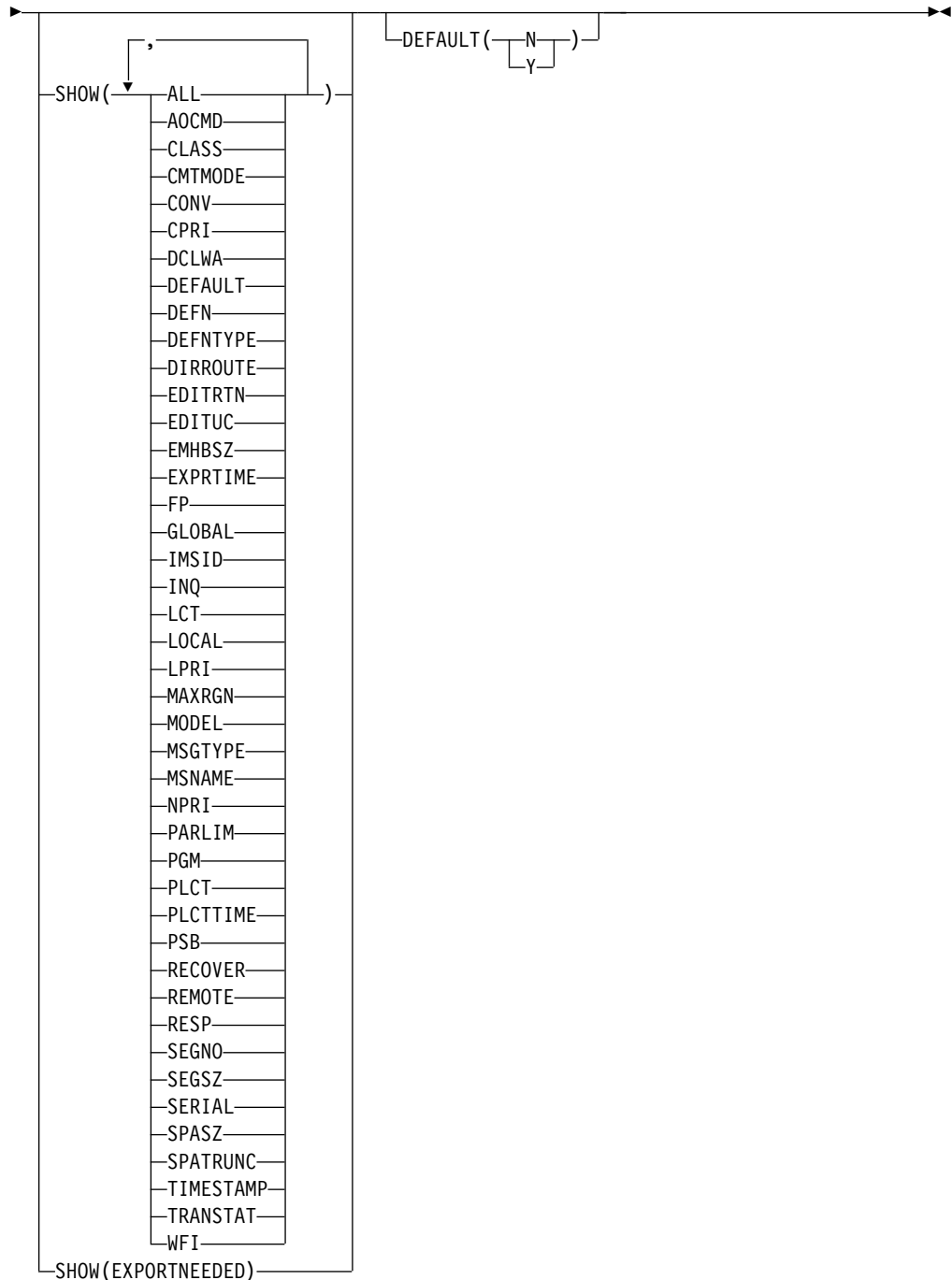
Keywords

The following keywords are valid for the QUERY TRANDESC command:

DEFAULT
  Selects transaction descriptors for display that possess the default value
  specified. DEFAULT(Y) displays the only default descriptor. DEFAULT(N)
  displays all the other descriptors that are not the default.

NAME
  Specifies the 1-8 character name of the transaction descriptor. Wildcards can be

specified in the name. The name is a repeatable parameter. The default is NAME(*), which returns all transaction descriptors.

SHOW
Specifies the transaction descriptor output fields to be returned. The transaction descriptor name is always returned along with the name of the IMS that created the output and the completion code. The filters that are supported with the SHOW keyword are:

ALL
Returns all information about the transaction descriptor itself.

Restriction: The `ExportNeeded` column, which is returned when SHOW(EXPORTNEEDED) is specified, is not returned with SHOW(ALL).

AOCMD
Specifies that you want the AOI option returned which indicates whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call. If AOCMD is defined as CMD, TRAN, or Y, and the AOI1 execution parameter is defined as AOI1=N, no authorization checking is done and the transaction is permitted to issue CMD and ICMD calls.

CLASS
Scheduling class used to select a transaction for scheduling. A transaction can be scheduled if there is a message processing region available for that class.

CMTMODE
Specifies when database updates and non-express output messages are committed. This operand affects emergency restart.

CONV
Conversation option indicates whether the transaction is conversational.

CPRI
The current priority. The current priority is the normal priority, when the transaction queue count is less than the limit count. The current priority is raised to the limit priority if the transaction queue count is equal to, or exceeds, the limit count. The current priority for transaction descriptors is always 0 because the current priority is a runtime scheduling value that does not apply to transaction descriptors, which are never scheduled.

DCLWA
Log write-ahead option.

DEFAULT
Default descriptor option.

N The descriptor is not the default.

Y The descriptor is the default. When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a default transaction descriptor called DFSDSTR1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor.

DEFN
Specifies that the resource definitions are to be returned.

The transaction descriptor attributes that can be returned are: AOCMD, CLASS, CMTMODE, CONV, DCLWA, DEFAULT, DIRROUTE, EDITUC, EDITRTN, EMHBSZ, EXPRTIME, FP, INQ, LCT, LPRI, MAXRGN, MSGTYPE, NPRI, PLCT, PLCTTIME, PARLIM, PGM, RECOVER, REMOTE, RESP, SERIAL, SIDR, SIDL, SEGNO, SEGSZ, SPASZ, SPATRUNC, TRANSTAT, WFI, the repository create and update time stamps, the IMS runtime create, update, import and access time stamps, and the IMS runtime MSNAME.

If SHOW(DEFN) is specified without any other SHOW filters or with the IMSID filter, all the definitional attributes, including those defined globally in the repository and those defined locally in the IMS system, are returned. The runtime resource definitions from the IMS system are returned by each IMS that receives the command. The stored resource definitions in the IMSRSC repository are returned by the command master IMS if the command master IMS is enabled to use the repository.

The command master IMS returns a response line for each generic stored resource definition obtained from the repository. This response line displays the attributes of the generic resource definition. When SHOW(DEFN) is specified without the IMSID filter and all the IMS systems have the same attribute values defined, only the response line for the generic definition is returned. The IMS IDs of the IMS systems that have the stored resource definition defined are not returned. If an IMS system has a stored resource definition with one or more attribute values that differ from the generic stored resource definition, an additional response line is returned for each IMS that has different attribute values.

If SHOW(DEFN,LOCAL) is specified, the runtime resource definitions from the IMS system are returned by each IMS that received the command.

If SHOW(DEFN,GLOBAL) is specified, the stored resource definitions from the repository are returned by the command master IMS. SHOW(DEFN,GLOBAL) is valid only when the command master IMS is enabled to use the repository.

If SHOW(DEFN) is specified with other parameters, only the requested definitional attributes are returned. For example, if SHOW(DEFN,TIMESTAMP) is specified, only the time stamps are returned.

Restrictions:
- SHOW(DEFN) cannot be specified with DEFNTYPE or MODEL.
- The LModelName, LModelType, and LDefnType columns, which are returned on the QRY TRANDESC SHOW(ALL) command, are not returned with SHOW(DEFN).
- The Repo and IMSid columns, which are returned with SHOW(DEFN), are not returned with SHOW(ALL).

If SHOW(DEFN,IMSID) is specified, a response line is returned for the generic stored resource definition, and an additional response line is returned for each IMS that has the resource defined in the repository, regardless of whether their stored resource definitions are the same as the generic resource definition.

When querying transaction descriptor information from the repository, resource definitions that are stored in the repository are used to determine the response lines with the repository information, and the runtime resource definitions are used to determine the response lines with the IMS

runtime resource information. The response lines are returned for each stored resource or runtime resource definition that matches the specified filter. If SHOW(DEFN,GLOBAL) is specified, only the stored resource definitions that match the specified filter are returned. If SHOW(DEFN,LOCAL) is specified, only the runtime resource definitions that match the specified filter are returned.

DEFNTYPE
  Definition type. This is how the descriptor or resource was defined.

DIRROUTE
  MSC directed routing option.

EDITRTN
  Input edit routine that edits messages before the program receives the message.

EDITUC
  The uppercase translation of input data.

EMHBSZ
  EMH buffer size required to run the Fast Path transaction.

EXPORTNEEDED
  Specifies that only the transaction descriptor resources that match the NAME parameter and have not been exported to the IMSRSC repository since they were created or last updated are to be returned on the QUERY command.

  If the NAME keyword is omitted or NAME(*) is specified, only the names of the transaction descriptors that need to be exported to the repository are returned.

  If a specific name or a generic name that is not NAME(*) is specified on the NAME keyword, a response line is returned for each transaction descriptor name specified on the command. If the specified transaction descriptor does not need to be exported to the repository, an N is returned in the ExportNeeded column.

  SHOW(EXPORTNEEDED) cannot be specified with any other SHOW parameters or other keywords and is valid only when IMS is enabled with the IMSRSC repository.

  The QUERY TRANDESC SHOW(EXPORTNEEDED) command is not valid on an XRF alternate or the RSR tracker.

  Tip: For resources and descriptors that have not been exported to the IMSRSC repository, you can issue the EXPORT DEFN TARGET(REPO) OPTION(CHANGESONLY) command to write the resource and descriptor definitions to the repository so that they are available for the next IMS cold start.

EXPRTIME
  Transaction expiration time in seconds.

FP Fast Path option.

GLOBAL
  Specifies that the stored resource definitions from the repository are to be returned.

If SHOW(GLOBAL,DEFN) is specified, the global resource definitions from the repository are returned by the command master IMS. SHOW(GLOBAL,DEFN) is valid only when the command master IMS is enabled to use the repository.

IMSID

Specifies that the IMS IDs of the IMS systems whose resource lists contain the specified resource name are to be returned. SHOW(IMSID) is processed only by the command master IMS and is valid only if the command master IMS is enabled to use the repository.

When SHOW(IMSID) is specified with the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the stored resource definitions.

When SHOW(IMSID) is specified without the DEFN filter, a separate line is returned for each IMS that has the resource defined, along with the resource name. No resource definitions are returned.

SHOW(IMSID) cannot be specified with any other SHOW filters other than DEFN and GLOBAL. If SHOW(IMSID,GLOBAL) is specified, GLOBAL is ignored; that is, SHOW(IMSID,GLOBAL) is treated as SHOW(IMSID). SHOW(DEFN,IMSID,LOCAL) is treated as SHOW(DEFN,LOCAL).

INQ

Inquiry option.

LCT

Limit count. Specifies the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction. The limit count value can range from 1 through 65 535.

The limit count value is ignored for a transaction processed by a BMP.

The limit count value is ignored in a shared-queues environment.

LOCAL

Specifies that the runtime resource definitions from the IMS system are to be returned.

SHOW(DEFN,LOCAL) returns only the local definitional attributes from the IMS system that processes the command.

LPRI

Limit priority. The scheduling priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class.

The limit priority value is ignored for a transaction processed by a BMP.

MAXRGN

Maximum region count. This limits the number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction. When the number of MPP regions is not limited, one transaction might monopolize all available regions. MAXRGN(0) means that no limit is imposed.

MODEL

The model name and model type used to create this resource or descriptor. If the descriptor or resource is created with one or more attributes defined and no model specified, the model name and model type is the default descriptor. The model name and model type are blank for the IMS-defined transaction descriptor DFSDSTR1. The CREATE command specified without the LIKE keyword creates a descriptor or resource using the default descriptor as a model. The default descriptor is either the IMS descriptor DFSDSTR1 or user-defined. The CREATE command specified with the LIKE keyword creates a descriptor or resource using a model. The descriptor or resource is created with all the same attributes as the model. Attributes set explicitly by the CREATE command override the model attributes. The model type can either be a descriptor (DESC) or a resource (RSC). The model name and model type are for reference only. The descriptor or resource attributes might not match the model, if attributes are overridden by CREATE or UPDATE command values, or the model is updated later. The model name and model type can be used to identify resources that were created with the same model. The model name and model type of a resource are exported and imported. The IMPORT command does not use the model name and model type when creating a resource.

MSGTYPE

Message segment type (single or multiple segment). It specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing.

If MSC-directed routing is used in a multiple IMS system configuration, IMS does not ensure that both the message and the transaction destined to process that message are either single segment or multiple segments.

MSNAME

The logical link path name, remote system ID, and local system ID are returned.

Logical link path name in a multiple IMS system configuration (MSC). A logical link path is a path between any two IMS systems. The IMS systems are identified by the remote system ID and the local system ID associated with the logical link path. The remote system ID identifies the system in which messages using this path are to be processed. The local system ID identifies this system.

The remote system ID (SIDR) identifies the IMS system on which the application program runs. A value of 0 means MSC is not enabled on this system. The local system ID and remote system ID are the same for local transactions.

The local system ID (SIDL) identifies the originating system to which responses are returned. A value of 0 means MSC is not enabled on this system. The local system ID and remote system ID are the same for local transactions.

NPRI

Normal scheduling priority. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The normal priority is assigned to the transaction as the scheduling

priority when the number of input transactions enqueued and waiting to be processed is less than the limit count value.

The normal priority value is ignored for a transaction processed by a BMP.

PARLIM

Parallel processing limit count. This is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. This is the threshold value to be used when the associated program is defined with a scheduling type of parallel. An additional region is scheduled whenever the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.

PARLIM(0) indicates that any input message can cause a new region to be scheduled. PARLIM(65535) indicates that parallel processing is disabled and IMS allows the transaction to be scheduled in only one region at a time.

PGM

The name of the program associated with this transaction. This name matches the PSB name in ACBLIB.

PLCT

Processing limit count. This is the maximum number of messages sent to the application program by the IMS for processing without reloading the application program.
- PLCT(0) means that a maximum of one message is sent to the application program at a single program load.
- PLCT(65535) means that no limit is placed on the number of messages processed at a single program load.

PLCTTIME

Processing limit count time. This is the amount of time (in hundredths of seconds) allowable to process a single transaction (or message). The number specifies the maximum CPU time allowed for each message to be processed in the message processing region.

PLCTTIME(6553500) means that no time limit is placed on the application program.

PSB

The name of the program associated with this transaction. This name matches the PSB name in ACBLIB.

RECOVER

Recovery option.

REMOTE

Remote option.

RESP

Response-mode option.

SEGNO

Segment number. This is the maximum number of application program output segments that are allowed into the message queues per Get Unique (GU) call from the application program.

SEGNO(0) means that the number of segments is not checked by the online system at execution time.

SEGSZ

Segment size. This is the maximum number of bytes allowed in any one output segment.

SEGSZ(0) means that the segment size is not checked by the online system at execution time.

SERIAL
Serial option.

SPASZ
Scratchpad area size for a conversational transaction.

SPATRUNC

The SPA data truncation option indicates whether the SPA data should be truncated or preserved across a program switch to a transaction that is defined with a smaller SPA.

When a conversation initially starts and a program switch occurs, the SPATRUNC option is checked and set or reset as specified. When the option is set, it remains set for the life of the conversation, or until a program switch occurs to a transaction that specifies the option is to be reset.

When a program switch occurs, the truncated data option for the new transaction is first checked, and that specification is set for the conversation and is used for the SPA inserted into the output message. If the option is not specified for the new transaction, the option currently in effect for the conversation is used.

TIMESTAMP
The creation time (TIMECREATE), last update time (TIMEUPDATE), last access time (TIMEACCESS), and last import time (TIMEIMPORT) time stamps are returned. The time is returned in local time in the format YYYY.JJJ HH:MM:SS.TH, where:
- YYYY is the year.
- JJJ is the Julian day (001 - 365).
- HH is the hour (01 - 24).
- MM is the minute (00 - 59).
- SS is the seconds (00 - 59).
- TH is the tenths and hundredths of a second (00 - 99).

TRANSTAT
Transaction level statistics option.

WFI
Wait-for-input option.

Usage notes

This command can be issued only through the Operations Manager API. This command applies to DB/DC and DCCTL systems.

This command is allowed on XRF alternate and RSR tracker systems.

The QUERY TRANDESC command is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

If you want to display information about resource definitions, specify SHOW(DEFN). If you want to know which IMS systems have the resource defined and also know the attributes or resource definitions at each IMS system, specify SHOW(DEFN,IMSID). If you want to know which IMS systems have the resource defined, specify SHOW(IMSID).

Output fields

The following table shows the QUERY TRANDESC output fields. The columns in the table are as follows:

Short label
　　　Contains the short label generated in the XML output.

Long label
　　　Contains the long label generated in the XML output.

Keyword
　　　Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope　Identifies the scope of the output field.

Meaning
　　　Provides a brief description of the output field.

Table 210. Output fields for the QUERY TRANDESC command

Short label	Long label	Keyword	Scope	Meaning
AOCMD	LAOCMD	AOCMD	LCL	Indicates whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call. The output value is obtained from the local IMS.
				CMD Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execution parameter is defined as C, R, or A, authorization checking is based on which transactions can issue a particular command. In this case, the commands (or the first three characters of the commands) need to be defined to RACF or an equivalent product as a user. The type-1 AOI transactions must be defined as profiles under the TIMS class, and for each transaction, the commands it can issue must be specified.
				N Indicates that the transaction is not permitted to issue type-1 AOI CMD calls. The transaction is permitted to issue type-2 AOI ICMD calls.
				TRAN Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execution parameter is defined as C, R, or A, the transaction code is used for authorization. The first authorization check results in the accessor environment element (ACEE) being built. This environment is kept for use by future authorization checks. The type-1 AOI transaction needs to be defined to RACF or an equivalent product as a user. The transactions will then be specified on RACF PERMIT statements for each command they are allowed to issue from a type-1 AOI transaction. Specifying AOI transactions as users to RACF might conflict with the name of a user already defined to RACF. If this conflict occurs, then either the transaction name or the existing user name needs to be changed.
				Y Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the user ID or the program name is used for authorization. For some environments, if a Get Unique call has not yet occurred, the program name is used for authorization.
CCTXT	CCText	*error*	LCL	Completion code text that briefly explains the meaning of the nonzero completion code.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
CMTM	LCmtMode	CMTMODE	LCL	Commit mode for the transaction: commit after a single message (SNGL) or multiple messages (MULT). The output value is obtained from the local IMS.
				MULT Database updates and non-express output messages are committed only when the application program terminates normally, when the processing limit count has been reached, or, in the case of a pseudo-WFI dependent region, when there are no more messages on the queue. For example, if five transactions are processed during a single schedule of a program, all five are committed only when the fifth one is completed and the program terminates. Until a transaction has been committed, locks for updated database records are not released and non-express output messages are not queued for output. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing.
				SNGL Database updates and non-express output messages are committed when the application program completes processing each transaction. IMS invokes commit processing either when the application program requests the next message (issues a GU to the IO-PCB), or when the application program terminates. If an application ends abnormally before committing its message, emergency restart requeues the message that was in process before the abend and makes it available for reprocessing.
CONV	LConv	CONV	LCL	Conversation option. Transaction is conversational (Y), or not (N). The output value is obtained from the local IMS.
				N Transaction is not conversational.
				Y Transaction is conversational. The transaction message is destined for a conversational program. A conversational program processes transactions made up of several steps. A conversational program receives a message from a terminal, replies to the terminal, but saves the data from the transaction in a scratchpad area (SPA). When the person at the terminal enters more data, the program has the data it saved from the last message in the SPA, so it can continue processing the request without the person at the terminal having to enter the data again.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
DCLW	LDCLWA	DCLWA	LCL	Perform log write-ahead for recoverable, nonresponse mode input messages and transaction output messages (Y) or not (N). The output value is obtained from the local IMS.
				N IMS does not perform log write-ahead.
				Y IMS performs log write-ahead for recoverable, nonresponse input messages and transaction output messages. If not defined for the transaction, the default is the DCLWA parameter in the IMSCTRL macro. This ensures that a nonresponse input transaction is made recoverable across IMS failures, prior to IMS acknowledging receipt of the input.
				Database changes are made recoverable prior to IMS sending associated output reply messages.
				This ensures that information in the log buffers is written to the IMS log, before the associated input acknowledgment or output reply is sent to the terminal.
DESC	DescName	TRANDESC	LCL	Transaction descriptor name.
DFNT	LDefnType	DEFNTYPE	LCL	Definition type (CREATE, IMPORT, UPDATE).
DFLT	LDflt	TRANDESC	LCL	Default descriptor (Y) or not (N).
DRRT	LDirRoute	DIRROUTE	LCL	Supports MSC directed routing (Y) or not (N). The output value is obtained from the local IMS.
				N The application program processing a transaction is not informed of the system that originated the transaction. The name of the originating LTERM is placed in the I/O PCB.
				Y The application program processing a transaction is informed of the system that originated the transaction, if MSC directed routing is used in a multiple IMS system configuration. An MSNAME corresponding to a logical path back to the originating system is placed in the I/O PCB.
EDTR	LEditRtn	EDITRTN	LCL	Input edit routine name.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
EDTT	LEditUC	EDITUC	LCL	Input data is to be translated to uppercase (Y) or not (N). The output value is obtained from the local IMS.
				N Input data is not translated to uppercase. It can consist of uppercase and lowercase characters as entered from the terminal.
				Y Input data is to be translated to uppercase before it is presented to the processing program. If FP(Y), the transaction is to be translated to uppercase before being presented to the edit/routing exit routine.
				Specifying EDITUC(Y) for VTAM terminals prevents the transmission of embedded device control characters.
EMHBS	LEMHBSz	EMHBSZ	LCL	EMH buffer size. The output value is obtained from the local IMS.
EXPN	ExportNeeded	EXPORTNEEDED	LCL	Indicates whether the transaction descriptor has been exported to the IMSRSC repository. The value can be Y or N.
EXPRT	LExprTm	EXPRTIME	LCL	Transaction expiration time. The output value is obtained from the local IMS.
FP	LFP	FP	LCL	Fast Path potential candidate (P), Fast Path exclusive (E), or FP option not enabled (N). The output value is obtained from the local IMS.
				E Fast Path exclusive transaction. Any message for this transaction is always routed to a Fast Path application program.
				N Fast Path option is not enabled. When FP(N) is specified, any attempt to use Fast Path resources or commands will yield unpredictable results.
				P Fast Path potential transaction. Any message for this transaction can potentially be routed to a Fast Path application program.
IMSID	IMSid	IMSID	GBL	The IMSIDs that have the resource defined. The output values are obtained from the repository.
INQ	LInq	INQ	LCL	Inquiry transaction (Y) or not (N). The output value is obtained from the local IMS.
				N Inquiry option is disabled.
				Y Inquiry option is enabled. This is an inquiry transaction that, when entered, does not cause a change in any database. Programs are prohibited from issuing ISRT, DLET, or REPL calls to a database when scheduled to process a transaction defined as INQ(Y).
				An application program cannot do an SQL INSERT, DELETE, or UPDATE when the IMS transaction is defined with INQ(Y).

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
LCLS	LCls	CLASS	LCL	Scheduling class used to determine which message regions can process the transaction locally on a particular IMS.
LCP	LCPRI	CPRI	LCL	Local current scheduling priority. The current scheduling priority is used to calculate which transaction is selected for scheduling.
LLCT	LLCT	LCT	LCL	Limit count in the local IMS. The limit count is the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction.
LLP	LLPRI	LPRI	LCL	Local limit scheduling priority. The limit scheduling priority is the priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value.
LMRG	LMaxRgn	MAXRGN	LCL	Local maximum region count. The maximum region count is the maximum number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction that is eligible for parallel scheduling.
LNP	LNPRI	NPRI	LCL	Local normal scheduling priority. The normal scheduling priority is the priority assigned to this transaction when the number of input transactions enqueued and waiting to be processed is less than the limit count value.
LPLCT	LPLCT	PLCT	LCL	Local processing limit count. The processing limit count is the number of transaction messages a program can process in a single scheduling.
LPLM	LParLim	PARLIM	LCL	Local parallel processing limit count. The parallel limit count is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. An additional message region is scheduled whenever the transaction queue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.
LSNO	LSegNo	SEGNO	LCL	Local application program output segment limit allowed in message queues for each GU call.
LSSZ	LSegSz	SEGSZ	LCL	Local application program output segment size limit allowed in the message queues for each GU call.
MDLN	LModelName	MODEL	LCL	Model name. Name of the resource or descriptor used as a model to create this descriptor. DFSDSTR1 is the IMS descriptor name for transactions. The output value is obtained from the local IMS.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
MDLT	LModelType	MODEL	LCL	Model type, either RSC or DESC. RSC means that the descriptor was created using another resource as a model. DESC means that the descriptor was created by using a descriptor as a model.
MSGT	LMsgType	MSGTYPE	LCL	Message type of single segment (SNGLSEG) or multiple segment (MULTSEG). The output value is obtained from the local IMS. **MULTSEG** Specifies that the incoming message can be more than one segment in length. It is not eligible for scheduling to an application program until an end-of-message indication is received, or a complete message is created by MFS. **SNGLSEG** Specifies that the incoming message is one segment in length. It becomes eligible for scheduling when the terminal operator indicates end-of-segment.
MSN	LMSName	MSNAME	LCL	Logical link path name.
PLCTT	LPLCTTime	PLCTTIME	LCL	Processing limit count time.
PSB	LPSBName	PGM or PSB	LCL	Program name associated with the transaction. The output value is obtained from the local IMS.
RAOCMD	LAOCMD	DEFN, AOCMD	GBL	Indicates whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call. The output value is obtained from the repository. For the values to be returned, see the description for "LAOCMD" in this table.
RCLS	Cls	DEFN, CLASS	GBL	Class value in the repository.
RCMTM	CmtMode	DEFN, CMTMODE	GBL	Commit mode for the transaction: commit after a single message (SNGL) or multiple messages (MULT). The output value is obtained from the repository.
RCONV	Conv	DEFN, CONV	GBL	Conversation option. Transaction is conversational (Y), or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LConv" in this table.
RCV	LRecover	RECOVER	LCL	Recovered during an IMS emergency or normal restart (Y) or not (N). The output value is obtained from the local IMS. **N** Recovery option is disabled. The transaction is not recovered. **Y** Recovery option is enabled. The transaction is recovered during IMS emergency or normal restart.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
RDCLW	DCLWA	DEFN, DCLWA	GBL	Perform log write-ahead for recoverable, nonresponse mode input messages and transaction output messages (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LDCLWA" in this table.
RDFLT	Dflt	DEFN	GBL	Default descriptor (Y) or not (N). The value is obtained from the repository.
RDRRT	DirRoute	DEFN, DIRROUTE	GBL	Supports MSC directed routing (Y) or not (N). The output value is obtained from the repository.
REDTR	EditRtn	DEFN, EDITRTN	GBL	Input edit routine name. The value is obtained from the repository.
REDTT	EditUC	DEFN, EDITUC	GBL	Input data is to be translated to uppercase (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LEditUC" in this table.
REMHBS	EMHBSz	DEFN, EMHBSZ	GBL	EMH buffer size. The output value is obtained from the repository.
REPO	Repo	DEFN	GBL	Indicates whether the output line contains the stored resource definitions. **Y** Indicates repository definitions. **(blank)** Indicates local definitions.
REXPRT	ExprTm	DEFN, EXPRTIME	GBL	Transaction expiration time. The output value is obtained from the repository.
RFP	FP	DEFN, FP	GBL	Fast Path potential candidate (P), Fast Path exclusive (E), or FP option not enabled (N). The output value is obtained from the repository. For the values to be returned, see the description for "LFP" in this table.
RGC	LRegCnt	RGC	LCL	Region count. The output value is obtained from the local IMS.
RINQ	Inq	DEFN, INQ	GBL	Inquiry transaction (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LInq" in this table.
RLCT	Lct	DEFN, LCT	GBL	Limit count value obtained from the repository.
RLP	LPRI	DEFN, LPRI	GBL	Local limit scheduling priority value in the repository. The limit scheduling priority is the priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value.
RMRG	RMaxRgn	DEFN, MAXRGN	GBL	Maximum region count obtained from the repository. The maximum region count is the maximum number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction that is eligible for parallel scheduling.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
RMSGT	MsgType	DEFN, MSGTYPE	GBL	Message type of single segment (SNGLSEG) or multiple segment (MULTSEG). The output value is obtained from the repository. For the values to be returned, see the description for "LMsgType" in this table.
RMT	LRemote	REMOTE	LCL	Remote transaction (Y) or not (N). The output value is obtained from the local IMS. **N** Local transaction. The transaction runs on the local system. **Y** Remote transaction. The transaction runs on a remote system.
RPLCT	PLCT	DEFN, PLCT	GBL	Processing limit count value in the repository. The processing limit count is the number of transaction messages a program can process in a single scheduling.
RPLCTT	PLCTTime	DEFN, PLCTTIME	GBL	Processing limit count time value in the repository.
RPLM	Parlim	DEFN, PARLIM	GBL	Parallel processing limit count value in the repository. The parallel limit count is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. An additional message region is scheduled whenever the transaction queue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.
RPSB	PsbName	DEFN, PGM	GBL	Program name associated with the transaction. The output value is obtained from the repository. For the values to be returned, see the description for "LPsbName" in this table.
RRCV	Recover	DEFN, RECOVER	GBL	Recovered during an IMS emergency or normal restart (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LRecover" in this table.
RRMT	Remote	DEFN, REMOTE	GBL	Remote transaction (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LRemote" in this table.
RRSP	Resp	DEFN, RESP	GBL	Response mode transaction (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LResp" in this table.
RSET	Serial	DEFN, SERIAL	GBL	Transaction is processed serially (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LSerial" in this table.
RSIDL	SIDL	DEFN, MSNAME	GBL	Local system ID. The output value is obtained from the repository.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
RSIDR	SIDR	DEFN, MSNAME	GBL	Remote system ID. The output value is obtained from the repository.
RSNO	SegNo	DEFN, SEGNO	GBL	Application program output segment limit allowed in message queues for each GU call. The value is obtained from the repository.
RSP	LResp	RESP	LCL	Response mode transaction (Y) or not (N). The output value is obtained from the local IMS. **N** Response mode option is disabled. For terminals specifying or accepting a default of OPTIONS=TRANRESP, input should not stop after this transaction is entered. **Y** Response mode option is enabled. For terminals specifying or accepting a default of OPTIONS=TRANRESP, no additional messages are to be allowed after this transaction is entered until this transaction sends a response message back to the terminal. Response mode can be forced or negated by individual terminal definition. RESP(Y) is ignored during online processing for all terminals that do not operate in response mode.
RSPASZ	SpaSz	DEFN, SPASZ	GBL	Conversational transaction scratchpad area size. The output value is obtained from the repository.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
RSPATR	SpaTrunc	DEFN, SPATRUNC	GBL	Conversational transaction SPA data should be truncated (R) or preserved (S) across a program switch to a transaction that is defined with a smaller SPA. The SPATRUNC value defined for the conversational transaction is stored in the repository. A QUERY TRANDESC command with SHOW(DEFN) returns a SPATRUNC value of R, S, or null from the repository values.
				S S is shown on the QUERY TRANDESC command for a conversational transaction in one of the following conditions:
				• If the transaction is defined with SPATRUNC=S on the CREATE TRAN or CREATE TRANDESC command
				• If SPA=STRUNC is specified on the TRANSACT macro
				• If the system-wide truncated data option is set as TRUNC=Y in the DFSDCxxx member
				R R is shown on the QUERY TRANDESC command for a conversational transaction in one of the following conditions:
				• If the transaction is defined with SPATRUNC=R on the CREATE TRAN or CREATE TRANDESC command
				• If SPA=RTRUNC is specified on the TRANSACT macro
				• If the system-wide truncated data option is not set as TRUNC=N in the DFSDCxxx member.
				null A null value indicates that the transaction does not have the SPATRUNC value defined and that the value is overridden with the system-wide truncated data option defined with the TRUNC= option on the DFSDCxxx member.
RSSSZ	SegSz	DEFN,SEGSZ	GBL	Application program output segment size limit allowed in the message queues for each GU call. The output value is obtained from the repository.
RTLS	TranStat	DEFN, TRANSTAT	GBL	Transaction level statistics logged (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for "LTranStat" in this table.
RTMCR	TimeCreate	DEFN, TIMESTAMP	GBL	Create time from the repository. This is the time that the resource was first created in the repository.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
RTMUP	TimeUpdate	DEFN, TIMESTAMP	GBL	Update time from the repository. This is the time that the resource was last updated in the repository.
RWFI	WFI	DEFN, WFI	GBL	Wait-for-input transaction (Y) or not (N). The output value is obtained from the repository. For the values to be returned, see the description for LWFI in this table.
SER	LSerial	SERIAL	LCL	Transaction is processed serially (Y) or not (N). The output value is obtained from the local IMS.
				N Serial option is disabled. Messages for the transaction are not processed serially. Message processing can be processed in parallel. Messages are placed on the suspend queue after a U3303 pseudoabend. Scheduling continues until repeated failures result in the transaction being stopped with a USTOP.
				Y Serial option is enabled. Messages for the transaction are processed serially. U3303 pseudoabends do not cause the message to be placed on the suspend queue but rather on the front of the transaction message queue, and the transaction is stopped with a USTOP. The USTOP of the transaction is removed when the transaction or the class is started with a /START command.
SIDL	LSIDL	MSNAME	LCL	Local system ID. The output value is obtained from the local IMS.
SIDR	LSIDR	MSNAME	LCL	Remote system ID. The output value is obtained from the local IMS.
SPASZ	LSPASz	SPASZ	LCL	Conversational transaction scratchpad area size. The output value is obtained from the local IMS.
SPATR	LSPATrunc	SPATRUNC	LCL	Conversational transaction SPA data should be truncated (R) or preserved (S) across a program switch to a transaction that is defined with a smaller SPA. The output value is obtained from the local IMS.
				S IMS preserves all the data in the SPA, even when a program switch is made to a transaction that is defined with a smaller SPA. The transaction with the smaller SPA does not see the truncated data, but when the transaction switches to a transaction with a larger SPA, the truncated data is used.
				R Truncated data is not preserved.
TLS	LTranStat	TRANSTAT	LCL	Transaction level statistics logged (Y) or not (N). The output value is obtained from the local IMS.
				N Transaction level statistics logging is not active.
				Y Transaction level statistics logging is active.

Table 210. Output fields for the QUERY TRANDESC command (continued)

Short label	Long label	Keyword	Scope	Meaning
TMAC	LTimeAccess	TIMESTAMP	LCL	The time that the descriptor was last accessed. The output value is obtained from the local IMS. The last access time is retained across warm start, emergency restart, EXPORT and IMPORT. The updating of the last access time is not logged. After a restart, the last access time reflects the time recorded in the restart checkpoint log records. For a transaction descriptor, the following action updates the last access time: • CREATE command or DFSINSX0 exit refers to the descriptor as a model.
TMCR	LTimeCreate	TIMESTAMP	LCL	The time that the descriptor was created. This is the result of a CREATE TRANDESC command, IMPORT command that creates the transaction descriptor, or IMS initialization. The create time is retained across warm start, emergency restart, EXPORT and IMPORT. The output value is obtained from the local IMS.
TMIM	LTimeImport	TIMESTAMP	LCL	The time that the descriptor was last imported. The import time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
TMUP	LTimeUpdate	TIMESTAMP	LCL	The last time the attributes of the runtime resource definition were updated as a result of the UPDATE TRANDESC command or the IMPORT command. The update time is retained across warm start and emergency restart. The output value is obtained from the local IMS.
WFI	LWFI	WFI	LCL	Wait-for-input transaction (Y) or not (N). The output value is obtained from the local IMS. **N** Wait-for-input option is disabled. **Y** Wait-for-input option is enabled. A message processing or batch processing application program that processes WFI transactions is scheduled and invoked normally. If the transaction to be processed is defined as WFI, the program is allowed to remain in main storage after it has processed the available input messages. The QC status code (no more messages) is returned to the program if the processing limit count is reached; a command is entered to change the status of the schedule transaction, database, program, or class; commands relating to the databases used by the transaction are entered; or IMS is terminated with a checkpoint shutdown.

Return, reason, and completion codes

Table 211. Return and reason codes for the QUERY TRANDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001010'	No descriptors were found to be returned. The descriptor names specified might be invalid, or: • There were no descriptors that match the filter specified. • If SHOW(EXPORTNEEDED) was specified, no resources needed to be exported to the IMSRSC repository.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'0000000C'	X'00003004'	No requests were successful.
X'00000010'	X'00004004'	No CQS address space.
X'00000010'	X'00004018'	No resource structure, or resource structure is not available.
X'00000010'	X'00004100'	Resource structure is full.
X'00000010'	X'00004104'	No RM address space.
X'00000010'	X'00004108'	No SCI address space.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004500'	IMS is not enabled to use the repository.
X'00000010'	X'00004501'	RM is not enabled to use the repository.
X'00000010'	X'00004502'	Repository is not available.
X'00000010'	X'00004503'	Repository is stopped.
X'00000010'	X'00004504'	Repository spare recovery is in progress.
X'00000010'	X'00004505'	No IMS resource list exists, or no resources for the resource type exist in the IMS resource list.
X'00000010'	X'00004507'	Access to the repository was denied.
X'00000010'	X'00004508'	Repository maximum put length exceeded.
X'00000010'	X'00004509'	RM data version is lower than the IMS data version.
X'00000010'	X'0000450A'	Repository Server (RS) is being shut down.
X'00000010'	X'0000450B'	RS is not available.
X'00000010'	X'0000450C'	RS is busy.
X'00000010'	X'0000450D'	RM failed to define some of the internal fields that are related to the IMSRSC repository.
X'00000014'	X'0000501C'	IMODULE GETMAIN error.
X'00000014'	X'00005100'	RM request error.
X'00000014'	X'00005104'	CQS error.
X'00000014'	X'00005108'	SCI request error.
X'00000014'	X'00005110'	Repository error.

Table 211. Return and reason codes for the QUERY TRANDESC command (continued)

Return code	Reason code	Meaning
X'00000014'	X'0000551C'	A request to view the data in the repository was rejected because a corrupted resource list was detected. The QUERY TRANDESC command is rejected because the resource list is unusable. Contact IBM Software Support. Send in the output from the command in error, and a dump of the local RM address space where the command executed or the dump of all RM address spaces in the IMSplex. Also send a copy of the IMSRSC repository RID and RMD data sets.

Table 212. Completion codes for the QUERY TRANDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for the transaction descriptor.
10	NO RESOURCES FOUND	Transaction descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.
90	INTERNAL ERROR	

Examples

The following are examples of the QUERY TRANDESC command:

Example 1 for QUERY TRANDESC command

TSO SPOC input:

```
QRY TRANDESC SHOW(ALL)
```

TSO SPOC output:

```
(screen 1)
DescName MbrName    CC LPSBname LCls  LLCT LPLCT LPLCTTime LCPRI LNPRI LLPRI
CONVDESC IMS1        0 DFSSAM04    1 65535 65535   6553500     0     1     1
DFSDSTR1 IMS1        0             1 65535 65535   6553500     1     1     1
FPEDESC  IMS1        0 EMHPSB2     1     0 65535   6553500     0     1     1
FPPDESC  IMS1        0 DFSSAM04    1 65535 65535   6553500     0     1     1
MSCDESC  IMS1        0 CPGM1B1     1 65535 65535   6553500     0     1     1
(scroll to the right screen 2)
DescName MbrName  LSegSz LSegNo LParLim LMaxRgn LEditRtn LFP LEMHBSz LCmtMode
CONVDESC IMS1          0      0   65535       0          N        0 SNGL
DFSDSTR1 IMS1          0      0   65535       0          N        0 SNGL
FPEDESC  IMS1          0      0   65535       0          E      200 SNGL
FPPDESC  IMS1          0      0   65535       0          P      256 SNGL
MSCDESC  IMS1          0      0   65535       0          N        0 SNGL
(scroll to the right screen 3)
DescName MbrName  LMsgType LSPATrunc LSPASz LSIDR LSIDL LDCLWA LDirRoute LEditUC
CONVDESC IMS1     MULTSEG          R    128    10    10 Y      N         Y
DFSDSTR1 IMS1     MULTSEG                 0    10    10 Y      N         Y
FPEDESC  IMS1     SNGLSEG                 0    10    10 Y      N         Y
FPPDESC  IMS1     SNGLSEG                 0    10    10 Y      N         Y
MSCDESC  IMS1     MULTSEG                 0    12    32 Y      N         Y
(scroll to the right screen 4)
DescName MbrName  LInq LRecover LResp LRemote LSerial LWFI LAOCMD LConv LTranStat
CONVDESC IMS1       N    Y        N     N       N       N    N      Y     N
```

```
DFSDSTR1 IMS1    N   Y       N   N       N       N   N       N   N
FPEDESC  IMS1    N   Y       Y   N       N       N   N       N   N
FPPDESC  IMS1    N   Y       Y   N       N       N   N       N   N
MSCDESC  IMS1    N   Y       N   Y       N       N   N       N   N
```
(scroll to the right screen 5)
```
DescName MbrName  LDflt LModelName LModelType LMSName  LTimeCreate          LTimeUpdate
CONVDESC IMS1     N     CDEBTRNA   RSC                 2011.181 18:15:27.52
DFSDSTR1 IMS1     Y                                    2011.181 15:22:55.07
FPEDESC  IMS1     N     DFSDSTR1   DESC                2011.181 18:21:51.51
FPPDESC  IMS1     N     DFSDSTR1   DESC                2011.181 18:25:10.73
MSCDESC  IMS1     N     DFSDSTR1   DESC       LINK31B4 2011.181 18:27:31.62
```
(scroll to the right screen 6)
```
DescName MbrName  LTimeAccess          LTimeImport
CONVDESC IMS1
DFSDSTR1 IMS1     2011.181 18:27:31.62
FPEDESC  IMS1
FPPDESC  IMS1
MSCDESC  IMS1
```
(scroll to the right screen 7)
```
DescName MbrName  LDefnType LExprTm
CONVDESC IMS1     CREATE         0
DFSDSTR1 IMS1     IMS            0
FPEDESC  IMS1     CREATE         0
FPPDESC  IMS1     CREATE         0
MSCDESC  IMS1     CREATE         0
```

OM API input:

```
CMD(QUERY TRANDESC SHOW(ALL))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.182 01:49:24.663483</statime>
<stotime>2011.182 01:49:24.664394</stotime>
<staseq>C800882E4F0BB385</staseq>
<stoseq>C800882E4F44A945</stoseq>
<rqsttkn1>USRT005 10184924</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>TRANDESC        </kwd>
<input>QRY TRANDESC SHOW(ALL) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="PSB" llbl="LPSBname" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="LCLS" llbl="LCls" scope="LCL" sort="n" key="0" scroll="yes"
 len="3" dtype="INT" align="right" />
<hdr slbl="LLCT" llbl="LLCT" scope="LCL" sort="n" key="0" scroll="yes"
 len="5" dtype="INT" align="right" />
<hdr slbl="LPLCT" llbl="LPLCT" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
```

```
<hdr slbl="PLCTT" llbl="LPLCTTime" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="INT" align="right" />
<hdr slbl="LCP" llbl="LCPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="LNP" llbl="LNPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="LLP" llbl="LLPRI" scope="LCL" sort="n" key="0" scroll="yes"
 len="2" dtype="INT" align="right" />
<hdr slbl="LSSZ" llbl="LSegSz" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="LSNO" llbl="LSegNo" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="LPLM" llbl="LParLim" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="LMRG" llbl="LMaxRgn" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="EDTR" llbl="LEditRtn" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="FP" llbl="LFP" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="EMHBS" llbl="LEMHBSz" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="CMTM" llbl="LCmtMode" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="MSGT" llbl="LMsgType" scope="LCL" sort="n" key="0"
 scroll="yes" len="7" dtype="CHAR" align="left" />
<hdr slbl="SPATR" llbl="LSPATrunc" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="right" />
<hdr slbl="SPASZ" llbl="LSPASz" scope="LCL" sort="n" key="0"
 scroll="yes" len="5" dtype="INT" align="right" />
<hdr slbl="SIDR" llbl="LSIDR" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="SIDL" llbl="LSIDL" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="DCLW" llbl="LDCLWA" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="DRRT" llbl="LDirRoute" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="EDTT" llbl="LEditUC" scope="LCL" sort="n" key="0"
scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="INQ" llbl="LInq" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="RCV" llbl="LRecover" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="RSP" llbl="LResp" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="RMT" llbl="LRemote" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="SER" llbl="LSerial" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="WFI" llbl="LWFI" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="AOCMD" llbl="LAOCMD" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="CONV" llbl="LConv" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="TLS" llbl="LTranStat" scope="LCL" sort="n" key="0"
 scroll="yes" len="1" dtype="CHAR" align="left" />
<hdr slbl="DFLT" llbl="LDflt" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="MDLN" llbl="LModelName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="MDLT" llbl="LModelType" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" />
<hdr slbl="MSN" llbl="LMSName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" />
<hdr slbl="TMCR" llbl="LTimeCreate" scope="LCL" sort="n" key="0"
```

```
|            scroll="yes" len="20" dtype="CHAR" align="left" />
|          <hdr slbl="TMUP" llbl="LTimeUpdate" scope="LCL" sort="n" key="0"
|            scroll="yes" len="20" dtype="CHAR" align="left" />
|          <hdr slbl="TMAC" llbl="LTimeAccess" scope="LCL" sort="n" key="0"
|            scroll="yes" len="20" dtype="CHAR" align="left" skipb="no" />
|          <hdr slbl="TMIM" llbl="LTimeImport" scope="LCL" sort="n" key="0"
|            scroll="yes" len="20" dtype="CHAR" align="left" />
|          <hdr slbl="DFNT" llbl="LDefnType" scope="LCL" sort="n" key="0"
|            scroll="yes" len="8" dtype="CHAR" align="left" />
|          <hdr slbl="EXPRT" llbl="LExprTm" scope="LCL" sort="n" key="0"
|            scroll="yes" len="5" dtype="INT" align="right" />
|        </cmdrsphdr>
|        <cmdrspdata>
|        <rsp>DESC(DFSDSTR1) MBR(IMS1    ) CC(   0) PSB(        ) LCLS(  1)
|          LLCT(65535) LPLCT(65535) LCP( 1) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
|          0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N) DCLW(Y)
|          DFNT(IMS) DFLT(Y) DRRT(N) EDTT(Y) EMHBS(   0) EXPRT(    0) FP(N)
|          INQ(N) MSGT(MULTSEG) PLCTT(6553500) RCV(Y) RMT(N) RSP(N) SER(N) SIDL(
|          10) SIDR( 10) SPASZ(    0) TMAC(2011.181 18:27:31.62) TMCR(2011.181
|          15:22:55.07) TLS(N) WFI(N) </rsp>
|        <rsp>DESC(FPPDESC ) MBR(IMS1    ) CC(   0) PSB(DFSSAM04) LCLS(  1)
|          LLCT(65535) LPLCT(65535) LCP( 0) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
|          0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N) DCLW(Y)
|          DFNT(CREATE) DFLT(N) DRRT(N) EDTT(Y) EMHBS( 256) EXPRT(    0) FP(P)
|          INQ(N) MDLT(DESC) MDLN(DFSDSTR1) MSGT(SNGLSEG) PLCTT(6553500) RCV(Y)
|          RMT(N) RSP(Y) SER(N) SIDL( 10) SIDR( 10) SPASZ(    0) TMCR(2011.181
|          18:25:10.73) TLS(N) WFI(N) </rsp>
|        <rsp>DESC(CONVDESC) MBR(IMS1    ) CC(   0) PSB(DFSSAM04) LCLS(  1)
|          LLCT(65535) LPLCT(65535) LCP( 0) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
|          0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(Y) DCLW(Y)
|          DFNT(CREATE) DFLT(N) DRRT(N) EDTT(Y) EMHBS(   0) EXPRT(    0) FP(N)
|          INQ(N) MDLT(RSC) MDLN(CDEBTRNA) MSGT(MULTSEG) PLCTT(6553500) RCV(Y)
|          RMT(N) RSP(N) SER(N) SIDL( 10) SIDR( 10) SPASZ( 128) SPATR(R)
|          TMCR(2011.181 18:15:27.52) TLS(N) WFI(N) </rsp>
|        <rsp>DESC(MSCDESC ) MBR(IMS1    ) CC(   0) PSB(CPGM1B1 ) LCLS(  1)
|          LLCT(65535) LPLCT(65535) LCP( 0) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
|          0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N) DCLW(Y)
|          DFNT(CREATE) DFLT(N) DRRT(N) EDTT(Y) EMHBS(   0) EXPRT(    0) FP(N)
|          INQ(N) MDLT(DESC) MDLN(DFSDSTR1) MSGT(MULTSEG) PLCTT(6553500) RCV(Y)
|          RMT(Y) RSP(N) SER(N) SIDL( 32) SIDR( 12) MSN(LINK31B4) SPASZ(    0)
|          TMCR(2011.181 18:27:31.62) TLS(N) WFI(N) </rsp>
|        <rsp>DESC(FPEDESC ) MBR(IMS1    ) CC(   0) PSB(EMHPSB2 ) LCLS(  1)
|          LLCT(   0) LPLCT(65535) LCP( 0) LNP( 1) LLP( 1) LSSZ(    0) LSNO(
|          0) LPLM(65535) LMRG(    0) AOCMD(N) CMTM(SNGL) CONV(N) DCLW(Y)
|          DFNT(CREATE) DFLT(N) DRRT(N) EDTT(Y) EMHBS( 200) EXPRT(    0) FP(E)
|          INQ(N) MDLT(DESC) MDLN(DFSDSTR1) MSGT(SNGLSEG) PLCTT(6553500) RCV(Y)
|          RMT(N) RSP(Y) SER(N) SIDL( 10) SIDR( 10) SPASZ(    0) TMCR(2011.181
|          18:21:51.51) TLS(N) WFI(N) </rsp>
|        </cmdrspdata>
|      </imsout>
```

Explanation: All transaction descriptors are returned with all output fields. All the transaction descriptor output fields do not fit on one screen, so the user must scroll to the right for additional output fields. The transaction descriptor name and the member name that built the line of output are displayed on every screen. The fields that are blank are not applicable to the specified transaction descriptor. IMS defines descriptor DFSDSTR1 at IMS cold start time to contain the IMS default values for transactions. A few other descriptors were created dynamically with CREATE TRANDESC commands, including CONVDESC for conversational transactions, FPEDESC for FP exclusive transactions, FPPDESC for FP potential transactions, and MSCDESC for remote MSC transactions. DFSDSTR1 shows the last access time for TimeAccess, which was updated by the last CREATE TRAN or CREATE TRANDESC command that referred to it as the default model.

Example 2 for QUERY TRANDESC command

TSO SPOC input:

```
QRY TRANDESC NAME(*) SHOW(DEFN,CLASS,FP)
```

TSO SPOC output:

```
DescName MbrName   CC Repo IMSid    Cls LCls FP LFP
CONVDESC IMS1       0 Y             1        N
CONVDESC IMS1       0    IMS1           1    N
CONVDESC IMS2       0    IMS2           1    N
CONVDESC IMS3       0    IMS3           1    N
DFSDSTR1 IMS1       0    IMS1           1    N
DFSDSTR1 IMS2       0    IMS2           1    N
DFSDSTR1 IMS3       0    IMS3           1    N
FPEDESC  IMS1       0 Y             1        E
FPEDESC  IMS1       0    IMS1           1    E
FPEDESC  IMS2       0    IMS2           1    E
FPEDESC  IMS3       0    IMS3           1    E
FPPDESC  IMS1       0 Y             1        P
FPPDESC  IMS1       0    IMS1           1    P
FPPDESC  IMS2       0    IMS2           1    P
FPPDESC  IMS3       0    IMS3           1    P
MSCDESC  IMS1       0 Y             1        N
MSCDESC  IMS1       0 Y  IMS1       1        N
MSCDESC  IMS1       0 Y  IMS2       1        N
MSCDESC  IMS1       0 Y  IMS3       1        N
MSCDESC  IMS1       0 Y  IMS4       1        N
MSCDESC  IMS1       0    IMS1           1    N
MSCDESC  IMS2       0    IMS2           1    N
MSCDESC  IMS3       0    IMS3           1    N
```

OM API input:

```
CMD(QRY TRANDESC NAME(*) SHOW(DEFN,CLASS,FP))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.190 02:08:43.039351</statime>
<stotime>2011.190 02:08:43.120611</stotime>
<staseq>C80A9B6AB5A77887</staseq>
<stoseq>C80A9B6AC97E3A9A</stoseq>
<rqsttkn1>USRT005 10190843</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS1     </master>
<userid>USRT005 </userid>
<verb>QRY </verb>
<kwd>TRANDESC        </kwd>
<input>QRY TRANDESC NAME(*) SHOW(DEFN,CLASS,FP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="5" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
```

```
<hdr slbl="REPO" llbl="Repo" scope="LCL" sort="d" key="2" scroll="no"
len="1" dtype="CHAR" align="left" />
<hdr slbl="IMSID" llbl="IMSid" scope="GBL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="RCLS" llbl="Cls" scope="GBL" sort="n" key="0" scroll="yes"
 len="3" dtype="INT" align="right" />
<hdr slbl="LCLS" llbl="LCls" scope="LCL" sort="n" key="0" scroll="yes"
 len="3" dtype="INT" align="right" />
<hdr slbl="RFP" llbl="FP" scope="GBL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
<hdr slbl="FP" llbl="LFP" scope="LCL" sort="n" key="0" scroll="yes"
 len="1" dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DFSDSTR1) MBR(IMS1    ) CC(   0) LCLS(  1) FP(N) IMSID(IMS1
 ) </rsp>
<rsp>DESC(FPPDESC ) MBR(IMS1    ) CC(   0) LCLS(  1) FP(P) IMSID(IMS1
 ) </rsp>
<rsp>DESC(MSCDESC ) MBR(IMS1    ) CC(   0) LCLS(  1) FP(N) IMSID(IMS1
 ) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1    ) CC(   0) LCLS(  1) FP(E) IMSID(IMS1
 ) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1    ) CC(   0) REPO(Y) RCLS(  1) RFP(E)
</rsp>
<rsp>DESC(FPPDESC ) MBR(IMS1    ) CC(   0) REPO(Y) RCLS(  1) RFP(P)
</rsp>
<rsp>DESC(MSCDESC ) MBR(IMS1    ) CC(   0) REPO(Y) RCLS(  1) RFP(N)
</rsp>
<rsp>DESC(MSCDESC ) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS1    )
 RCLS(  1) RFP(N) </rsp>
<rsp>DESC(MSCDESC ) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS2    )
 RCLS(  1) RFP(N) </rsp>
<rsp>DESC(MSCDESC ) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS3    )
 RCLS(  1) RFP(N) </rsp>
<rsp>DESC(MSCDESC ) MBR(IMS1    ) CC(   0) REPO(Y) IMSID(IMS4    )
 RCLS(  1) RFP(N) </rsp>
<rsp>DESC(DFSDSTR1) MBR(IMS3    ) CC(   0) LCLS(  1) FP(N) IMSID(IMS3
 ) </rsp>
<rsp>DESC(FPPDESC ) MBR(IMS3    ) CC(   0) LCLS(  1) FP(P) IMSID(IMS3
 ) </rsp>
<rsp>DESC(MSCDESC ) MBR(IMS3    ) CC(   0) LCLS(  1) FP(N) IMSID(IMS3
 ) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS3    ) CC(   0) LCLS(  1) FP(E) IMSID(IMS3
 ) </rsp>
<rsp>DESC(DFSDSTR1) MBR(IMS2    ) CC(   0) LCLS(  1) FP(N) IMSID(IMS2
 ) </rsp>
<rsp>DESC(FPPDESC ) MBR(IMS2    ) CC(   0) LCLS(  1) FP(P) IMSID(IMS2
 ) </rsp>
<rsp>DESC(MSCDESC ) MBR(IMS2    ) CC(   0) LCLS(  1) FP(N) IMSID(IMS2
 ) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS2    ) CC(   0) LCLS(  1) FP(E) IMSID(IMS2
 ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The stored resource definitions and the runtime resource definitions
for the specified resources are returned. The Dflt (Default) column is returned
because the local IMS runtime definitions are returned to identify the default
descriptor. DFSDSTR1 is the default descriptor and is only at each IMS system. The
default descriptor definitions are not in the repository. The TRANDESC descriptor
exists only in the repository.

Example 3 for QUERY TRANDESC command

TSO SPOC input:

QUERY TRANDESC NAME(TRND*,NOTFN*) SHOW(EXPORTNEEDED)

TSO SPOC output:

```
DescName MbrName   CC CCText             ExportNeeded
NOTFN*   IMS1      10 NO RESOURCES FOUND
TRND0001 IMS1       0                    N
TRND0002 IMS1       0                    Y
```

OM API input:

```
CMD(QUERY TRANDESC SHOW(EXPORTNEEDED)
```

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUERY USER command

Use the QUERY USER command to display information about VTAM users across the IMSplex. A user is either a dynamic (ETO) user, or a static or dynamic (ETO) ISC subpool user. In this context, a user ID that is used for user or terminal security is not considered as a user, but rather as a user ID. This command can be specified only through the OM API and is valid on an XRF alternate.

Subsections:

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 213. Valid environments for the QUERY USER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY USER	X		X
NAME	X		X
SHOW	X		X
STATUS	X		X

Syntax

```
>>-+-QUERY-+--USER--+-NAME(*)-----------------------------+---------->
   '-QRY---'        |            ,<----------             |
                    |           |         |               |
                    '-NAME(--v--+-username-+-+-)-'
                               '-username*-'

>--+-SHOW(GLOBAL,LOCAL)-------------------------------+--+------------------------><
   |        .-GLOBAL,LOCAL,-.                         |  |   ,<---------------      |
   +-SHOW(--+---------------+--| A |-)-------+        |  '-STATUS(--v--+-ALLOC----+-)-'
   |                                         |                        +-CONV-----+
   |        ,<----------                     |                        +-DEADQ----+
   |       |         |                       |                        +-EXCL-----+
   '-SHOW(--v--+-GLOBAL-+-+--+--------+--)---'                         +-MFST-----+
              '-LOCAL--'    '-,-| A |-'                                +-PRST-----+
                                                                      +-RESP-----+
                                                                      +-RESPINP--+
                                                                      +-RESPINPFP+
                                                                      +-RM-------+
                                                                      +-RMACTIVE-+
                                                                      +-RMOWNED--+
                                                                      +-STATIC---+
                                                                      +-STOSGN---+
                                                                      '-TEST-----'
```

A:

```
          ,<-----------
         |           |
|--------v--+-ALL-------+------------------------------------------|
            +-AUTOLOGON-+
            +-CONV------+
            +-EMHQ------+
            +-ID--------+
            +-LTERM-----+
            +-NODE------+
            +-OWNER-----+
            +-PRESET----+
            +-QCNT------+
            +-RECOVERY--+
            +-STATUS----+
            +-USERID----+
            '-VERSION---'
```
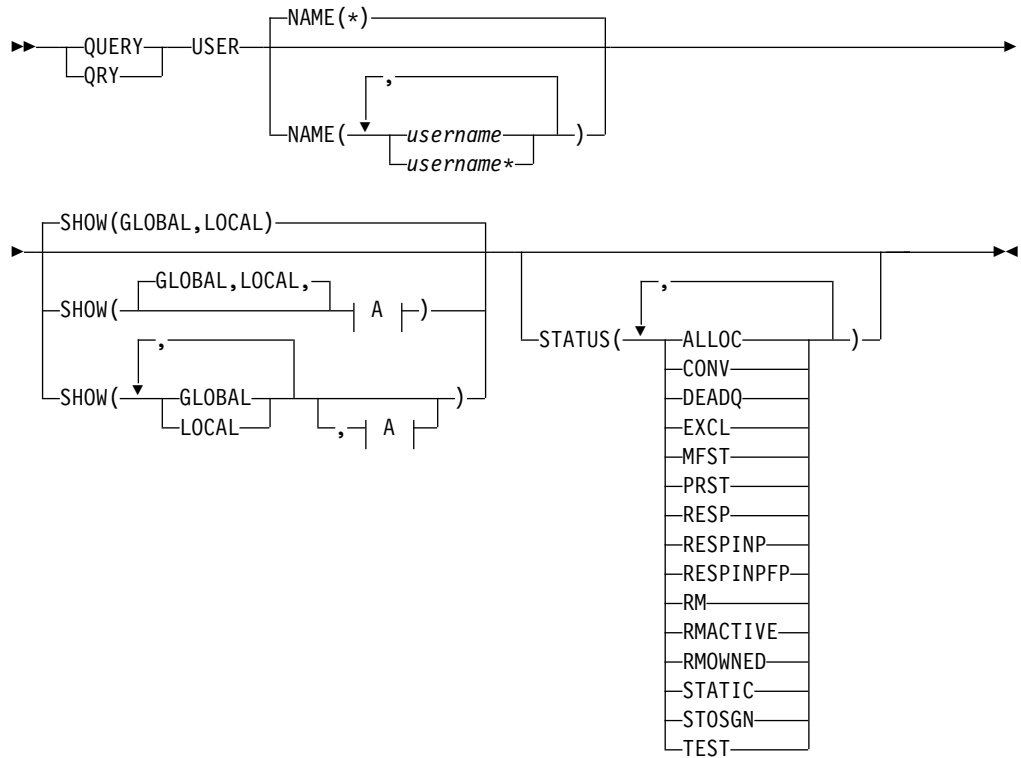
Keywords

The following keywords are valid for the QUERY USER command:

NAME()

Specifies the names of one or more users (dynamic user or ISC subpool user) that are to be displayed. Valid names are 1-8 characters, and wildcards can be specified. To display all users, specify NAME(*). NAME(*) is the default.

SHOW()

Specifies the user output fields to be returned. The user name is always returned along with the name of the IMS that created the output and the completion code. If SHOW is not specified, only the user names are returned, provided that the STATUS filter is not specified. This provides a method for a system management application to obtain a list of all user names that are currently known in the IMSplex.

Two parameters, GLOBAL and LOCAL, are used to specify the location (global resources or local resources) where IMS should obtain the information that is to be displayed. The default is both GLOBAL and LOCAL. The rest of the parameters are used to specify what information is displayed.

The parameters supported with the SHOW keyword, which can be specified in any order, are:

ALL

Returns all of the output fields, except for those fields displayed when the LTERM and CONV parameters are specified. To display lterm and conversation information with all other output, specify SHOW(ALL,LTERM,CONV).

AUTOLOGON

Displays the current autologon information for the specified user. This includes the name of the node with which to establish a session, the VTAM mode table name, the name of the node descriptor to be used to build the node, and (if ISC) the identifier of the other system half-session qualifier.

CONV

The conversation ID, transaction, and conversation status associated with the user. Each conversation is returned on a separate command response display line. Conversation status is not included when SHOW(ALL) is specified. To display conversation information with all other output, specify SHOW(ALL,CONV).

EMHQ

Displays the user message queue count in the Expedited Message Handler (EMH) queues. The queue count is the sum of the queue counts for each lterm associated with the specified user.

EMHQ is valid only when the GLOBAL parameter is specified on the SHOW keyword. If GLOBAL is not specified, then the EMHQ parameter is ignored.

EMHQ is processed by the command master only. It is ignored by all other IMS systems.

EMHQ is valid only if shared EMH is used in a shared queues environment. Otherwise, this parameter is ignored.

If the user resource exists in the resource structure, then the lterms associated with the global user resource are used to obtain the queue counts from the EMH queues. Otherwise, the lterms associated with the local user resource are used, but if the user does not exist locally on the command master, then the queue count is 0.

GLOBAL

When GLOBAL is specified, the command master displays global information, depending on the other SHOW parameters specified. This includes information from shared queues and the resource structure.

The GLOBAL parameter is processed by the command master only. All other IMS systems ignore this parameter. If LOCAL is not also specified, then all IMS systems other than the command master ignore the command.

GLOBAL is applicable only when the command master is using shared queues or sysplex terminal management (or both). GLOBAL is not applicable when the command master is not using shared queues or sysplex terminal management. In this environment, if LOCAL is also specified, then GLOBAL is ignored. Otherwise, the command master rejects the command.

If shared queues are enabled, and global queue counts are requested, then the command master will make requests to CQS to determine the appropriate queue counts. This includes both MSGQ and EMHQ.

If sysplex terminal management is enabled, then the command master will make requests to RM to determine the appropriate global status.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

ID Displays the other half-session qualifier name of the ISC node.

LOCAL
When LOCAL is specified, then all IMS systems including the command master display local information, depending on the other SHOW parameters specified. This includes information local to the IMS processing the command.

The LOCAL parameter is processed by all IMS systems, including the command master.

LOCAL is applicable in any environment, regardless of whether shared queues or sysplex terminal management are used.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line (or set of output lines), and local information is displayed as another output line (or set of output lines).

LTERM
Displays the logical terminal (lterm) names, if any, associated with the user. A user might have zero or more logical terminals associated with it. Each lterm associated with the user is returned on a separate command response line.

The lterm status is not included when SHOW(ALL) is specified. To display lterm information with all other output, specify SHOW(ALL,LTERM).

NODE
Displays the dynamic or ISC node that the user is associated with.

OWNER
Displays the owner of the user resource in the resource structure. This applies only when sysplex terminal management is enabled, and is processed by the command master only. All other IMS systems ignore this parameter.

The owner is the IMSID (or RSENAME for XRF systems) of the IMS system that owns the user. An IMS system owns a user resource if the resource is active (the user is signed on), or an IMS system is maintaining significant status for that resource.

PRESET

Displays the preset destination name for a user. A user is in preset destination mode following a /SET command. The preset destination name is either a transaction name or a logical terminal (LTERM) name. All messages entered from this user are sent to the preset destination transaction or LTERM.

QCNT

Displays the user message queue count. The queue count is the sum of the queue counts for each lterm associated with the specified user.

The local queue counts value returned on this command represents the messages being processed by the IMS system where this command is issued. In a shared-queues environment, do not use the local queue counts for queue evaluation purposes. The global queue counts, which are returned on this command, represent the current message queue counts on the shared queues structure.

When the LOCAL parameter is also specified on the SHOW keyword, then all IMS systems that process the command, including the command master, display the local queue count. This is valid whether shared message queues are enabled.

When the GLOBAL parameter is also specified on the SHOW keyword, and shared message queues are enabled, then the command master displays the global queue count on the shared message queues (MSGQ). If the user resource exists in the resource structure, then the lterms associated with the global user resource are used to obtain the queue counts from shared queues. Otherwise, the lterms associated with the local user resource are used, but if the user does not exist locally on the command master, then the queue count is 0.

The local and global queue counts are displayed as separate output fields.

RECOVERY

Displays the status recovery mode (SRM) and level of recovery for the user. End-user significant status can be conversation, Fast Path, full-function response mode, or STSN (set-and-test-sequence-number) status.

SRM determines where end-user significant status, if any exists, is recovered following a user signoff, or session or IMS termination. The output displays the SRM for the user as either GLOBAL (sysplex terminal management recovers it in the resource structure), LOCAL (IMS recovers it locally, which indicates an affinity to a particular IMS), or NONE (status is discarded).

Level of recovery determines what end-user significant status, if any exists, is recovered (if SRM is GLOBAL or LOCAL). The output displays whether conversation status is recovered (RCVYCONV), Fast Path status is recovered (RCVYFP), full-function response mode is recovered (RCVYRESP), and STSN status is recovered (RCVYSTSN).

STATUS

Returns local or global status of the user. See "QUERY USER status" on page 707 for a list and meaning of possible status that might be returned.

USERID

The RACF user ID that was used to sign the user on to a node.

VERSION

Displays the RM version number of the user resource. This is the version number assigned to the user, which is assigned by MVS, and maintained by RM, when the resource is created or updated in the resource structure. VERSION applies only when sysplex terminal management is enabled. VERSION is ignored when sysplex terminal management is not enabled.

STATUS()

Selects users for display that possess at least one of the specified user statuses. The status might exist locally or globally if sysplex terminal management (STM) is enabled.

The STATUS filter is valid in both a sysplex terminal management environment and in a non-sysplex terminal management environment. In a sysplex terminal management environment, the status selected might exist locally, globally, or both. If sysplex terminal management is not enabled, then the status only exists locally.

If SHOW(LOCAL) is specified, then IMS will select only those users with the appropriate status in the local system. The command is processed by all IMS systems, including the command master.

If SHOW(GLOBAL) is specified, and sysplex terminal management is enabled, then IMS will select only those users with the appropriate status in the resource structure. The command is processed only by the command master. If SHOW(GLOBAL) is specified, but sysplex terminal management is not enabled, then the command is rejected.

If SHOW(GLOBAL,LOCAL) is specified (the default), then IMS will select those users with the appropriate status either locally or in the resource structure (if sysplex terminal management is enabled). The command is processed by all IMS systems. The command master processes both global and local information.

The output returned when the status filter is specified includes the status of the user, even if SHOW(STATUS) is not specified.

To determine which filters can be used to select users with corresponding status, see "QUERY USER status" on page 707.

Usage notes

The QUERY USER command can be specified only through the OM API.

The QUERY USER command can be issued on an XRF alternate system, but SHOW(GLOBAL) is not supported. Only local information can be displayed.

The processing of the QUERY USER command is different depending on whether IMS sysplex terminal management is enabled.
- If IMS sysplex terminal management is not enabled, processing is local for each system. The results of type-1 and type-2 commands are similar.
- If IMS sysplex terminal management is enabled, type-1 and type-2 command processing is similar when displaying local information. However, they differ in how global information is displayed.
- For type-1 /DISPLAY commands with IMS sysplex terminal management enabled, the command master displays information from either the resource

structure or the local system, but not both. If the resource being displayed is not owned by any system or is owned by the command master, the command master displays the global resource. However, if the resource is owned by a system other than the command master, the command master displays only the local resource, and the owning system is responsible for displaying the global resource.

- For type-2 QUERY commands with IMS sysplex terminal management enabled, the command master is the only system that displays global resource information, regardless of whether the resource is owned. In addition, the command master displays local resource information. All other IMS systems that process the command display local resource information only. This approach allows more flexibility in displaying all information in an IMSplex.

The SHOW keyword determines which IMS systems process the command, and what information is displayed.

- If SHOW(GLOBAL) is specified, then the command master displays global information, which includes the global queue count if shared queues are enabled, and status from the resource structure if sysplex terminal management is enabled (STM=YES defined in DFSDCxxx PROCLIB member). This is true whether or not the user is active on any particular IMS system. All other IMS systems to which OM routes the command ignore the GLOBAL parameter with return code X'00000004' and reason code X'00001000'.
- If SHOW(LOCAL) is specified, then each IMS system to which OM routes the command (including the command master) processes the command, and displays information that is local to each system.
- If both GLOBAL and LOCAL are specified (which is the default), then the command master displays both global and local information, and all other IMS systems to which OM routes the command displays local information.

Similar IMS type-1 commands

The following table shows variations of the QUERY USER command and the type-1 IMS commands that perform similar functions.

Table 214. Type-1 equivalents for the QUERY USER command

QUERY USER command	Similar IMS type-1 command
QUERY USER SHOW(AUTOLOGON)	/DISPLAY USER user AUTOLOGON
QUERY USER SHOW(CONV)	/DISPLAY CONV USER user
QUERY USER SHOW(EMHQ)	/DISPLAY USER user QCNT EMHQ
QUERY USER SHOW(ID)	/DISPLAY ASMT USER user
QUERY USER SHOW(LTERM)	/DISPLAY ASMT USER user
QUERY USER SHOW(NODE)	/DISPLAY ASMT USER user /DISPLAY USER user
QUERY USER SHOW(OWNER)	/DISPLAY USER user RECOVERY
QUERY USER SHOW(PRESET)	/DISPLAY USER user
QUERY USER SHOW(QCNT)	/DISPLAY USER user /DISPLAY USER user QCNT
QUERY USER SHOW(RECOVERY)	/DISPLAY USER user RECOVERY
QUERY USER SHOW(STATUS)	/DISPLAY USER user
QUERY USER SHOW(USERID)	/DISPLAY ASMT USER user /DISPLAY USER user

Table 214. Type-1 equivalents for the QUERY USER command (continued)

QUERY USER command	Similar IMS type-1 command
QUERY USER STATUS(CONV)	/DISPLAY CONV
QUERY USER STATUS(status)	/DISPLAY STATUS USER

Output fields

The following table shows the QUERY USER output fields. The columns in the table are:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the column heading for the output field in the formatted output.

SHOW parameter
> Identifies the parameter on the SHOW keyword that caused the field to be generated. *Error* appears for output fields that are returned for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field. GBL indicates that the field can be generated only by the command master when displaying global information for SHOW(GLOBAL). LCL indicates that the field can be generated by any IMS displaying local information for SHOW(LOCAL). N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 215. Output fields for the QUERY USER command

Short label	Long label	SHOW parameter	Scope	Meaning
AUTID	AutLID	AUTOLOGON	GBL	Identifier of the other system half-session qualifier for autologon in the resource structure. This is only applicable if the user is associated with a parallel session ISC node.
AUTLD	AutLDsc	AUTOLOGON	GBL	The logon descriptor for autologon in the resource structure.
AUTMD	AutLMdt	AUTOLOGON	GBL	The VTAM mode table name for autologon in the resource structure.
AUTND	AutLNode	AUTOLOGON	GBL	The node name for autologon in the resource structure.
CC	CC	N/A	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 708 for more information. The completion code is always returned.
CCTXT	CCText	*Error*	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.

Table 215. Output fields for the QUERY USER command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
CONVID	ConvID	CONV	GBL	The conversation ID for a conversation associated with the user, as it exists in the resource structure. A user might have zero, one, or more conversations. Each conversation will have its own line of output.
CONVSTT	ConvStat	CONV	GBL	The status of a conversation associated with the user, as it exists in the resource structure. A user might have zero, one, or more conversations. The status can be: • CONVHELD: Conversation is held • CONVACTV: Conversation is active • CONVSCHD: Conversation is scheduled
CONVTRN	ConvTran	CONV	GBL	The transaction for a conversation associated with the user, as it exists in the resource structure. A user might have zero, one, or more conversations.
EMHQ	EMHQCnt	EMHQ	GBL	Global lterm queue count in the EMH (Expedited Message Handler) queues. EMHQ is displayed only if shared EMH is used.
GBL	Gbl	GLOBAL	GBL	If 'Y', then the output reflects the status found globally in RM. If blank, then the output reflects the status found locally.
ID	ID	ID	GBL	For ISC parallel-session terminals, displays the global half-session qualifier of the other system.
LAUTID	LAutLID	AUTOLOGON	LCL	Identifier of the other system half-session qualifier for autologon in the local system. This is applicable only if the user is associated with a parallel session ISC node.
LAUTLD	LAutLDsc	AUTOLOGON	LCL	The logon descriptor for autologon in the local system.
LAUTMD	LAutLMdt	AUTOLOGON	LCL	The VTAM mode table name for autologon in the local system.
LAUTND	LAutLNode	AUTOLOGON	LCL	The node name for autologon in the local system.
LCONVID	LConvID	CONV	LCL	The conversation ID for a conversation associated with the user, as it exists in the local system. A user might have zero, one, or more conversations. Each conversation will have its own line of output.
LCONVSTT	LConvStat	CONV	LCL	The status of a conversation associated with the user, as it exists in the local system. A user might have zero, one, or more conversations. The status can be: • CONVHELD: Conversation is held • CONVACTV: Conversation is active • CONVSCHD: Conversation is scheduled
LCONVTRN	LConvTran	CONV	LCL	The transaction for a conversation associated with the user, as it exists in the local system. A user might have zero, one, or more conversations.

Table 215. Output fields for the QUERY USER command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LID	LID	ID	LCL	For ISC parallel-session terminals, displays the local half-session qualifier of the other system.
LLTERM	LLterm	LTERM	LCL	Local logical terminal names. The logical terminal names associated with the user.
LNODE	LNode	NODE	LCL	Identifies the dynamic or static node associated with the user on the local system.
LPRST	LPreset	PRESET	LCL	Identifies the preset destination transaction or LTERM name when the user is in preset destination mode, which is established by the /SET command. All messages entered at this terminal are sent to the preset destination transaction code or LTERM.
LQ	LQCnt	QCNT	LCL	Local queue count.
LRCVY	LRcvy	RECOVERY	LCL	The level of recovery for end-user significant status in the local system, which indicates what type of status is recoverable. Any value presented here implies that the corresponding status is recoverable. If SRM is LOCAL, the status will be recovered locally. If SRM is GLOBAL, the status will be recovered globally. These values are not applicable if SRM is NONE or there is no SRM. The status values that can be returned (more than one are possible) are: • CONV: IMS conversations are recoverable (RCVYCONV=YES). • FP: Fast Path status is recoverable (RCVYFP=YES). • RESP: Full-function response mode status is recoverable (RCVYRESP=YES). • STSN: STSN status is recoverable (RCVYSTSN=YES).
LSRM	LSRM	RECOVERY	LCL	The status recovery mode in the local system, which determines where the end-user significant status is maintained and recovered from. The output will be one of the following: • GBL: Status is saved globally in the IMS resource structure. • LCL: Status is saved in local control blocks and log records. • NONE: Status is not saved in the IMS resource structure or log records. • Blank: SRM is not yet established, or the user is not signed on and there is no end-user significant status.
LSTT	LclStat	STATUS	LCL	Local user status. See "QUERY USER status" on page 707 for a list and explanation of the possible user status.
LTERM	Lterm	LTERM	GBL	Global logical terminal names. The logical terminal names associated with the user.
LUID	LUserid	USERID	LCL	Identifies the local user ID signed on to the user.

Table 215. Output fields for the QUERY USER command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LVER	LVersion#	VERSION	LCL	Version number for the user resource being maintained in the local system. This field applies only when STM is enabled.
MBR	MbrName	N/A	N/A	IMSplex member (modular unit) that built the output line. IMS identifier of the IMS that built the output. The IMS identifier is always returned.
NODE	Node	NODE	GBL	Identifies the dynamic or static node associated with the user in the resource structure.
OWNER	Owner	OWNER	GBL	Resource owner. IMS identifier or RSENAME of IMS where the user is active. If no owning IMS system exists and RM contains an entry for the resource, the owner field will be blank.
QCNT	QCnt	QCNT	GBL	Global queue count on the shared queues. Global queue count can only be displayed if shared queues are used.
RCVY	Rcvy	RECOVERY	GBL	The level of recovery for end-user significant status in the resource structure, which indicates what type of status is recoverable. Any value presented here implies that the corresponding status is recoverable. If SRM is LOCAL, the status will be recovered locally. If SRM is GLOBAL, the status will be recovered globally. These values are not applicable if SRM is NONE or there is no SRM. The status values that can be returned (more than one are possible) are: • CONV: IMS conversations are recoverable (RCVYCONV=YES). • FP: Fast Path status is recoverable (RCVYFP=YES). • STSN: STSN status is recoverable (RCVYSTSN=YES).
SRM	SRM	RECOVERY	GBL	The status recovery mode in the resource structure, which determines where the end-user significant status is maintained and recovered from. The output will be one of the following: • GBL: Status is saved globally in the IMS resource structure. • LCL: Status is saved in local control blocks and log records. • NONE: Status is not saved in the IMS resource structure or log records. • Blank: SRM is not yet established, or the user is not signed on and there is no end-user significant status.
STT	Status	STATUS	GBL	Global user status. See "QUERY USER status" on page 707 for a list and explanation of the possible node status.
UID	Userid	USERID	GBL	Identifies the RACF user ID signed on to the user.

Table 215. Output fields for the QUERY USER command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
USER	User	N/A	N/A	The user name. The user name is always returned.
VER	Version#	VERSION	GBL	Version number for the user resource being maintained in the resource structure. This field applies only when STM is enabled.

QUERY USER status

The following table shows the possible user status that can be displayed. The columns in the table are:

Status The user status that is displayed.

STATUS parameter
 The STATUS() filter that will select users with the specified status.

Scope The scope of the status. GBL indicates that the status can be global (it exists in the resource structure when STM is enabled), and is returned with the STT short label. LCL indicates that the status can be local, and is returned with the LSTT short label.

Meaning
 Provides a brief description of the status.

Table 216. QUERY USER status

Status	STATUS parameter	Scope	Meaning
ALLOC	ALLOC	GBL and LCL	A user or ISC subpool is allocated to a node.
CONVACT	CONV	GBL and LCL	An active conversation exists.
CONVHELD	CONV	GBL and LCL	All conversations are held.
DEADQ	DEADQ	LCL	The user has dead letter queues, or whose last access time was outside the limit set by the DLQT JCL parameter. The DEADQ status can be removed by signing on the user or entering the /DEQUEUE or /ASSIGN command.
EXCL	EXCL	GBL and LCL	The user is in exclusive mode set by the /EXCLUSIVE command. The exclusive mode restricts the output received by the terminal affected.
MFST	MFST	GBL and LCL	The user is in MFSTEST mode, set by the UPDATE USER START(MFST) command or the /TEST MFS command. Terminals supported by Message Format Service use format blocks from a special test library if the requested format block is in the test library; otherwise the blocks are obtained from the production library.

Table 216. QUERY USER status (continued)

Status	STATUS parameter	Scope	Meaning
PRST	PRST	LCL	The user is in preset destination mode. PRST mode is established by the /SET command. All messages entered at this terminal are sent to the preset destination transaction code or logical terminal.
RESP	RESP	LCL	The user is in response mode and the response reply message is available for output or in the process of being sent.
RESPINP	RESPINP	GBL and LCL	The user is in response mode and the response mode input is still in-doubt; for example, the response reply message is not available for output.
RESPINPFP	RESPINPFP	GBL and LCL	The user is in Fast-Path response mode and the response mode input is still in-doubt; for example, the response reply message is not available for output.
RM	RM	GBL	The user exists in the resource structure managed by RM.
RMACTIVE	RMACTIVE	GBL	The user is active (signed on) in the IMSplex, as indicated in the RM structure (RM active).
RMOWNED	RMOWNED	GBL	The user is owned by an IMS system in the IMSplex, as indicated in the RM structure (RM owned).
STATIC	STATIC	GBL and LCL	The user was defined during system definition.
STOSGN	STOSGN	GBL and LCL	The user was stopped from signing on with the UPDATE USER command or the /STOP USER command.
TEST	TEST	LCL	The user is in test mode set by the /TEST command. Test or echo mode sends any input from the terminal back to the terminal.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY USER command. The OM return and reason codes that may be returned as a result of the QUERY USER command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the QUERY USER command.

Table 217. Return and reason codes for the QUERY USER command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001000'	The command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000008'	X'00002014'	An invalid character was specified in the resource name.

Table 217. Return and reason codes for the QUERY USER command (continued)

Return code	Reason code	Meaning
X'00000008'	X'00002040'	An invalid parameter value was specified. An invalid SHOW or STATUS value might have been specified.
X'0000000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See Table 218 for details.
X'0000000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code. See Table 218 for details.
X'00000010'	X'00004004'	Command processing terminated because CQS was not active.
X'00000010'	X'00004005'	Command processing terminated because CQS was not connected to the queue structure.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	Command processing terminated because the resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'00004104'	Command processing terminated because RM is not available.
X'00000010'	X'00004108'	Command processing terminated because SCI is not available.
X'00000014'	X'00005004'	A DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	Command processing terminated because of an RM error.
X'00000014'	X'00005104'	Command processing terminated because of a CQS error.
X'00000014'	X'00005108'	Command processing terminated because of an SCI error.
X'00000014'	X'00005FFF'	Command processing terminated because of an internal IMS error.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 218. Completion codes for the QUERY USER command

Completion code	Completion code text	Meaning
0		The QUERY USER command completed successfully for the resource.

Table 218. Completion codes for the QUERY USER command (continued)

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
98	CQS REQUEST ERROR	Global queue counts could not be obtained because of a CQS error.
1A2	User Resource is in error	The user resource was found in the resource structure, and an associated resource was needed, but it was either not found or appeared to be in error. This is normally an error condition. However, it could be a temporary condition caused by terminal or command activity. The command should be retried.

Examples

The following are examples of the QUERY USER command:

Example 1 for QUERY USER command

TSO SPOC input:
```
QRY USER NAME(USER2*,XYZ) SHOW(LOCAL)
```

TSO SPOC output:
```
User     MbrName CC CCText
USER23   IMS1     0
USER23   IMS2     0
USER24A  IMS2     0
USER24B  IMS2     0
USER24C  IMS2     0
XYZ      IMS1    10 NO RESOURCES FOUND
XYZ      IMS2    10 NO RESOURCES FOUND
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. STM and shared queues are irrelevant because only LOCAL information is requested. IMS1, the command master, displays only local information because no global information is requested. IMS2 displays local information only.
- USER23 exists on IMS1 and IMS2.
- USER24A exists on IMS2.
- USER24B exists on IMS2.
- USER24C exists on IMS2.
- XYZ does not exist on any system.

Example 2 for QUERY USER command

TSO SPOC input:
```
QRY USER NAME(USER2*)
```

TSO SPOC output:

```
User      MbrName CC  Gbl
USER23    IMS1     0  Y
USER23    IMS1     0
USER23    IMS2     0
USER24A   IMS1     0  Y
USER24A   IMS2     0
USER24B   IMS1     0  Y
USER24B   IMS2     0
USER24C   IMS2     0
USER25    IMS1     0  Y
USER26    IMS1     0  Y
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information only.
- USER23 exists on IMS1, IMS2 and in the resource structure.
- USER24A exists on IMS2 and in the resource structure.
- USER24B exists on IMS2 and in the resource structure.
- USER24C exists on IMS2 only.
- USER25 exists in the resource structure only.
- USER26 exists in the resource structure only.

Example 3 for QUERY USER command

TSO SPOC input:
```
QRY USER NAME(USER2*) STATUS(STATIC) SHOW(LOCAL)
```

TSO SPOC output:

```
User      MbrName CC CCText            LclStat
USER24A   IMS2     0                   ALLOC,STATIC
USER24B   IMS2     0                   ALLOC,STATIC
USER24C   IMS2     0                   STATIC
USER2*    IMS1    10 NO RESOURCES FOUND
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM and shared queues are irrelevant because no global information is requested. IMS1, the command master, displays local information only because SHOW(LOCAL) is specified. IMS2 displays local information only. All static users are displayed, and status is displayed because the STATUS filter was specified. IMS1 did not find any static users that matched the name specified.
- USER24A exists on IMS2, and is allocated to a node.
- USER24B exists on IMS2, and is allocated to a node.
- USER24C exists on IMS2, but is not allocated.

Example 4 for QUERY USER command

TSO SPOC input:
```
QRY USER NAME(USER25) SHOW(GLOBAL,CONV,LTERM,STATUS)
```

TSO SPOC output:

(screen 1)
```
User      MbrName CC  Gbl  Lterm     ConvID ConvTran ConvStat
USER25    IMS1     0  Y
USER25    IMS1     0  Y    LTERM25A
USER25    IMS1     0  Y    LTERM25B
USER25    IMS1     0  Y                   1 TRAN1A   CONVHELD
```

```
USER25   IMS1    0  Y                    2 TRAN1A   CONVHELD
USER25   IMS1    0  Y                    3 TRAN1A   CONVACTV

(scrolled right to screen 2)
User     MbrName Gbl  Status
USER25   IMS1    Y    CONVACT,RM
USER25   IMS1    Y
USER25   IMS1    Y
USER25   IMS1    Y
USER25   IMS1    Y
USER25   IMS1    Y
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global information only. IMS2 ignores the command (RC=4, RSN=x1000) because only global information is requested.

USER25 exists in the resource structure. IMS1 displays a global line that shows that the user is not active (not signed on) in the IMSplex, and has a conversation active. There are two lterms assigned to the user, and are displayed on separate output lines. There are three conversations associated with the user, and are displayed on separate output lines.

Example 5 for QUERY USER command

TSO SPOC input:
```
QRY USER NAME(USER23) SHOW(CONV,STATUS,OWNER,RECOVERY)
```

TSO SPOC output:
```
(screen 1)
User     MbrName CC Gbl  Owner  SRM  Rcvy
USER23   IMS1    0  Y    IMS2   LCL  CONV,FP
USER23   IMS1    0
USER23   IMS2    0
USER23   IMS2    0
USER23   IMS2    0

(scrolled right to screen 2)
User     MbrName Gbl  ConvID ConvTran ConvStat Status
USER23   IMS1    Y                             ALLOC,RM,RMACTIVE,RMOWNED
USER23   IMS1
USER23   IMS2
USER23   IMS2
USER23   IMS2

(scrolled right to screen 3)
User     MbrName Gbl  LSRM LRcvy    LConvID LConvTran LConvStat LclStat
USER23   IMS1    Y
USER23   IMS1         LCL  CONV,FP
USER23   IMS2         LCL  CONV,FP                                        ALLOC,CONVACT
USER23   IMS2                       1 TRAN1A   CONVHELD
USER23   IMS2                       2 TRAN1B   CONVACTV
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are irrelevant because queue counts are not requested. IMS1, the command master, displays global and local information. IMS2 displays local information.

USER23 exists on IMS1, IMS2 and in the resource structure. IMS1 displays a global line that shows the user is active and owned on IMS2, and its status recovery

mode is LOCAL, which means conversation information is not known globally. IMS1 also displays a local line showing that USER23 exists locally, but has no local status. IMS2 displays the local information for the active user, which includes one status line that shows that an active conversation exists, and an additional output line for each conversation active or held locally.

Example 6 for QUERY USER command

TSO SPOC input:
```
QRY USER NAME(USER23) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
User     MbrName CC Gbl QCnt EMHQCnt Owner  SRM  Rcvy     Userid  Node    Version#
USER23   IMS1     0 Y      0       0 IMS2   LCL  CONV,FP  UID23   NODE23        38
USER23   IMS1     0
USER23   IMS2     0

(scrolled right to screen 2)
User     MbrName Gbl  AutLNode AutLMdt  AutLDsc  AutLID Status
USER23   IMS1    Y    NODE23   SLU2MOD2 NODE23          ALLOC,RM,RMACTIVE,RMOWNED
USER23   IMS1
USER23   IMS2

(scrolled right to screen 3)
User     MbrName Gbl  LQCnt LSRM LRcvy     LUserid  LNode    LVersion#  LPreset
USER23   IMS1    Y
USER23   IMS1           0 LCL  CONV,FP                               0
USER23   IMS2           0 LCL  CONV,FP  UID23    NODE23         38

(scrolled right to screen 4)
User     MbrName Gbl  LAutLNode LAutLMdt LAutLDsc LAutLID LclStat
USER23   IMS1    Y
USER23   IMS1
USER23   IMS2         NODE23    SLU2MOD2 NODE23           ALLOC,CONVACT
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Shared queues are active. IMS1, the command master, displays global and local information. IMS2 displays local information.

USER23 exists in IMS1, IMS2, and in the resource structure. IMS1 displays a global line that shows global queue counts and global status from the resource structure. Global status indicates that the user is active on IMS2, and its status recovery mode is LOCAL, which means conversation information is not known globally. IMS1 also displays a local line showing that USER23 exists locally, but has no local status. IMS2 displays the local information for the active user, which shows that conversation status exists locally.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related tasks:

Setting DEADQ status time with the DLQT parameter (Communications and Connections)

Related reference:

/DISPLAY ASMT command (Commands)

/DISPLAY CONV command (Commands)

QUERY USEREXIT command

Use the type-2 QUERY USEREXIT command to display information about the user exits that are defined in the USER_EXITS section of the DFSDFxxx member of the IMS PROCLIB data set. Information about user exits that are not specified in the USER_EXITS section of the DFSDFxxx member is not displayed in the output of the QUERY USEREXIT command.

Subsections:

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the QUERY USEREXIT command and keywords can be issued.

Table 219. Valid environments for the QUERY USEREXIT command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
QUERY USEREXIT	X	X	X
SHOW	X	X	X
TYPE	X	X	X

Syntax
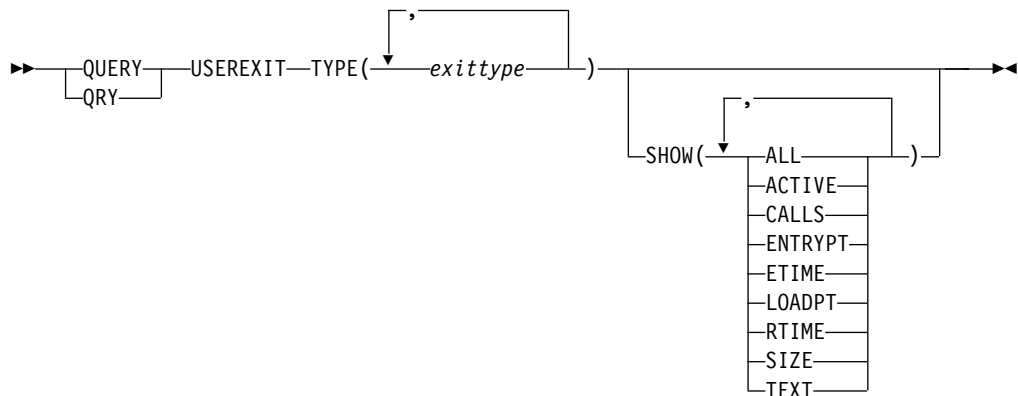
Keywords

The following keywords are valid for the QUERY USEREXIT command:

TYPE()
Specifies the user exit type or types for which you want information displayed. You can specify a single user exit type or a list of user exit types separated by commas. If the SHOW keyword is not specified, only the names of the exit routines for the specified types are returned. The valid user exit types are:

AOIE
Type-2 Automated Operator User Exit

BSEX
Build Security Environment user exit

ICQSEVNT
IMS CQS Event user exit

ICQSSTEV
IMS CQS Structure Event user exit

IMSMON
IMS Monitor user exit

INITTERM
Initialization/Termination user exit

LOGEDIT
Log Edit user exit

LOGWRT
Logger user exit

NDMX
Non-Discardable Messages user exit

OTMAIOED
OTMA Input/Output Edit user exit

OTMAYPRX
OTMA Destination Resolution user exit

OTMARTUX
OTMA Resume TPIPE Security user exit

PPUE
Partner Product user exit

RASE
Resource Access Security user exit

RESTART
Restart user exit

SHOW()
Specifies the information about the user exit routines to be returned in the output fields of the command response. The exit type and module name fields are always returned along with the name of the IMS that created the output for the user exit type and the completion code. The valid fields that can be specified are the following:

ALL
All possible output fields are returned.

ACTIVE

The number of currently active instances of the user exit routine. This is a point-in-time number that represents the number of calls to the user exit that are still in progress and have not returned to IMS.

CALLS

The number of calls to the user exit since the last user exit routine refresh. For performance reasons serialization is not obtained when IMS collects this number. For an exit type that can run multiple instances in parallel, this number is an approximation. The maximum value that this field can contain is 4,294,967,295 (2^{32}-1). When the call count exceeds this value, the field rolls over and starts again from zero.

ENTRYPT

The entry point address of the user exit routine.

ETIME

The total (cumulative) elapsed time in milliseconds spent in the exit module since it was last refreshed. For performance reasons serialization is not obtained when IMS collects this number. For an exit type that can run multiple instances in parallel, this number is an approximation. The maximum value that can be displayed in this field is 2,147,483,647 (2^{31}-1). If the elapsed time exceeds this value, 2147483647 is displayed.

LOADPT

The address at which the user exit routine was loaded.

RTIME

The local date and time that the user exit routine was last refreshed (or initially loaded, if no refreshes have occurred). The format of the output field is:

`yyyy-mm-dd hh:mm:ss.th`

SIZE

The size in bytes of the user exit load routine. This value is displayed in hexadecimal.

TEXT

The 32 bytes starting from offset +04 from the entry point of the exit module, translated to EBCDIC with non-printable characters replaced by periods (.). This is a common location for module identification information. If your user exit routines contain printable identification data at this point in the module, you can use the TEXT option to display that information.

Usage notes

You can issue the QRY USEREXIT command only through the Operations Manager (OM) API.

The output contains an entry for each user exit module within each user exit type specified in the QUERY USEREXIT command. The output of this command is defined in XML and is available to automation programs that communicate with OM.

The QRY USEREXIT command is routed to all IMS systems in the IMSplex as its default routine.

Output fields

The following table shows the QUERY USEREXIT output fields. The columns in the table are:

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned.

Scope   Identifies the scope of the output field.

Meaning
> Provides a brief description of the output field.

Table 220. Output fields for the QUERY USEREXIT command

Short label	Long label	Keyword	Scope	Meaning
ACTIVE	Active	ACTIVE	LCL	Number of active instances of this exit module.
CALLS	Calls	CALLS	LCL	Number of calls to this user exit module since the last refresh.
CC	CC	N/A	N/A	Completion code for the line of output. The completion code is always returned.
CCTXT	CCText	N/A	N/A	Completion code text that briefly explains the meaning of the nonzero completion code.
ENTRYPT	EntryPt	ENTRYPT	LCL	The entry point of this user exit module.
ETIME	ElapseTime	ETIME	LCL	The total time spent in this user exit module since the last refresh.
LOADPT	LoadPt	LOADPT	LCL	The load point of this user exit module.
MBR	MbrName	N/A	N/A	IMSPLEX member that built the output line. Member name is always returned.
NAME	ModName	N/A	LCL	User exit module name. User exit module name is always returned.
RTIME	RefreshTime	RTIME	LCL	The time this user exit module was last refreshed.
SIZE	ModSize	SIZE	LCL	The size in hexadecimal of this user exit module.
TEXT	ModuleText	TEXT	LCL	32 bytes from this user exit module translated into EBCDIC.
TYPE	ExitType	TYPE	LCL	User exit type requested by the QUERY command. User exit type is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the QUERY USEREXIT command are standard for all commands entered through the OM API.

The following table contains the return, reason, and completion codes for the QUERY USEREXIT command. Included in the tables is a brief explanation of the codes.

Table 221. Return and reason codes for the QUERY USEREXIT command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUERY USEREXIT command completed successfully.
X'00000004'	X'00001010'	The QUERY USEREXIT command is not processed because no user exit routines were found that matched the TYPE parameter.
X'0000000C'	X'00003000'	The QUERY USEREXIT command was successful for at least one user exit type. The QUERY USEREXIT command was not successful for one or more user exit types. The completion code indicates the reason for the error with the user exit type. The completion codes that can be returned by the QUERY USEREXIT command are listed in Table 222.
X'0000000C'	X'00003004'	The QUERY USEREXIT command was not successful for any of the user exit types specified. The completion code indicates the reason for the error with the user exit type. The completion codes that can be returned by the QUERY USEREXIT command are listed in Table 222.
X'00000014'	X'00005004'	The QUERY USEREXIT command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The QUERY USEREXIT command processing terminated because of an internal error.

Errors that are unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains completion codes that can be returned on a QUERY USEREXIT command.

Table 222. Completion codes for the QUERY USEREXIT command

Completion code	Completion code text	Meaning
0		The QUERY USEREXIT command completed successfully for the user exit routine.

Table 222. Completion codes for the QUERY USEREXIT command (continued)

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	None of the user exit types specified are known to the IMS that processed the command. The user exit types might have been typed in error. Confirm that the user exit types are spelled correctly on the command.

Examples

The following are examples of the QUERY USEREXIT command:

Example 1 for QUERY USEREXIT command

TSO SPOC input:

```
QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME)
```

TSO SPOC output:

```
Response for: QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME)
ExitType ModName  MbrName   CC      Calls RefreshTime
INITTERM DFSITRX2 SYS3       0          1 2013-04-22 10:20:41.12
INITTERM DFSITRX0 SYS3       0          1 2013-04-22 10:20:41.12
INITTERM DFSITRX1 SYS3       0          1 2013-04-22 10:20:41.12
```

OM API input:

```
CMD(QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.4.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2013.112 17:23:12.064868</statime>
<stotime>2013.112 17:23:12.065191</stotime>
<staseq>CB4069B7E1D642A1</staseq>
<stoseq>CB4069B7E1EA75A1</stoseq>
<rqsttkn1>USRT003 10102312</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT003 </userid>
<verb>QRY </verb>
<kwd>USEREXIT        </kwd>
<input>QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TYPE" llbl="ExitType" scope="LCL" key="YES" len="8"
 dtype="CHAR" align="left" />
<hdr slbl="NAME" llbl="ModName" scope="LCL" key="YES" len="8"
 dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" key="NO" len="4"
 dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" key="NO" len="4" dtype="INT"
 align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
```

```
<hdr slbl="CALLS" llbl="Calls" scope="LCL" key="YES" len="10"
 dtype="INT" align="right" />
<hdr slbl="RTIME" llbl="RefreshTime" scope="LCL" key="no" len="22"
 dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>TYPE(INITTERM) NAME(DFSITRX2) MBR(SYS3) CC(   0) CALLS(
 1) RTIME(2013-04-22 10:20:41.12) </rsp>
<rsp>TYPE(INITTERM) NAME(DFSITRX0) MBR(SYS3) CC(   0) CALLS(
 1) RTIME(2013-04-22 10:20:41.12) </rsp>
<rsp>TYPE(INITTERM) NAME(DFSITRX1) MBR(SYS3) CC(   0) CALLS(
 1) RTIME(2013-04-22 10:20:41.12) </rsp>
</cmdrspdata>
</imsout>
```

Example 2 for QUERY USEREXIT command

TSO SPOC input:

```
QRY USEREXIT TYPE(AOIE) SHOW(CALLS,RTIME)
```

TSO SPOC output:

```
Response for: QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME)
ExitType ModName  MbrName   CC        Calls RefreshTime
AOIE     DFSAOE00 SYS3       0            1 2013-04-22 10:20:41.12
AOIE     USREXIT2 SYS3       0            1 2013-04-22 10:20:41.12
```

OM API input:

```
CMD(QRY USEREXIT TYPE(AOIE) SHOW(CALLS,RTIME))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.4.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2013.112 17:23:12.064868</statime>
<stotime>2013.112 17:23:12.065191</stotime>
<staseq>CB4069B7E1D642A1</staseq>
<stoseq>CB4069B7E1EA75A1</stoseq>
<rqsttkn1>USRT003 10102312</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT003 </userid>
<verb>QRY </verb>
<kwd>USEREXIT        </kwd>
<input>QRY USEREXIT TYPE(INITTERM) SHOW(CALLS,RTIME) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TYPE" llbl="ExitType" scope="LCL" key="YES" len="8"
 dtype="CHAR" align="left" />
<hdr slbl="NAME" llbl="ModName" scope="LCL" key="YES" len="8"
 dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" key="NO" len="4"
 dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" key="NO" len="4" dtype="INT"
 align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="CALLS" llbl="Calls" scope="LCL" key="YES" len="10"
 dtype="INT" align="right" />
<hdr slbl="RTIME" llbl="RefreshTime" scope="LCL" key="no" len="22"
```

```
  dtype="CHAR" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>TYPE(AOIE) NAME(DFSAOE00) MBR(SYS3) CC(   0) CALLS(
 1) RTIME(2013-04-22 10:20:41.12) </rsp>
<rsp>TYPE(AOIE) NAME(USREXIT2) MBR(SYS3) CC(   0) CALLS(
 1) RTIME(2013-04-22 10:20:41.12) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Information about Initialization/Termination exits is displayed. The information includes the number of calls to the user exit since it was loaded and the date and time that the user exit routine was loaded.

Related concepts:

⬀ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

⬀ Command keywords and their synonyms (Commands)

QUERY USERID command

Use the QUERY USERID command to display information about user IDs across the IMSplex. In this context, a user ID is used for user or terminal security. A user ID is not the same as a user, which is either a dynamic (ETO) user, or a static or dynamic (ETO) ISC subpool user. This command can be specified only through the OM API. It is not valid on an XRF alternate.

Subsections:
- "Environment"
- "Syntax" on page 722
- "Keywords" on page 722
- "Usage notes" on page 724
- "Similar IMS type-1 commands" on page 725
- "Output fields" on page 725
- "QUERY USERID status" on page 726
- "Return, reason, and completion codes" on page 727
- "Examples" on page 728

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 223. Valid environments for the QUERY USERID command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
QUERY USERID	X		X
NAME	X		X
SHOW	X		X

Syntax

```
>>--+-QUERY-+--USERID--+-NAME(*)-------------------------+------------------->
    '-QRY---'          |               .-,---------.     |
                       |               V           |     |
                       '-NAME(--+---useridname--+--+--)--'
                                '-useridname*---'
```

```
   .-SHOW(GLOBAL,LOCAL)--------------------.
>--+---------------------------------------+-------------------------------><
   |          .-GLOBAL,LOCAL,-.            |
   +-SHOW(----+---------------+--| A |--)--+
   |          .-,--------.                 |
   |          V          |                 |
   '-SHOW(----+-GLOBAL-+-+--+-----------+--)'
             '-LOCAL--'    '-,--| A |--'
```

A:

```
        .-,--------.
        V          |
|---+---+-ALL-----+-+-----------------------------------|
        +-NODE----+
        +-OWNER---+
        +-STATUS--+
        +-USER----+
        '-VERSION-'
```
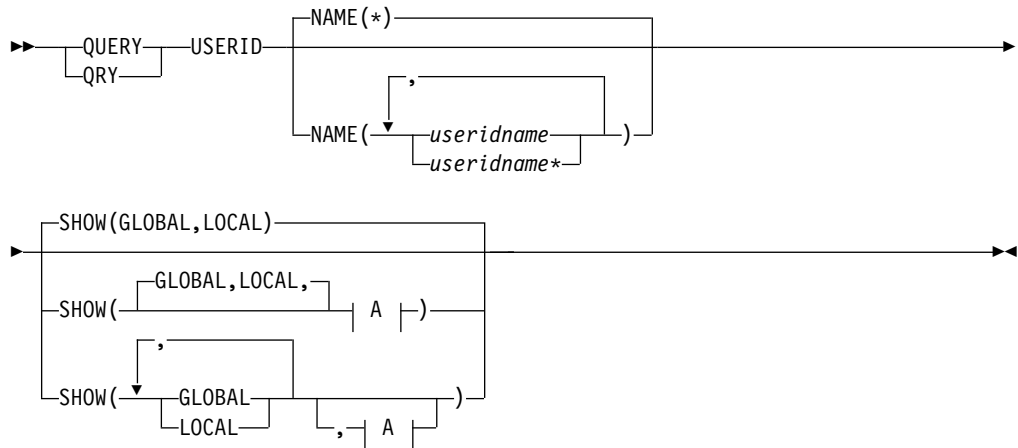
Keywords

The following keywords are valid for the QUERY USERID command:

NAME()
> Specifies the names of one or more user IDs that are to be displayed. Valid names are 1-8 characters, and wildcards can be specified. To display all user IDs, specify NAME(*). NAME(*) is the default.

SHOW()
> Specifies the user ID output fields to be returned. The user ID is always returned along with the name of the IMS that created the output and the completion code. If SHOW is not specified, only the user IDs are returned. This provides a method for a system management application to obtain a list of all user IDs that are currently known in the IMSplex.
>
> Two parameters, GLOBAL and LOCAL, are used to specify the location (global resources, or local resources) where IMS should obtain the information that is to be displayed. The default is both GLOBAL and LOCAL.
>
> The rest of the parameters are used to specify what information is displayed.
>
> The parameters supported with the SHOW keyword, which can be specified in any order, are:
>
> **ALL**
>> Returns all the output fields.
>
> **GLOBAL**
>> When GLOBAL is specified, the command master displays global

information, depending on the other SHOW parameters specified. This includes information from the resource structure.

The GLOBAL parameter is processed by the command master only. All other IMS systems ignore this parameter. If LOCAL is not also specified, then all IMS systems other than the command master ignore the command.

GLOBAL is applicable only if the command master is using sysplex terminal management.

GLOBAL is not applicable if the command master is not using sysplex terminal management. In this environment, if LOCAL is also specified, then GLOBAL is ignored. Otherwise, the command master rejects the command.

If sysplex terminal management is enabled, then the command master will make requests to RM to determine the appropriate global status.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line, and local information is displayed as another output line.

LOCAL
Specifies that all IMS systems, including the command master, display local information, depending on the other SHOW parameters specified. This includes information local to the IMS processing the command.

The LOCAL parameter is processed by all IMS systems, including the command master.

LOCAL is applicable in any environment, regardless of whether sysplex terminal management is used.

If both GLOBAL and LOCAL are specified (which is the default), then the command master builds global and local information separately. Global information is displayed as one output line, and local information is displayed as another output line.

NODE
Displays the node that the user ID is signed on to. This can be a static node or a dynamic node.

OWNER
Displays the owner of the user ID resource in the resource structure. This applies only when sysplex terminal management is enabled, and is processed by the command master only. All other IMS systems ignore this parameter.

The owner is the IMSID (or RSENAME for XRF systems) of the IMS system that owns the user ID. An IMS system owns a user ID resource if the resource is active (the user ID is signed on).

STATUS
Returns local or global status of the user ID. See "QUERY USERID status" on page 726 for a list and meaning of possible status that might be returned.

USER
Displays the dynamic or ISC user associated with the user ID.

VERSION
Displays the RM version number of the user ID resource. This is the

version number assigned to the user ID, which is assigned by MVS, and maintained by RM, when the resource is created or updated in the resource structure. VERSION applies only when sysplex terminal management is enabled. VERSION is ignored when sysplex terminal management is not enabled. Only the global version number is displayed. IMS does not maintain the version number in the local system.

Usage notes

The QUERY USERID command can be specified only through the OM API.

The QUERY USERID command is not supported on an XRF alternate system.

The processing of the QUERY USERID command is different depending on whether IMS sysplex terminal management is enabled.
- If IMS sysplex terminal management is not enabled, processing is local for each system. The results of type-1 and type-2 commands are similar.
- If IMS sysplex terminal management is enabled, type-1 and type-2 command processing is similar when displaying local information. However, they differ in how global information is displayed.
- For type-1 /DISPLAY commands with IMS sysplex terminal management enabled, the command master displays information from either the resource structure or the local system, but not both. If the resource being displayed is not owned by any system or is owned by the command master, the command master displays the global resource. However, if the resource is owned by a system other than the command master, the command master displays only the local resource, and the owning system is responsible for displaying the global resource.
- For type-2 QUERY commands with IMS sysplex terminal management enabled, the command master is the only system that displays global resource information, regardless of whether the resource is owned. In addition, the command master displays local resource information. All other IMS systems that process the command display local resource information only. This approach allows more flexibility in displaying all information in an IMSplex.

The SHOW keyword determines which IMS systems process the command, and what information is displayed.
- If SHOW(GLOBAL) is specified, then the command master displays global information, which includes status from the resource structure if sysplex terminal management is enabled (STM=YES defined in DFSDCxxx PROCLIB member). This is true whether or not the user ID is active on any particular IMS system. All other IMS systems to which OM routes the command ignore the GLOBAL parameter with return code X'00000004' and reason code X'00001000'. However, user ID resources are not maintained in the resource structure if multiple signons are allowed (the same user ID is allowed to sign on to multiple terminals simultaneously). The first IMS that joins the IMSplex determines whether multiple signons are allowed in the IMSplex by specifying SGN=G, M, or Z in the IMS startup parameters.
- If SHOW(LOCAL) is specified, then each IMS system to which OM routes the command (including the command master) processes the command, and displays information that is local to each system.
- If both GLOBAL and LOCAL are specified (which is the default), then the command master displays both global and local information, and all other IMS systems to which OM routes the command display local information.

Similar IMS type-1 commands

The following table shows variations of the QUERY USERID command and the type-1 IMS commands that perform similar functions.

Table 224. Type-1 equivalents for the QUERY USERID command

QUERY USERID command	Similar IMS type-1 command
QUERY USERID SHOW(NODE)	/DISPLAY ASMT USER user /DISPLAY USER user
QUERY USERID SHOW(STATUS)	/DISPLAY USER user
QUERY USERID SHOW(USER)	/DISPLAY ASMT USER user /DISPLAY USER user

Output fields

The following table shows the QUERY USERID output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

SHOW parameter
Identifies the parameter on the SHOW keyword that caused the field to be generated. *Error* appears for output fields that are returned for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field. GBL indicates that the field can be generated only by the command master when displaying global information for SHOW(GLOBAL). LCL indicates that the field can be generated by any IMS displaying local information for SHOW(LOCAL). N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 225. Output fields for the QUERY USERID command

Short label	Long label	SHOW parameter	Scope	Meaning
CC	CC	N/A	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" on page 727 for more information. The completion code is always returned.
CCTXT	CCText	*Error*	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
GBL	Gbl	GLOBAL	GBL	If 'Y', then the output reflects the status found globally in RM. If blank, then the output reflects the status found locally.

Table 225. Output fields for the QUERY USERID command (continued)

Short label	Long label	SHOW parameter	Scope	Meaning
LNODE	LNode	NODE	LCL	Identifies the dynamic or static node associated with the user ID on the local system.
LSTT	LclStat	STATUS	LCL	Local user ID status. See "QUERY USERID status" for information about the user ID status that might be returned.
LUSER	LUser	USER	LCL	Identifies the dynamic or ISC user associated with the lterm on the local system.
MBR	MbrName	N/A	N/A	IMSplex member that built output line. IMS identifier of the IMS that built the output. The IMS identifier is always returned.
NODE	Node	NODE	GBL	Identifies the dynamic or static node associated with the user ID in the resource structure.
OWNER	Owner	OWNER	GBL	Resource owner. IMS identifier or RSENAME of IMS where the user ID is active. If no owning IMS system exists and RM contains an entry for the resource, the owner field will be blank.
STT	Status	STATUS	GBL	Global user ID status from the resource structure. See "QUERY USERID status" for information about the user ID status that might be returned.
UID	UserID	N/A	N/A	User ID name. The user ID is always returned.
USER	User	USER	GBL	Identifies the dynamic or ISC user associated with the user ID in the resource structure.
VER	Version#	VERSION	GBL	Version number for the user ID resource being maintained in the resource structure. This field applies only when STM is enabled.

QUERY USERID status

The following table shows the possible user ID status that can be displayed. The columns in the table are:

Status The user ID status that is displayed.

Scope The scope of the status. GBL indicates that the status can be global (it exists in the resource structure when STM is enabled), and is returned with the STT short label. LCL indicates that the status can be local, and is returned with the LSTT short label.

Meaning
Provides a brief description of the status.

Table 226. QUERY USERID status

Status	Scope	Meaning
RM	GBL	The user ID exists in the resource structure managed by RM.
RMACTIVE	GBL	The user ID is active (signed-on) in the IMSplex, as indicated in the RM structure (RM active).

Table 226. QUERY USERID status (continued)

Status	Scope	Meaning
RMOWNED	GBL	The user ID is owned by an IMS system in the IMSplex, as indicated in the RM structure (RM owned).

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the QUERY USERID command. The OM return and reason codes that may be returned as a result of the QUERY USERID command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the QUERY USERID command.

Table 227. Return and reason codes for the QUERY USERID command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully.
X'00000004'	X'00001000'	The command was not processed on the IMS system as the IMS system is not the command master. No resource information is returned.
X'00000008'	X'00002014'	An invalid character was specified in the resource name.
X'00000008'	X'00002040'	An invalid parameter value was specified. An invalid SHOW value may have been specified.
X'0000000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See Table 228 on page 728 for details.
X'0000000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code. See Table 228 on page 728 for details.
X'00000010'	X'00004004'	Command processing terminated because CQS was not active.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	Command processing terminated because the resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'00004104'	Command processing terminated because RM is not available.
X'00000010'	X'00004108'	Command processing terminated because SCI is not available.
X'00000014'	X'00005004'	A DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005100'	Command processing terminated because of an RM error.

Table 227. Return and reason codes for the QUERY USERID command (continued)

Return code	Reason code	Meaning
X'00000014'	X'00005104'	Command processing terminated because of a CQS error.
X'00000014'	X'00005108'	Command processing terminated because of an SCI error.
X'00000014'	X'00005FFF'	Command processing terminated because of an internal IMS error.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 228. Completion codes for the QUERY USERID command

Completion code	Completion code text	Meaning
0		The QUERY USERID command completed successfully for the resource.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If this is a wildcard request there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
1A3	Userid resource is in error	The user ID resource was found in the resource structure, and an associated resource was needed, but it was either not found or appeared to be in error. This is normally an error condition. However, it could be a temporary condition caused by terminal or command activity. The command should be retried.

Examples

The following are examples of the QUERY USERID command:

Example 1 for QUERY USERID command

TSO SPOC input:
```
QRY USERID NAME(USERID*,XYZ) SHOW(ALL)
```

TSO SPOC output:
```
(screen 1)
UserID   MbrName CC CCText            Gbl Owner Node    User      Version#
USERID01 IMS1     0                   Y   IMS1  NODE01  USER01           1
USERID01 IMS1     0
USERID02 IMS1     0                   Y   IMS2  NODE02  USER02           1
USERID02 IMS2     0
USERID03 IMS1     0                   Y   IMS1  NODE03                   1
USERID03 IMS1     0
XYZ      IMS1    10 NO RESOURCES FOUND Y
```

```
XYZ      IMS1    10 NO RESOURCES FOUND
XYZ      IMS2    10 NO RESOURCES FOUND
```

```
(scrolled right to screen 2)
UserID   MbrName Gbl  Status                     LNode    LUser
USERID01 IMS1    Y    RM,RMACTIVE,RMOWNED
USERID01 IMS1                                    NODE01   USER01
USERID02 IMS1    Y    RM,RMACTIVE,RMOWNED
USERID02 IMS2                                    NODE02   USER02
USERID03 IMS1    Y    RM,RMACTIVE,RMOWNED
USERID03 IMS1                                    NODE03
XYZ      IMS1    Y
XYZ      IMS1
XYZ      IMS2
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Single-signon for user IDs is enforced. IMS1, the command master, displays global and local information. IMS2 displays local information only.
- USERID01 is signed on to dynamic or ISC node NODE01, user USER01 on IMS1.
- USERID02 is signed on to dynamic or ISC node NODE02, user USER02 on IMS2.
- USERID03 is signed on to static node NODE03 on IMS1.
- XYZ does not exist.

Example 2 for QUERY USERID command

TSO SPOC input:
```
QRY USERID NAME(USERID*) SHOW(ALL)
```

TSO SPOC output:
```
UserID   MbrName CC LNode      LUser
USERID11 IMS1     0 NODE11     USER11
USERID12 IMS1     0 NODE12A    USER12A
USERID12 IMS1     0 NODE12B    USER12B
USERID12 IMS2     0 NODE12C    USER12C
USERID12 IMS2     0 NODE12D
USERID13 IMS2     0 NODE13
```

Explanation: There are two IMS systems in the IMSplex: IMS1 and IMS2. RM is maintaining status (STM=YES). Mutliple-signons for user IDs are enabled, so user ID resources are not maintained in RM. IMS1 displays local information only because user IDs are not maintained in RM. IMS2 displays local information only.
- USERID11 is signed on to dynamic or ISC node NODE11, user USER11 on IMS1.
- USERID12 is signed on to multiple terminals simultaneously: dynamic or ISC nodes NODE12A and NODE12B on IMS1, dynamic or ISC node NODE12C on IMS2, and static node NODE12D on IMS2.
- USERID13 is signed on to static node NODE13.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➡ /DISPLAY ASMT command (Commands)

➡ /DISPLAY USER command (Commands)

➡ Command keywords and their synonyms (Commands)

Chapter 6. QUEUE commands

Use the IMS QUEUE commands to queue or dequeue LTERMs and transactions.

These commands can be issued to an IMSplex using the Batch SPOC utility.

Subsections:
- "QUEUE LTERM command"
- "QUEUE TRAN command" on page 736

QUEUE LTERM command

Use the QUEUE LTERM command to enqueue a message to the specified LTERM or to dequeue and discard messages currently enqueued to the LTERM.

When IMS uses local queue manager data sets, all IMS systems that receive this command process it.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 732
- "Usage notes" on page 733
- "Output fields" on page 733
- "Return, reason, and completion codes" on page 734
- "Examples" on page 735

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 229. Valid environments for QUEUE LTERM command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUEUE LTERM	X		X
NAME	X		X
DATA	X		X
OPTION	X		X

Syntax

```
►►──┬─QUEUE─┬──LTERM──NAME(ltermname)──────────────────────────────────►
    └─QUE───┘
```

```
            ┌─OPTION──(ENQ)──DATA──(──message data──)─┐
──►─────────┤                                         ├───────────────────────►◄
            └─OPTION──(──┬─DEQALL─┬──)────────────────┘
                         └─DEQ1───┘
```
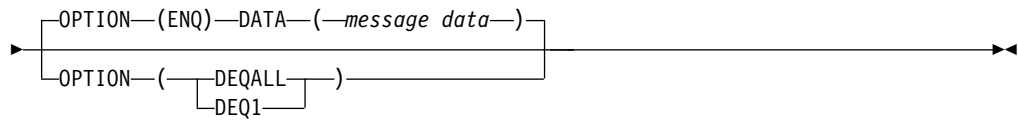
Keywords

The following keywords are valid for the QUEUE LTERM command:

NAME(*lterm*)
>   Specifies the 1- to 8-character name of the LTERM.

DATA(*message data*)
>   Specifies the message data to enqueue to the LTERM. The data is variable in
>   length. This parameter is valid only for OPTION(ENQ). If the COMM macro is
>   defined with OPTION=NOBLANK, DATA is an optional parameter. If the
>   COMM macro is defined with OPTION=BLKREQD (the default), at least one
>   character, even if it is a blank, must be specified in the message data area.
>
>   The message data specified can be in mixed case. If the LTERM is defined as
>   EDIT=ULC, the data is enqueued as specified on the command without any
>   uppercase translation. If the LTERM is defined as EDIT=UC, the data is
>   translated to uppercase before the message is enqueued. No EDIT routine is
>   called before the data is enqueued.
>
>   The data is enqueued as a single segment message. The maximum length of
>   data that can be enqueued is 32 763.

OPTION()
>   Specifies the option parameters.
>
>   **DEQALL**
>   >   Indicates that all of the messages currently enqueued to the LTERM should
>   >   be dequeued and discarded. The dequeue function is similar to a
>   >   /DEQUEUE LTERM command. The node and user must be stopped before
>   >   messages can be dequeued. To use the DEQALL option, the LTERM must
>   >   be defined locally on the IMS system, and the node (if the LTERM is
>   >   associated to a static node) or the user (if the LTERM is associated to
>   >   dynamic node) must be stopped.
>
>   **DEQ1**
>   >   Indicates the first (oldest) message on the queue for the LTERM should be
>   >   dequeued and discarded. The dequeue function is similar to a /DEQUEUE
>   >   LTERM command. The node and user must be stopped before a message
>   >   can be dequeued. To use the DEQ1 option, the LTERM must be defined
>   >   locally on the IMS system, and the node (if the LTERM is associated to a
>   >   static node) or the user (if the LTERM is associated to dynamic node) must
>   >   be stopped.
>
>   **ENQ**
>   >   Indicates that a message is to be enqueued to the specified LTERM. The
>   >   enqueue function is similar to a /BROADCAST command. If the LTERM is
>   >   stopped, no message can be enqueued to it.
>   >
>   >   The QUEUE LTERM OPTION(ENQ) command is not supported for remote
>   >   LTERMs.

Usage notes

If you use the ENQ option when IMS is using shared message queues, with or without STM=YES, only the command master IMS system processes the command. Messages are enqueued to the shared queues. The LTERM does not need to be defined in the IMS system that processes the command. If the LTERM does not exist, the command master IMS attempts to create the LTERM and associated user structure if ETO is active. If the LTERM is stopped, no message can be enqueued to it. If you use the ENQ option and the LTERM name specified is a remote LTERM, the message is placed on the shared queues using the MSNAME associated with the remote LTERM. The command is not processed if the MSNAME associated with the remote LTERM is stopped.

If you use the DEQALL or DEQ1 option when IMS is using shared message queues and STM=NO, only the command master processes the command. When IMS is using shared message queues and STM=YES, the command is processed by the command master IMS if the user or node is not owned, and by the owning system if the user or node is owned. Messages are dequeued from the shared queues. The node (if LTERM associated to a static node) or the user (if LTERM associated to dynamic node) must be stopped. The LTERM does not need to be defined in the IMS processing the command. If the LTERM does not exist, the IMS processing the command first attempts to create the LTERM and its associated user structure if ETO is active, and then processes the command.

The maximum length of the data that can be enqueued is 32 763 bytes. The data is enqueued to the LTERM as a single segment.

The QUEUE LTERM command includes completion code text with the non-zero completion code. The text can be up to 32 bytes and provides the meaning for the returned completion code:

Note: Before dequeuing one or all messages from shared message queues, stop the user (for ETO terminals) or NODE (or static terminals) for all IMS systems sharing the queues. The LTERM should not be in conversation mode.

Output fields

Output from QUEUE LTERM is returned encapsulated in `<cmdrsphdr>` and `<cmdrspdata>` XML tags. The short label is generated in the XML output. The SHOW keyword listed is the keyword on the command that causes the field to be generated. A value of "n/a" indicates that the output field is always returned. The following table shows the output fields for the QUEUE LTERM command.

Table 230. Output fields for the QUEUE LTERM command

Short label	SHOW keyword	Meaning
CC	n/a	The completion code for the output line. The completion code is always returned.
CCTXT	LCL	The completion code text that briefly explains the meaning of the completion code.
MBR	n/a	The IMSplex member that built the output line. The member name is always returned.
LTERM	n/a	The name of the LTERM resource. The LTERM name is always returned.

Table 230. Output fields for the QUEUE LTERM command (continued)

Short label	SHOW keyword	Meaning
LQCNT	DEQ1, DEQALL	The number of messages dequeued from a local queue.
QNCT	DEQ1, DEQALL	The number of messages dequeued from the shared message queue.

Return, reason, and completion codes

Commands that are issued through the OM API, including QUEUE LTERM, produce a standard set of OM return and reason codes, which are defined in the CSLORR request.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 231. QUEUE LTERM return and reason codes

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUEUE LTERM command completed successfully.
X'00000004'	X'00001000'	The IMS system is not the command master, so the QUEUE LTERM command is not processed on the IMS system.
X'00000008'	X'00002004'	An invalid keyword combination of DATA and OPTION(DEQ1|DEQALL) is specified, so the QUEUE LTERM command is not processed.
X'00000008'	X'00002008'	An insufficient number of keywords is specified, so the QUEUE LTERM command is not processed.
X'0000000C'	X'00003004'	The QUEUE LTERM command is not successful for the resource name specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the command are listed in the QUEUE LTERM completion code table.
X'00000010'	X'0000400C'	The QUEUE LTERM command is invalid on the XRF alternate.
X'00000010'	X'00004014'	The QUEUE LTERM command is invalid on the RSR tracker.
X'00000010'	X'0000401C'	The QUEUE LTERM command is invalid on the FDBR region.
X'00000010'	X'00004034'	The message queues are not available, so the QUEUE LTERM command is not processed.
X'00000014'	X'00005004'	DFSOCMD0 GETBUF storage could not be obtained, so the QUEUE LTERM command is not processed.
X'00000014'	X'00005008'	DFSPOOL HIOP storage could not be obtained, so the QUEUE LTERM command is not processed.

The following table contains completion codes that can be returned on a QUEUE
LTERM command.

Table 232. QUEUE LTERM completion codes

Completion code	Meaning
0	The QUEUE LTERM command completed successfully for the specified resource.
10	Because the resource name was not found, the QUEUE LTERM command is not processed.
12	Because the LTERM is active, the QUEUE LTERM OPTION(DEQ1|DEQALL) command is not processed.
1A	Because the LTERM is in conversation, the QUEUE LTERM OPTION(DEQ1|DEQALL) command is not processed.
5F	Because the LTERM name has invalid characters, the QUEUE LTERM command is not processed.
8A	An unsupported wildcard name was used, so the QUEUE LTERM command is not processed.
8B	A DFSQMGR request error occurred; the QUEUE LTERM command is not processed.
8C	The LTERM is stopped, so the QUEUE LTERM OPTION(ENQ) command is not processed.
8D	The LTERM is not stopped, so the QUEUE LTERM OPTION(DEQ1|DEQALL) command is not processed.
93	The QUEUE LTERM command is rejected, because the LTERM name specified is a reserved name.
9F	The QUEUE LTERM OPTION(ENQ) command is not supported for remote LTERMs.

Examples

The following are examples of the QUEUE LTERM command:

Example 1 for QUEUE LTERM command

TSO Input:
```
QUE LTERM NAME(LTERM1) OPTION(ENQ) DATA(MESSAGE1)
```

TSO Output:
```
Lterm    MbrName    CC
LTERM1   SYS3        0
```

Explanation: The QUEUE LTERM command completed successfully and was
processed by the command master IMS.

Example 2 for QUEUE LTERM command

TSO Input:
```
QUEUE LTERM NAME(IMSUS01) OPTION(DEQALL)
```

TSO Output:
```
Lterm    MbrName    CC    Qcnt
IMSUS01  IMS2        0       4
```

Explanation: Dequeues all messages on the LTERM queue. The QUEUE LTERM command completed successfully and processed by the command master IMS.

Example 3 for QUEUE LTERM command

TSO Input:
```
QUEUE LTERM NAME(IMSUS01) OPTION(DEQALL)
```

TSO Output:
```
Lterm     MbrName     CC     LQcnt
IMSUS01   IMS2        0        3
IMSUS01   IMS3        0        0
```

Explanation: Dequeues all messages on the LTERM queue. The QUEUE LTERM command completed successfully and is processed by all IMS systems that receive the command.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

QUEUE TRAN command

Use the QUEUE TRAN command to enqueue a message to the specified transaction or to dequeue and discard messages currently enqueued to the transaction.

The QUEUE TRAN command is defined to OM as ROUTE=ALL. When this command is issued, OM routes the command to all IMS systems that have registered for the command. Depending on the environment, either only the command master IMS processes the command or all IMS process the command. If only the command master processes the command, all other IMS systems that receive the command return with a return and reason code indicating that they are not the IMS command master.

Subsections:
- "Environment"
- "Syntax" on page 737
- "Keywords" on page 737
- "Usage notes" on page 738
- "Output fields" on page 739
- "Return, reason, and completion codes" on page 739
- "Examples" on page 741

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 233. Valid environments for QUEUE TRAN command and keywords

Command / Keyword	DB/DC	DBCTL	DCCTL
QUEUE TRAN	X		X
NAME	X		X
DATA	X		X
OPTION	X		X

Syntax

```
                                        ┌─OPTION(ENQ)─DATA(messagedata)─┐
►►──┬─QUEUE─┬──TRAN─NAME(tranname)──────┤                               ├──►◄
    └─QUE───┘                           └─OPTION(─┬─DEQALL─┬─)──────────┘
                                                  └─DEQ1───┘
```
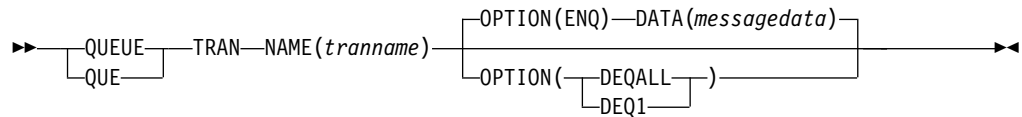
Keywords

The following keywords are valid for the QUEUE TRAN command:

NAME(`tranname`**)**
  Specifies the 1- to 8-character name of the transaction.

DATA(`messagedata`**)**
  Specifies the message data to enqueue to the transaction. The data is variable in length. This parameter is only valid for OPTION(ENQ). If the COMM macro is defined with OPTION=NOBLANK, DATA is an optional parameter. If the COMM macro is defined with OPTION=BLKREQD (the default), at least one character, even if it is a blank, must be specified in the message data area.

  The data can be specified in mixed case. If the transaction is defined as EDIT=ULC, the data is enqueued as it was specified on the command, without any uppercase translation. If the transaction is defined as EDIT=UC, the data is translated to uppercase before the message is enqueued. No EDIT routine is called before the data is enqueued.

  The data is enqueued as a single segment message. The maximum length of data that can be enqueued is 32 763.

OPTION()
  Specifies the option parameters:

  ENQ
    This is the default. Indicates that a message is to be enqueued to the specified transaction. The ENQ function is processed only by the command master IMS in both the local queues and shared-queues environment. Any output generated by the transaction is sent as an unsolicited output message to the OM that originated the command. Any MFS formatting associated with the output message is not returned.

    The transaction specified for the ENQ cannot be full-function response mode or a Fast Path transaction. The initiated transaction can be conversational, but it cannot be in conversational mode with OM.

  DEQALL
    Indicates that all messages currently enqueued to the transaction should be dequeued and discarded. The transaction must be stopped before messages can be dequeued.

DEQALL is valid in both local queues environment and the shared-queues environment. If IMS is in the local queues environment, all IMS systems that receive the command process the QUEUE TRAN OPTION(DEQALL) command. If IMS is in the shared-queues environment, only the command master IMS processes the QUEUE TRAN OPTION(DEQALL) command.

A QUEUE TRAN OPTION(DEQALL) command can dequeue transaction messages queued to a transaction from APPC or OTMA Clients. The command only dequeues the transaction messages. The APPC or OTMA transaction instance blocks (TIB) created at the IMS system are not deleted and the storage may remain allocated until the next IMS restart. For APPC, the TIB is not released and the APPC conversation (the client) hangs if there is no timeout value specified. The APPCIOT=(x,Y) timeout value for APPC would deallocate the APPC conversation and release the TIB after the timeout limit has been reached.

DEQ1

Indicates that the first (oldest) message on the queue for the transaction should be dequeued and discarded. The transaction must be stopped before the message can be dequeued.

A QUEUE TRAN OPTION(DEQ1) command can dequeue transaction messages queued to a transaction from APPC or OTMA Clients. The command only dequeues the transaction messages. The APPC or OTMA transaction instance blocks (TIB) created at the IMS system are not deleted and the storage may remain allocated until the next IMS restart. For APPC the TIB is not released and the APPC conversation (the client) hangs if there is no timeout value specified. The APPCIOT=(x,Y) timeout value for APPC would deallocate the APPC conversation and release the TIB after the timeout limit has been reached.

Usage notes

This command is processed if the transaction is defined locally at the IMS system. If the transaction is not defined at the IMS system, the Destination Creation exit routine, DFSINSX0, is called. If the exit routine successfully creates a transaction, the command is processed. If the exit does not create a transaction, a response line is returned with an error completion code indicating that the transaction is not found.

The effect of this command differs depending on whether the IMS processing it uses RM's resource structure. If it does use the resource structure, a message can be enqueued even if the transaction is stopped locally if the transaction does not have a global status of STOQ. To dequeue one or all messages, however, the transaction must be stopped globally and locally for scheduling (STOSCHD). If the IMS does not use the resource structure, no messages can be enqueued if the transaction is stopped for queuing (STOQ). To dequeue one or all messages, the transaction must be stopped for scheduling (STOSCHD).

The QUEUE TRAN OPTION(ENQ) command is not supported for conversational, response mode, Fast Path, and remote transactions. This form of the command performs security checking on the transaction name and on the user ID that issued the command. For OPTION(ENQ), workload management (WLM) CLASSIFY calls are issued using information passed to the OM from the command. The information passed includes LUNAME=DFSOMAPI and the user ID. Also for OPTION(ENQ), if the transaction name specified is a remote transaction, the message is placed on the shared queues (on the transaction ready queue) using the remote transaction name.

The command is not processed if the MSNAME associated with the remote transaction is stopped. The TM and MSC Message Routing and Control user exit routine, DFSMSCE0, is not called for a QUEUE TRAN command.

The maximum length of the data that can be enqueued is 32 763 bytes. The data is enqueued to the transaction as a single segment.

If the transaction is processed by an IMS Version 10 or higher system, any generated output to the input destination is sent as an unsolicited output message to the OM that originated the command. The transaction name and IMS ID that processed the transaction are included in the output as follows:

```
8 byte transactionname + 8 byte IMSID + output message
```

If the transaction is processed by an IMS Version 9 or earlier, any generated output to the input destination is not queued. The application program receives a status code of AD.

Output fields

Output from the QUEUE TRAN command is returned encapsulated in <cmdrsphdr> and <cmdrspdata> XML tags. The short label is generated in the XML output. The keyword listed is the keyword on the command that causes the field to be generated. A value of "n/a" indicates that the output field is always returned. A value of *error* indicates that the output field is returned if there is an error (for example, a nonzero completion code).

Table 234. QUEUE TRAN output

Short label	SHOW keyword	Meaning
CC	n/a	The completion code for the output line. The completion code is always returned.
CCTXT	n/a	The completion code text that explains the completion code. The member name is always returned.
MBR	*error*	The IMSPlex member that built the output line.
TRAN	n/a	The name of the transaction resource.
LQCNT	OPTION(DEQ1 | DEQALL)	The number of messages dequeued from a local queue.
QNCT	OPTION(DEQ1 | DEQALL)	The number of messages dequeued from the shared message queue.

Return, reason, and completion codes

Commands that are issued through the OM API, including QUEUE TRAN, produce a standard set of OM return and reason codes, which are defined in CSLORR.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 235. QUEUE TRAN return and reason codes

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The QUEUE TRAN command completed successfully.
X'00000004'	X'00001000'	The IMS system is not the command master, so the QUEUE TRAN command is not processed on the IMS system.
X'00000008'	X'00002004'	An invalid keyword combination of DATA and OPTION(DEQ1|DEQALL) is specified, so the QUEUE TRAN command is not processed.
X'00000008'	X'00002008'	An insufficient number of keywords is specified, so the QUEUE TRAN command is not processed.
X'0000000C'	X'00003004'	The QUEUE TRAN command is not successful for the resource name specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the command are listed in the QUEUE TRAN completion code table.
X'00000010'	X'0000400C'	The QUEUE TRAN command is invalid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000401C'	The QUEUE TRAN command is invalid on the FDBR region.
X'00000010'	X'00004034'	The message queues are not available, so the QUEUE TRAN command is not processed.
X'00000014'	X'00005004'	DFSOCMD0 GETBUF storage could not be obtained, so the QUEUE TRAN command is not processed.
X'00000014'	X'00005008'	DFSPOOL HIOP storage could not be obtained, so the QUEUE TRAN command is not processed.
X'00000014'	X'00005FFF'	The QUEUE TRAN command could not be processed because of an internal error.

The following table contains completion codes that can be returned on a QUEUE TRAN command.

Table 236. QUEUE TRAN completion codes

Completion code	Meaning
0	The QUEUE TRAN command completed successfully for the specified resource.
10	The resource was not found.
17	Another command is in progress for this transaction.
1A	Transaction has an active conversation. Use the /EXIT command to terminate the conversation and to dequeue messages.
51	No resource structure
52	Resource structure full

Table 236. QUEUE TRAN completion codes (continued)

Completion code	Meaning
5F	The QUEUE TRAN command is rejected, because the transaction name specified has invalid characters.
8A	An unsupported wildcard name was used, so the QUEUE TRAN command is not processed.
8B	A DFSQMGR request error occurred, so the QUEUE TRAN is not processed.
8C	The transaction is stopped, so the QUEUE TRAN OPTION(ENQ) is not processed.
8D	The transaction is not stopped, so the QUEUE TRAN OPTION(DEQ1 | DEQALL) is not processed.
94	RM Request error
96	A RACF security failure has occurred, so the QUEUE TRAN OPTION(ENQ) is not processed.
98	CQS request error
99	Transaction is not initialized
9A	QUEUE TRAN OPTION(ENQ) is not supported for conversational mode.
9B	QUEUE TRAN OPTION(ENQ) is not supported for Fast Path transactions.
9C	QUEUE TRAN OPTION(ENQ) is not supported for response mode transactions.
9D	The MSNAME associated to remote transaction is stopped.
9F	QUEUE TRAN OPTION(ENQ) is not supported for a remote transaction.

Examples

The following are examples of the QUEUE TRAN command:

Example 1 for QUEUE TRAN command

TSO Input:
```
QUEUE TRAN NAME(PART) OPTION(DEQALL)
```

TSO Output:
```
Trancode  MbrName  CC  LQcnt
----------------------------------
PART      IMS1     0   5
PART      IMS2     0   2
```

Explanation: The command was processed by all the IMS systems that received the command. IMS1 had five messages dequeued from its local queue, and IMS2 had two messages dequeued from its local queue.

Example 2 for QUEUE TRAN command

TSO SPOC input:
```
QUEUE TRAN NAME(PART) DATA(message1)
```

TSO SPOC output:
```
Trancode MbrName    CC
PART     IMS2        0
```

Explanation: The QUEUE TRAN command completed successfully and was processed by the command master.

Example 3 for QUEUE TRAN command

TSO SPOC input:
```
QUEUE TRAN NAME(PART) OPTION(DEQALL)
```

TSO SPOC output:
```
Trancode MbrName    CC    Qcnt
PART     IMS3        0       2
```

Explanation: This example shows the QUEUE TRAN OPTION(DEQALL) in a shared-queues environment. Two messages are dequeued by the command master IMS.

Example 4 for QUEUE TRAN command

TSO SPOC input:
```
QUEUE TRAN NAME(PART) OPTION(DEQALL)
```

TSO SPOC output:
```
Trancode MbrName    CC    LQcnt
PART     IMS2        0       2
PART     IMS3        0       0
PART     SYS3        0       1
```

Explanation: This example shows the QUEUE TRAN OPTION(DEQALL) in a Local Queues environment. Each IMS processes the command.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

Chapter 7. /QUIESCE command

The /QUIESCE command initiates the shutdown and deallocates the user for the specified ISC node.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 744
- "Example" on page 744

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 237. Valid environments for the /QUIESCE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/QUIESCE	X		X
NODE	X		X
USER	X		X

Syntax
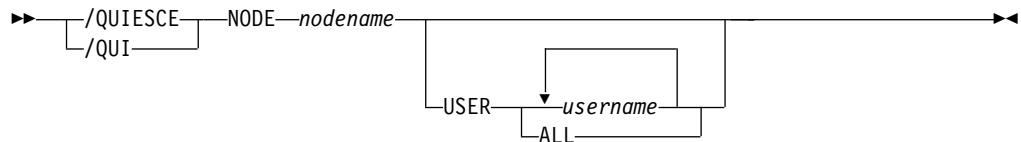
Keywords

The following keywords are valid for the /QUIESCE command:

NODE
Specifies the VTAM node for the user to be shut down and deallocated.

USER
If the USER keyword is omitted, all half-sessions of an ISC node are affected. The half-sessions must be connected.

Restrictions for using NODE and USER parameters together:
- Commands with the NODE USER keyword pair are valid only if:
    - The USER is signed on to the NODE
    - In an ISC environment, the USER is allocated to the NODE
    - The nodes and users already exist
- /QUIESCE NODE USER commands are valid for ISC nodes only.

Usage notes

This command can be issued to an IMSplex using the Batch SPOC utility.

The `/QUIESCE NODE` command is valid for ISC nodes only.

/QUIESCE resets preset mode, test mode, lock node, lock lterm, pstop lterm, and purge lterm because these statuses are not significant and are not kept after a logon or restart. /QUIESCE also takes other actions depending on the recovery settings for the node:

RCVYCONV=NO
> /QUIESCE causes any IMS conversations (active and held) to be terminated. Any conversational message that is queued or being processed has its output response message delivered asynchronously.

RCVYFP=NO
> /QUIESCE causes Fast Path status and messages to be discarded

RCVYRESP=NO
> /QUIESCE resets full-function response mode.

If global resource information is not kept in Resource Manager, /QUIESCE deallocates the user and resets status locally. If global resource information is kept in Resource Manager, /QUIESCE deallocates the user and resets status globally. If the user has no significant status, /QUIESCE deletes the user in Resource Manager. If the node has no significant status, and there are no other half-sessions for the node, /QUIESCE deletes the node in Resource Manager.

If ROUTE is specified, it should be specified with ROUTE(*). The command fails if not routed to the IMS where the node is active.

Example

Entry ET:
```
  /QUIESCE NODE CAL USER LAX
```

Response ET:
```
  /DFS058I  QUIESCE COMMAND COMPLETED
```

Explanation: The half-session of node CAL using user LAX is shut down.

Related reference:

Chapter 8. /RCLSDST command

The /RCLSDST (remote close destination) command causes IMS to disconnect the VTAM terminal from which the command is entered.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Example" on page 746

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command can be issued.

Table 238. Valid environments for the /RCLSDST command

Command	DB/DC	DBCTL	DCCTL
/RCLSDST	X		X

Syntax

```
►►──┬─/RCLSDST─┬──────────────────────────────────────────────────────►◄
    └─/RCL─────┘
```
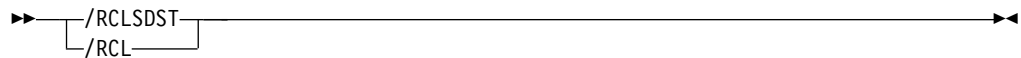
Usage notes

If you are in an active conversational mode, /EXIT or /HOLD must be entered before /RCLSDST is executed. If this command is issued by a signed on user, the user is signed off.

This command does not reset preset mode.

/RCLSDST resets preset mode, test mode, lock node, lock lterm, pstop lterm, and purge lterm because these statuses are not significant and, therefore, are not kept after logons and restart. /RCLSDST also takes other actions depending on the recovery settings for the node:

RCVYSTSN=NO

/RCLSDST acts like a /CHANGE NODE COLDSESS command for FINANCE and SLUP nodes by setting the session status to 'cold'. /RCLSDST acts like a /QUIESCE NODE command for ISC (LU6.1) nodes by initiating the shutdown and deallocating the user for the specified node. This action changes the session status to 'cold'. With these actions taken by the /RCLSDST command, the next session initiation request for this node is allowed to again attempt a session cold start. For ETO nodes, the control block structure could be deleted, if no significant status exists.

RCVYCONV=NO

/RCLSDST causes any held IMS conversations to be terminated. Any

conversational message that is queued or being processed has its output response message delivered asynchronously.

RCVYFP=NO
/RCLSDST causes Fast Path status and messages to be discarded.

RCVYRESP=NO
/RCLSDST resets full-function response mode.

If global resource information is not kept in Resource Manager, /RCLSDST logs a node off and resets status locally. If global resource information is kept in Resource Manager, /RCLSDST logs a node off and resets status globally. If the node has no status, /RCLSDST deletes the node in Resource Manager.

Example

Entry ET:
```
/RCLSDST
```

Response ET:
```
DFS058I  RCLSDST COMMAND COMPLETED
```

Explanation: The entering terminal is logged off IMS.

Chapter 9. /RCOMPT command

The /RCOMPT command sets a particular VTAM terminal component to a ready/not ready state.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes" on page 748
- "Example" on page 748

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 239. Valid environments for the /RCOMPT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RCOMPT	X		X
CNS	X		X
NOTRDY	X		X
PCH	X		X
PDS	X		X
PRT	X		X
RDR	X		X
READY	X		X
TDS	X		X
UDS	X		X
VID	X		X

Syntax

```
                    ┌─1──────────────────
                    │                    ┌─1────┐   ┌─READY──┐
►►──┬─/RCOMPT─┬──┬──┼─CNS─┬──────────┬───┴─compt#─┴───┴─NOTRDY─┴──────────────►◄
    └─/RCO────┘  │  ├─PCH─┤
                 │  ├─PDS─┤
                 │  ├─PRT─┤
                 │  ├─RDR─┤
                 │  ├─TDS─┤
                 │  ├─UDS─┤
                 │  └─VID─┘
                 ├─2──────────────────
                 ├─3──────────────────
                 └─4──────────────────
```
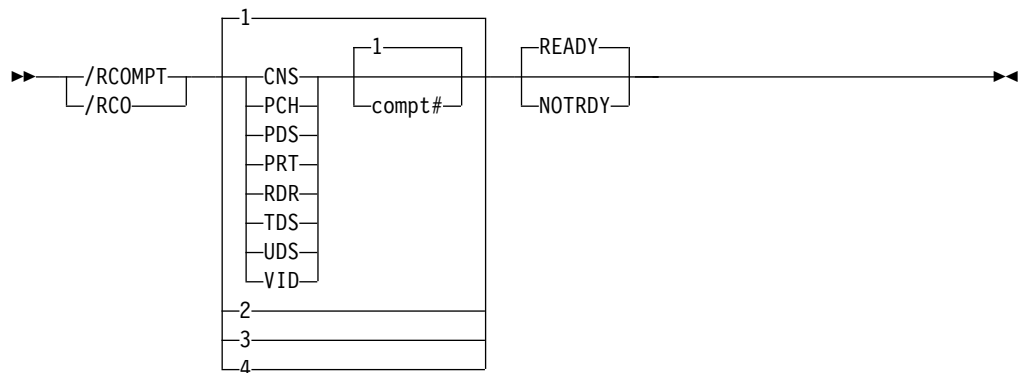
Usage notes

Output messages queued for a particular component will not be sent unless the component is ready. Depending on terminal type, output operations for other components can continue.

Note: Defaults are READY and 1.

The ready/not ready state set by the /RCOMPT command can be altered by the following:
- Another /RCOMPT command
- A /COMPT, /START, or /RSTART command
- An I/O error on the terminal component

The command format takes one of the following forms:
- A keyword is used.

  A search is made of the components (as defined in the TERMINAL macro during IMS system definition or logon descriptor) for the component defined that corresponds to the specified keyword. When a match is found, that component type is made ready or not ready as specified by the command.
- A keyword is used with a number other than 1 following the keyword.

  The corresponding occurrence of that component type is made ready or not ready, as specified by the command.
- Number 1 through 4 is used instead of a keyword.

  The component affected is the one defined in that position during system definition or logon descriptor independent of component type.

When using ISC, only parameters 1, 2, 3, and 4 are valid.

Example

Entry ET:
```
/RCOMPT VID 2 READY
```

Response ET:
```
DFS058I  RCOMPT COMMAND COMPLETED
```

Explanation: The second display component is declared operable to IMS.

Entry ET:
```
/RCOMPT 4 READY
```

Response ET:
```
DFS058I  RCOMPT COMMAND COMPLETED
```

Explanation: The fourth component defined is declared ready to IMS.

Related reference:

➡ /ASSIGN command (Commands)

Chapter 10. /RDISPLAY command

The /RDISPLAY command refers to the terminal assigned as the master terminal and displays either the logical terminal name and the line and physical terminal numbers, or the logical terminal name and the VTAM NODE name.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Equivalent IMS type-2 commands"
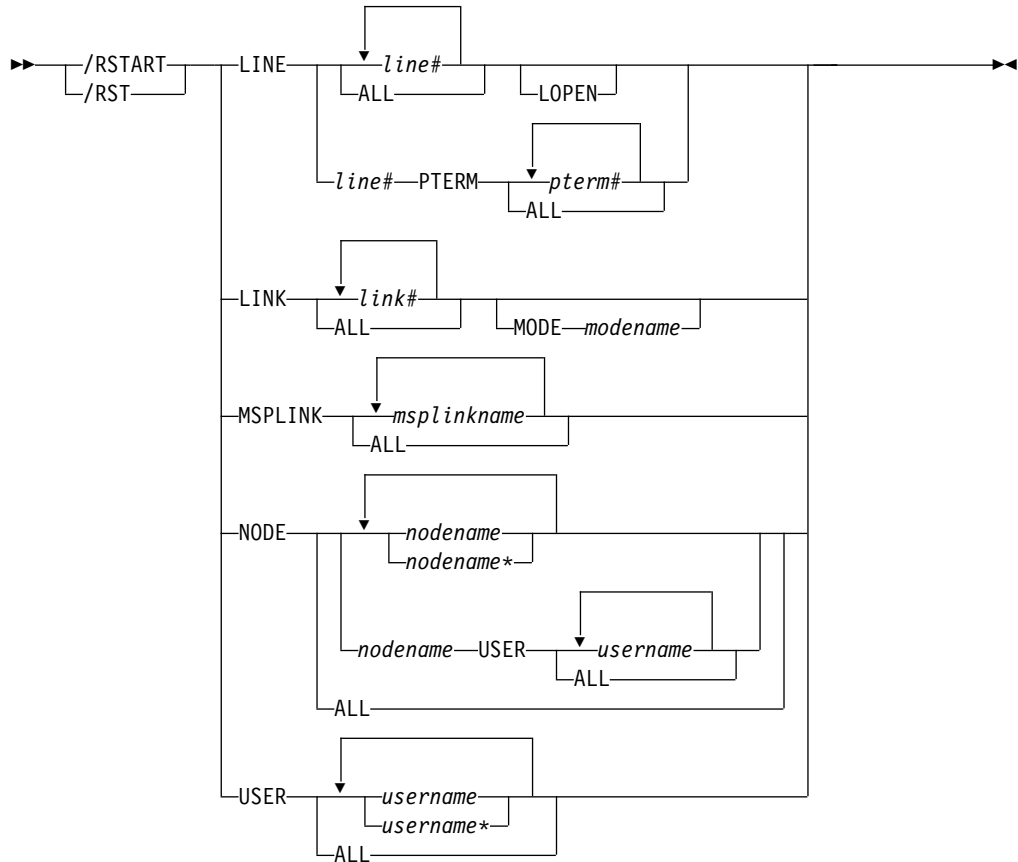- "Examples" on page 750

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command and keyword can be issued.

Table 240. Valid environments for the /RDISPLAY command and keyword

Command/Keyword	DB/DC	DBCTL	DCCTL
/RDISPLAY	X		X
MASTER	X		X

Syntax

```
>>──┬─/RDISPLAY─┬──MASTER──────────────────────────────────────────────────><
    └─/RDI──────┘
```

Keywords

The following keyword is valid for the /RDISPLAY command:

MASTER
   Specifies the identity of the terminal designated as the master terminal.

Usage notes

If the 3270 master terminal capability was generated during IMS system definition, the logical terminal name, line, and physical terminal number of the secondary master terminal are also displayed.

This command can be issued to an IMSplex using the Batch SPOC utility.

Equivalent IMS type-2 commands

The following table shows variations of the /RDISPLAY command and the IMS type-2 commands that perform similar functions.

Table 241. Type-2 equivalents for the /RDISPLAY command

Task	/RDISPLAY command	Similar IMS type-2 command
Displays the primary and secondary master terminal.	/RDISPLAY MASTER	QUERY LTERM STATUS(MTO,SMTO)

Examples

The following are examples of the /RDISPLAY command:

Example 1 for /RDISPLAY command

Entry ET:

```
/RDISPLAY MASTER
```

Response ET:

```
LTERM CNTRL
PTERM 3-1
*91010/123704*
```

Explanation: CNTRL is the master terminal logical terminal and is assigned to LINE 3 PTERM 1.

Example 2 for /RDISPLAY command

Entry ET:

```
/RDISPLAY MASTER
```

Response ET:

```
LTERM CTRL1
PTERM 4-2
LTERM CTRL2
PTERM 4-4
*91010/12370*
```

Explanation: CTRL1 is the primary master terminal logical terminal and is assigned to LINE 4 PTERM 2. CTRL2 is the secondary master terminal logical terminal and is assigned to LINE 4 PTERM 4.

Related reference:

"QUERY LTERM command" on page 299

Chapter 11. /RECOVER commands

The /RECOVER commands are used with the recovery list of database data sets and areas.

In an IMSplex, the /RECOVER command initiates the operation of the Database Recovery facility. OM sends the /RECOVER command to one IMS.

These commands can be issued to an IMSplex using the Batch SPOC utility.

/RECOVER commands are:
- "/RECOVER ADD command"
- "/RECOVER REMOVE command" on page 756
- "/RECOVER START command" on page 760
- "/RECOVER STOP command" on page 765
- "/RECOVER TERMINATE command" on page 768

/RECOVER ADD command

Use the /RECOVER ADD command to add database data sets and areas to a list (recovery list) of database data sets and areas to be recovered using the Database Recovery Facility.

The database data sets and areas can be specified as database data sets, areas, databases, or groups.

Subsections:
- "Environment"
- "Syntax" on page 752
- "Keywords" on page 752
- "Usage notes" on page 755
- "Examples" on page 755

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 242. Valid environments for the /RECOVER ADD command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
ADD	X	X	
AREA	X	X	
CAGROUP	X	X	
DB	X	X	
DBDS	X	X	
DBDSGRP	X	X	

Table 242. Valid environments for the /RECOVER ADD command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
OFFLINE	X	X	
RCVTOKEN	X	X	
RECOVGRP	X	X	
SMSOPTS	X	X	
STAGLOBAL	X	X	
STALOCAL	X	X	
USEAREA	X	X	
USEDBDS	X	X	

Syntax

```
►►──┬─/RECOVER─┬──ADD──┬───────────────────────────────────┬──┬─OFFLINE───┬──────►
    └─/REC─────┘       └─┬─RCVTOKEN─┬──recoveryname─────────┘  ├─STAGLOBAL─┤
                         └─RCK──────┘                          └─STALOCAL──┘


►──┬──────────┬──┬─SMSOPTS─┬──────────────┬──┬─AREA────┬──areaname──────┬──►◄
   ├─USEDBDS──┤  └─────────┬─optionname─┬─┘  │         └────────────────┘
   └─USEAREA──┘            └────────────┘    ├─CAGROUP──┬──groupname──────┤
                                             │          └─────────────────┘
                                             ├─DB───┬──dbname──────────────┤
                                             │      └──────────────────────┤
                                             ├─DBDS──┬──dbname──ddname──────┤
                                             │       └──────────────────────┤
                                             ├─DBDSGRP──┬──groupname─────────┤
                                             │          └────────────────────┤
                                             └─RECOVGRP──┬──groupname─────────┘
                                                         └─────────────────────┘
```
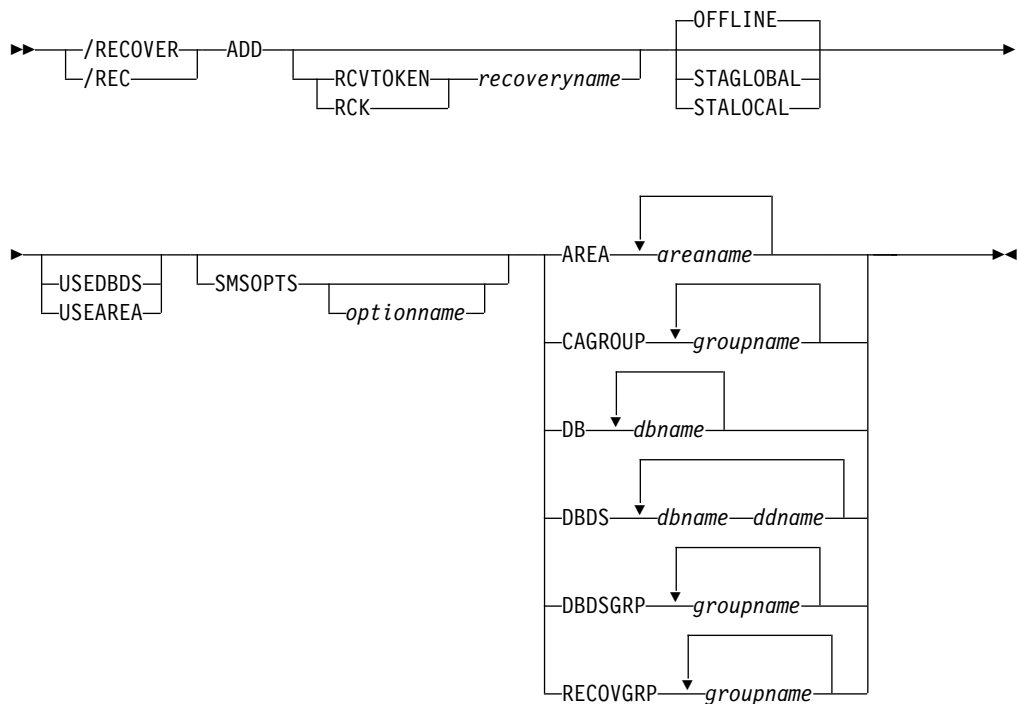
Keywords

The following keywords are valid for the /RECOVER ADD command:

OFFLINE

> With this option, the database remains offline from the IMS system at the
> completion of the recovery. This option is used to enable the database
> administrator to verify that the recovery has completed successfully and the
> databases are ready for use. This is the default if it is not specified on the
> /RECOVER START command.

> The IMS system performing the recovery has exclusive authorization of the
> database to perform the recovery. The OFFLINE option allows the IMS system
> to deauthorize the databases at the completion of the recovery without issuing

a START command. This will leave the databases available for processing. If a PITR option has been used in a recovery, the database data set in the RECON is IMAGE COPY NEEDED.

STAGLOBAL

This option is used when the full-function database or databases and Fast Path areas are used in a sysplex data sharing environment. A /START DB command with the GLOBAL option is issued on the IMS system which ran the recovery for all full-function databases affected by recovery. A /START AREA command with the GLOBAL option is used on the IMS system which ran the recovery for all Fast Path areas recovered. This option makes the database or databases available to IMS transactions as soon as all the DBDSs in the recovery list have been recovered. Authorization to use the database is returned to IMS.

STALOCAL

This option is used when the DRF is being executed in the operational IMS system where the database or databases are used. A /START DB command is issued on the IMS system which ran the recovery for all full-function databases affected by recovery. A /START AREA is issued on the IMS system which ran the recovery for all Fast Path areas recovered. Fast Path databases are not started with the /START DB command. Only one option needs to be specified and it applies to all DBDSs and areas added to the recovery list.

USEDBDS | USEAREA

These options are for Fast Path areas, full-function, and HALDB DBDS. When specified, DRF does not need to restore the image copy before applying log updates. You can use this option when you restore image copies prior to recovery.

Note: You must restore non-standard image copies prior to recovery.

This parameter determines whether DRF will restore the image copy for a DBDS before applying log updates.

RCVTOKEN | RCK

Specifies the unique identifier associated with a recovery list that the /RECOVER ADD command operates against. RCVTOKEN is optional. If it is not specified, IMS generates a recovery name. However, if the command is intended to add entries to an existing recovery list, RCVTOKEN must be specified with the intended recovery list token *recoveryname*. When RCVTOKEN is specified in a command, it must come before any keywords that identify the names of database data sets, areas, or groups.

recoveryname

Specifies the unique recovery token associated with the recovery list that the /RECOVER ADD command operates against. This token can be up to eight characters in length.

SMSOPTS

Specifies that the DFSMSdss options are to be associated with the entries being added to the recovery list. The options are only used when restoring image copies created by the Image Copy 2 utility.

optionname

Specifies a unique SMS option.

DELCAT

Specifies that the data set is to be restored using the DFSMSdss optional keyword, DELETECATALOGENTRY.

Attention: This option is to be used with extreme care. `DELCAT` is required if SMSOPTS is supplied. This option allows you to recover from a scenario where entire volumes are lost but the catalog entries remain. When this option is specified, SMS deletes the prior catalog entries for the database data sets and areas being restored as part of recovery. Before using this option, read the caution under DELETECATALOGENTRY option of the RESTORE command.

AREA
Specifies that one or more Fast Path areas are to be added to a recovery list.

areaname
Specifies a unique Fast Path area.

CAGROUP
Specifies that one or more change accumulation groups, as defined in the RECON data sets, are to have their database data sets and areas added to a recovery list.

groupname
Specifies that the database data sets and areas belonging to the named CA group are to be added to the recovery list.

DB Specifies that all the areas or full-function database data sets for one or more databases are to be added to a recovery list.

dbname
Specifies the database and the associated database data sets or areas that are to be added to a recovery list.

DBDS
Specifies that one or more full-function database data sets are to be added to a recovery list.

dbname ddname
Specifies a full-function database data set is to be added to a recovery list. Full-function database data sets are specified with the /RECOVER ADD DBDS command as an ordered pair. The first member of the pair is the database name. The second member is the DD name. If more than one full-function database data set is specified, the complete ordered pair must be specified for each database data set. All parameters must be separated by at least one blank space.

DBDSGRP
Specifies that one or more DBDS groups as defined in the RECON data sets will have their database data sets and areas added to a recovery list.

groupname
Specifies that the database data sets and areas belonging to the named DBDS group are to be added to the recovery list.

RECOVGRP
Specifies that the listed groups are recovery groups. A recovery group is a group of full-function databases, DEDB areas, or both that the user defines to IMS as related. All DBDSs that make up the full-function databases and all the DEDB areas making up the recover groups specified in the command are added to a recovery list.

groupname
Specifies the unique name of the group whose database data sets and areas are to be added to a recovery list.

Usage notes

Successful completion of a /RECOVER ADD command results in the specified database data sets and areas being added to a recovery list. database data sets and areas can be added to a recovery list by specifying one or more database data sets, databases, change accumulation groups (CAGROUP), database data set groups (DBDSGRP), or recovery groups (RECOVGRP). If a database or group is specified, all database data sets and areas making up the database or group are added to a recovery list. If the specified database is a master database for a partitioned HALDB, all database data sets from all partitions that make up the HALDB are added to a recovery list. All groups (including databases) are defined in DBRC.

Database data sets and areas must be registered with DBRC to be recovered with the IMS Recovery Services. If the database data set, area, or group name is not known to DBRC, it is not added to a recovery list and a message is issued.

If a database data set or area specified in a /RECOVER ADD command (individually or as part of a group) is already on a recovery list, processing for the duplicate is ignored and a message is issued. Other database data sets and areas that are not duplicates are processed normally.

This command can be issued in IMS DBCTL and IMS DB/DC environments.

Examples

The COMMAND IN PROGRESS message is issued for /RECOVER ADD commands but is not shown in the following examples.

Example 1 for /RECOVER ADD command

In this example, a /RECOVER ADD STALOCAL command is issued for full-function database data sets. Following a successful recovery of the database data sets, the database is started on the IMS that runs the recovery.

```
/REC ADD STALOCAL DBDS DBNAME1 DDNAME1 DBNAME2 DDNAME2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
```

Example 2 for /RECOVER ADD command

In this example, a /RECOVER ADD command is issued for full-function database data sets. One of the database data sets is not registered in RECON and is rejected.

```
/RECOVER ADD DBDS DBNAME1 DDNAME1 DBNAME1 DDNAME2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6010W UNABLE TO ADD TO RECOVERY LIST, NOT FOUND IN RECON, DBNAME2 DDNAME2
```

Example 3 for /RECOVER ADD command

In this example, a /RECOVER ADD command is issued for full-function database data set. The database that database data set belongs to is still authorized to two IMS systems.

```
/RECOVER ADD DBDS DBNAME1 DDNAME1
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME1 DDNAME1 AUTHORIZED BY IMS1
DFS4299I FRD6003I DBNAME1 DDNAME1 AUTHORIZED BY IMS2
```

Example 4 for /RECOVER ADD command

Databases can be specified as a whole with the /RECOVER ADD DB command. In this example, a full-function database and a Fast Path database have all their database data sets and areas, respectively, added to the recovery list.

```
/REC ADD DB FFDB1 FPDB2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I FFDB1 DDNAME1
DFS4299I FRD6003I FFDB1 DDNAME2
DFS4299I FRD6003I DBAREA3 DDAREA3
DFS4299I FRD6003I DBAREA4 DDAREA4
```

Example 5 for /RECOVER ADD command

If more than one DATAGROUP is specified, the group names must be separated by at least one blank space. In this example, a /RECOVER ADD command is issued for two database groups.

```
/REC ADD DATAGROUP GRPNAME1 GRPNAME2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBNAME4 DDAREA4
DFS4299I FRD6003I DBNAME5 DDNAME5
DFS4299I FRD6003I DBNAME6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBNAME8 DDAREA8
```

Example 6 for /RECOVER ADD command

If one or more RECOVGRP is specified, the group names must be separated by at least one blank. In this example, a /RECOVER ADD command is issued for two recovery groups.

```
/REC ADD RECOVGRP GRPNAME1 GRPNAME2
DFS4299I FRD6011I THE FOLLOWING ENTRIES ARE ADDED TO THE RECOVERY LIST:
DFS4299I FRD6003I DBNAMEA DDNAMEA
DFS4299I FRD6003I DBNAMEB DDAREAB
DFS4299I FRD6003I DBNAMEC DDNAMEC
DFS4299I FRD6003I DBNAMED DDAREAD
DFS4299I FRD6003I DBNAMEE DDNAMEE
DFS4299I FRD6003I DBNAMEF DDAREAF
```

/RECOVER REMOVE command

Use the /RECOVER REMOVE command to remove some or all database data sets and areas from the recovery list.

It can only be issued prior to issuing the /RECOVER START command.

Subsections:
- "Environment"
- "Syntax" on page 757
- "Keywords" on page 757
- "Usage notes" on page 759
- "Examples" on page 759

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 243. Valid environments for the /RECOVER REMOVE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
ALLENTRIES	X	X	
AREA	X	X	
CAGROUP	X	X	
DB	X	X	
DBDS	X	X	
DBDSGRP	X	X	
RCVTOKEN	X	X	
RECOVGRP	X	X	
REMOVE	X	X	

Syntax
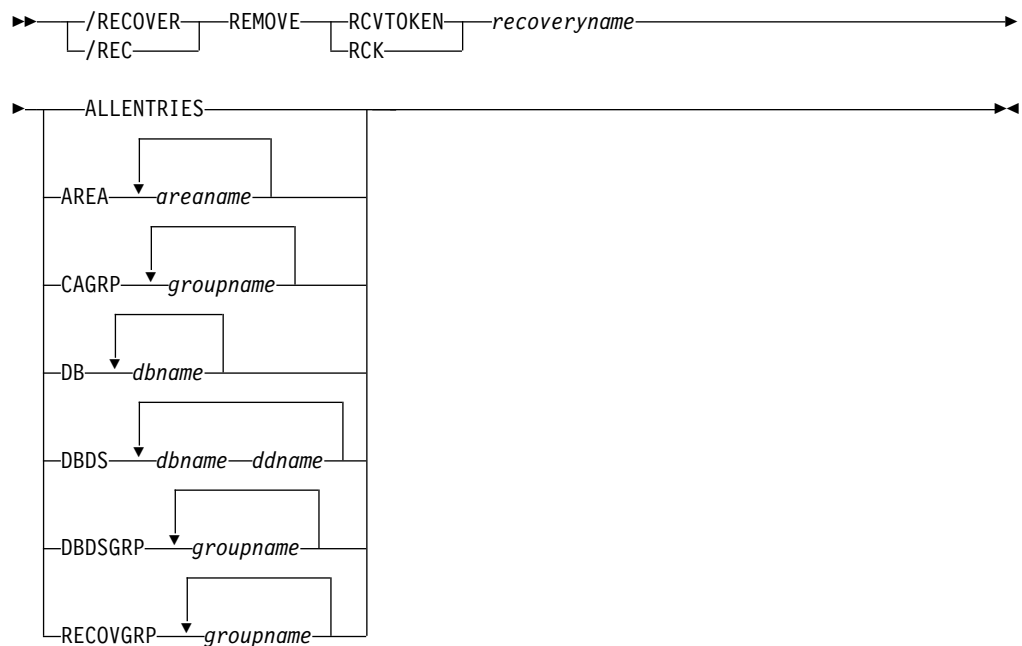
Keywords

The following keywords are valid for the /RECOVER REMOVE command:

RCVTOKEN | RCK

Specifies the unique identifier associated with the recovery list that the /RECOVER REMOVE command operates against. The RCVTOKEN keyword must come before any keywords that identify the names of database data sets, areas, or groups.

recoveryname

Specifies the unique recovery token associated with the recovery list that the /RECOVER REMOVE command operates against. This token can be up to eight characters in length.

ALLENTRIES
> Specifies that the recovery list is to be eliminated.

AREA
> Specifies that one or more Fast Path areas are to be removed from the recovery list.

> *areaname*
>> Specifies a unique Fast Path area.

CAGROUP
> Specifies that the database data sets and areas of one or more change accumulation groups as defined in the RECON data sets are to be removed from the recovery list.

> *groupname*
>> Specifies that the database data sets and areas belonging to a specific CA group are to be removed from the recovery list.

DATAGROUP
> Specifies that the database data sets and areas of one or more database groups (as defined in the RECON data sets) are to be removed from the recovery list.

> *groupname*
>> Specifies the database data sets and areas of the unique group name that are to be removed from the recovery list.

DB Specifies that the full-function database data sets or Fast Path areas making up one or more databases are to be removed from the recovery list.

> *dbname*
>> Specifies database data sets or areas of the database that are to be added to a recovery list.

DBDS
> Specifies that one or more full-function database data sets are to be removed from the recovery list.

> *dbname*
>> Specifies the database data sets or areas of the database that are to be removed from the recovery list.

> *ddname*
>> Specifies the DD name of the database data set. If DBDS is specified on the /RECOVER REMOVE command, *dbname* and *ddname* must be specified together.

DBDSGRP
> Specifies that the database data sets and areas of one or more DBDS groups as defined in the RECON data sets are to be removed from the recovery list.

> *groupname*
>> Specifies the database data sets and areas of the group that are to be removed from the recovery list.

RECOVGRP
> Specifies that this group is a recovery group. All DBDSs that make up the full-function databases and all the DEDB areas are removed from the recovery list.

> *groupname*
>> Specifies the database data sets and areas of the database that are to be removed from the recovery list.

Usage notes

Use the /RECOVER STOP command to remove entries after recovery has started.

- If /RECOVER REMOVE is issued before the /RECOVER START command, database data sets and areas specified on the /RECOVER REMOVE command individually or as part of databases or groups are removed from the recovery list. A subsequent /RECOVER START command initiates recovery for the remaining members in the recovery list.
- If a /RECOVER REMOVE ALLENTRIES command is issued before the /RECOVER START command, all elements in the list are removed, and the recovery list is eliminated.
- If the /RECOVER REMOVE command is issued after the /RECOVER START command, the /RECOVER REMOVE command is rejected.

If /RECOVER REMOVE is issued with one or more databases or groups, all database data sets and areas that are part of the database or group specified are removed from the recovery list. If a /RECOVER REMOVE command results in the removal of every data set or area entry from the recovery list, the recovery list is eliminated.

This command executes in IMS DBCTL and IMS DB/DC environments.

Examples

The command IN PROGRESS message is issued for /RECOVER commands but is not shown in these examples.

Example 1 for /RECOVER REMOVE command

As with the /RECOVER ADD command, full-function database data sets and Fast Path areas are specified with the /RECOVER REMOVE DBDS command. With the DBDS option, each full-function database data set must be specified as an ordered pair. Each element must be separated by at least one blank space. The first element of the pair is the database name. The second element is the DDNAME. In this example, a /RECOVER REMOVE command is issued for a single full-function database data set.

```
/RECOVER REMOVE RCVTOKEN DFS00001 DBDS DBNAME1 DDNAME1

DFS4299I FRD6016I THE FOLLOWING ENTRIES WERE REMOVED FROM THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
```

If more than one full-function database data set is specified in a /RECOVER REMOVE DBDS command, each dbname/ddname ordered pair must be separated by at least one blank space.

```
/RECOVER REMOVE RCVTOKEN DFS00001 DBDS DBNAME1 DDNAME1 DBNAME3 DDNAME3

DFS4299I FRD6016I THE FOLLOWING ENTRIES WERE REMOVED FROM THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME3 DDNAME3
```

Example 2 for /RECOVER REMOVE command

In this example, a /RECOVER REMOVE command is issued for a single Fast Path area that was not added to the recovery list.

```
/REC REMOVE RCVTOKEN DFS00001 AREA DDAREA1

DFS4299I FRD6018W UNABLE TO REMOVE AREA DDAREA1: NOT IN RECOVERY LIST
```

Example 3 for /RECOVER REMOVE command

In this example, a /RECOVER REMOVE command is issued for a full-function
database and Fast Path database. All full-function database data sets and Fast Path
areas making up the two databases are removed from the recovery list.

```
/REC REMOVE RCVTOKEN DFS00001 DB FFDB1 FFDB2

DFS4299I FRD6016I THE FOLLOWING ENTRIES WERE REMOVED FROM THE RECOVERY LIST:
DFS4299I FRD6003I FFDB1 DDNAME1
DFS4299I FRD6003I FFDB1 DDNAME2
DFS4299I FRD6003I DBAREA3 DDAREA3
DFS4299I FRD6003I DBAREA4 DDAREA4
```

Example 4 for /RECOVER REMOVE command

In this example, a /RECOVER REMOVE command is issued to stop recovery for
the entire recovery list.

```
/REC REMOVE RCVTOKEN DFS00001 ALLENT

DFS4299I FRD6016I THE FOLLOWING ENTRIES WERE REMOVED FROM THE RECOVERY LIST:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBNAME5 DDNAME5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBAREA7 DDAREA7
DFS4299I FRD6003I DBAREA8 DDAREA8
DFS4299I FRD6017I RECOVERY LIST IS NOW EMPTY
```

Related reference:

"/RECOVER START command"

/RECOVER START command

Use the /RECOVER START command to start the recovery process for all the
members of a recovery list.

The recovery process includes performing the following tasks:
- Image copies are restored to the database data sets and areas in the recovery list.
- Change accumulation data is applied to the database data sets and areas in the
  recovery list.
- The database data sets and areas are brought up to date by applying data
  changes from log data sets (or up to the recovery time).
- Log data sets cached to a VTS are staged to DASD as a user option.

Subsections:
- "Environment" on page 761
- "Syntax" on page 761
- "Keywords" on page 761
- "Usage notes" on page 763
- "Examples" on page 763

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 244. Valid environments for the /RECOVER START command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
ERRORABORT	X	X	
ERRORCONT	X	X	
NOCHECK	X	X	
OFFLINE	X	X	
PITR	X	X	
RCVTIME	X	X	
RCVTOKEN	X	X	
READNUM	X	X	
REMOVE	X	X	
STAGLOBAL	X	X	
STALOCAL	X	X	
START	X	X	
VERIFY	X	X	

Syntax
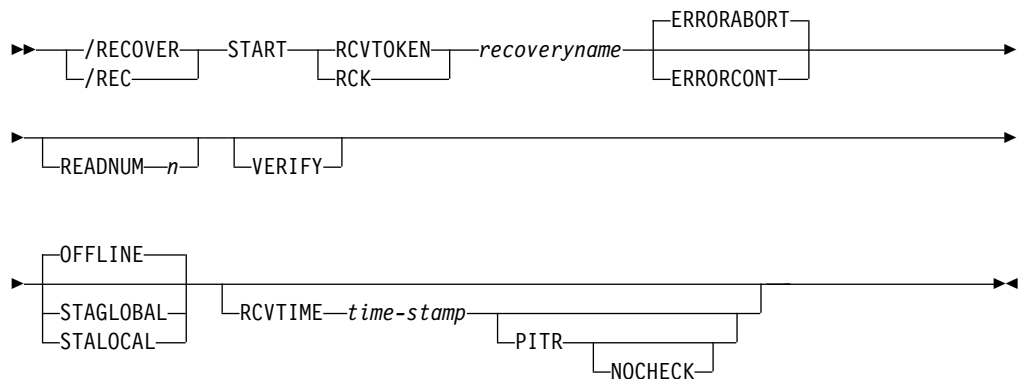
Keywords

The following keywords are valid for the /RECOVER START command:

RCVTOKEN | RCK
    Specifies the token of the recovery list that will be processed. When RCVTOKEN is specified in a command, it must come before any keywords that identify the names of database data sets, areas, or groups.

> *recoveryname*
>   Specifies the unique recovery token associated with the recovery list to be processed. This token can be up to eight characters in length.

ERRORABORT

Specifies that recovery stops for all entries in the recovery list if recovery of any database data set can not be completed. ERRORABORT is the default.

ERRORCONT

Specifies that recovery is to continue if recovery is able to complete processing for any database data set.

READNUM *n*

Specifies the number of input devices used in parallel during recovery. Image copies are restored using the number of input devices specified by *n*. When image copies are restored, log data sets are read using the number of input devices specified by *n*.

VERIFY

Specifies the report only execution and obtains a list of the log, change accumulation, and image copy data sets required to process recovery for the associated recovery list. This option allows users to perform set up procedures before invoking the recovery process.

OFFLINE

This option leaves all the databases offline after the recovery is complete. When the recovery is complete, DRF will deauthorize the databases. This leaves the databases in a state that allows normal DBRC processing.

STAGLOBAL

This option is used in a sysplex data sharing environment or with two IMS systems sharing data on the same central processing complex (CPC). A /START DB command with the GLOBAL option is issued internally for DL/I databases and a /START AREA command with the GLOBAL option is issued for the Fast Path areas. OFFLINE, STALOCAL, or STAGLOBAL options specified on the /RECOVER START command are used only for database data sets and areas that were added and did not have any of those options specified.

STALOCAL

This option is used to start the databases on the IMS system that ran the DRF recovery. A /START DB command with the LOCAL option is issued internally.

RCVTIME

Specifies the time stamp to which a point in time or time stamp recovery is to be performed.

time-stamp

The time stamp must have a format that is recognizable to IMS. Note that the UTC offset portion of a time stamp cannot be specified using a symbolic value in this command.

Additionally, the time stamp must by surrounded by single quotation marks ('). For example:

```
/RECOVER START RCVTOKEN R1 RCVTIME '022671213156'
```

PITR

Specifies that a time stamp recovery (TSR) will be performed to the time specified with the RCVTIME parameter regardless if there are any active database allocations for the specified database data sets.

NOCHECK

If a portion of the database data sets making up a database are in the recovery list being started, NOCHECK specifies that the Database Recovery Facility will

not stop a time stamp recovery or a time stamp recovery to any prior point in time (PITR) if one of the following situations occurs:

- All members of the recovery group are not in the same recovery list.
- All members of the recovery group are not being recovered to an equivalent point in time.

Usage notes

Only one /RECOVER START command is allowed to execute in one IMS at a time. If DRF is to run in conjunction with multiple IMS systems simultaneously, ensure that log contention situations do not occur. To avoid log contention situations, ensure that recovery instances that would read the same log data sets do not execute simultaneously in multiple IMS systems.

You can choose to automatically /START any or all members of the recovery list after successful completion of recovery, either on all IMS systems on which they are defined or just the one where the recovery is executed.

If ERRORABORT is in effect, the recovery list will not start until all the DBDSs in the list can be authorized for recovery. This is not true if the ERRORCONT parameter is specified on the /RECOVER START command. Instead, the recovery will continue.

If coordinated online change removes from the system any database data set or area that has been previously added to a recovery list, message DFS4266I with reason code NOT FOUND will be issued after the /RECOVER START command for that recovery list is entered.

Examples

The `command IN PROGRESS` message is issued for /RECOVER commands but is not shown in the following examples.

Example 1 for /RECOVER START command

In this example, /RECOVER START initiates recovery for the database data sets and areas from previous examples. Recovery continues until it completes or until one of the database data sets or areas is operable.

```
/REC START RCVTOKEN RCVTKN1 ERRORCONT
DFS4299I FRD6021I RECOVERY STARTED FOR:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
```

Example 2 for /RECOVER START command

In this example, /RECOVER START RCVTOKEN initiates recovery for the database data sets and areas owned by the recovery token RCVTKN2.

```
/REC START RCVTOKEN RCVTKN2
DFS4299I FRD6021I RECOVERY STARTED FOR:
DFS4299I FRD6003I DBNAME1 DDNAME1
```

```
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
```

Example 3 for /RECOVER START command

In this example, /RECOVER START RCVTOKEN OFFLINE READNUM 6 initiates
recovery for the database data sets and areas from previous examples. Recovery
will not continue if any error is detected for any member of the recovery list. The
database data sets and areas remain offline after recovery completes.

```
/RECOVER START RCVTOKEN RCVTKN2 OFFLINE READNUM 6
DFS4299I FRD6021I RECOVERY STARTED FOR:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
```

Example 4 for /RECOVER START command

In this example, /RECOVER START RCVTOKEN ERRORCONT RCVTIME
time-stamp is issued. TSR continues until it completes or until one of the database
data sets or areas undergoing recovery encounters an error. After recovery
completes, a message is issued listing each database data set and area successfully
recovered.

```
/REC START RCVTOKEN RCVTKN2 ERRORABORT RCVTIME '020011015257' NOCHECK
DFS4299I FRD6021I RECOVERY STARTED FOR RCVTKN2, ERRORABORT, TSR
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
    ...
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME1 DDNAME1
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME2 DDNAME2
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA1 DDAREA1
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME3 DDNAME3
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA4 DDAREA4
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA5 DDAREA5
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA6 DDAREA6
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME7 DDNAME7
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA8 DDAREA8
DFS4277I RECOVERY COMPLETE FOR:  RCVTKN2
```

Example 5 for /RECOVER START command

In this example, /RECOVER START RCVTOKEN RCVTIME*time-stamp* PITR is
issued. Point-in-time recovery will continue until it completes or until one of the

database data sets encounters an error. A message is issued listing the database
data sets and areas that were not in the recovery list, but might need recovery
using point-in-time recovery.

```
/REC START RCVTOKEN RCVTKN2 RCVTIME '020011015257' PITR
DFS4299I FRD6021I RECOVERY STARTED FOR RCVTKN2, ERRORCONT, PITR
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
DFS4299I FRD6003I DBAREA5 DDAREA5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBNAME7 DDNAME7
DFS4299I FRD6003I DBAREA8 DDAREA8
   ...
DFS4299I FRD6024A GROUP MEMBER DBNAME9 DDNAME9 NOT IN RECOVERY LIST: MEMBER OF A GROUP
DFS4299I FRD6024A GROUP MEMBER DBNAMEA DDNAMEA NOT IN RECOVERY LIST: MEMBER OF A GROUP
DFS4299I FRD6024A GROUP MEMBER DBAREAB DDAREAB NOT IN RECOVERY LIST: MEMBER OF A GROUP
DFS4299I FRD6024A GROUP MEMBER DBNAMEC DDNAMEC NOT IN RECOVERY LIST: MEMBER OF A GROUP
   ...
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME1 DDNAME1
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME2 DDNAME2
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA1 DDAREA1
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME3 DDNAME3
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA4 DDAREA4
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA5 DDAREA5
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA6 DDAREA6
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBNAME7 DDNAME7
DFS4299I FRD4031I DATASET RESTORE COMPLETE:  DBAREA8 DDAREA8
DFS4277I RECOVERY COMPLETE FOR:  RCVTKN2
```

Related reference:

"/RECOVER REMOVE command" on page 756

"/RECOVER STOP command"

➡ Examples for the DFSURDB0 utility (Database Utilities)

/RECOVER STOP command

Use the /RECOVER STOP command to stop recovery for all database data sets
and areas on the recovery list.

Subsections:

- "Environment"
- "Syntax" on page 766
- "Keywords" on page 766
- "Usage notes" on page 767
- "Examples" on page 767

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 245. Valid environments for the /RECOVER STOP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
STOP	X	X	

Syntax

```
►►──┬─/RECOVER─┬──STOP──┬──ALLENTRIES─────────┬──────────────────────────►◄
    └─/REC─────┘        ├──ALLENT────┬──SAVE──┘
                        │            └────────┘
                        │              ┌──,───┐
                        │              ▼      │
                        ├──AREA─────────areaname───┤
                        │
                        │              ┌──,───┐
                        │              ▼      │
                        ├──CAGRP────────groupname───┤
                        │
                        │          ┌──,───┐
                        │          ▼      │
                        ├──DB────────dbname───┤
                        │
                        │            ┌──,───┐
                        │            ▼      │
                        ├──DBDS───────dbname──ddname───┤
                        │
                        │               ┌──,───┐
                        │               ▼      │
                        ├──DBDSGRP───────groupname───┤
                        │
                        │               ┌──,───┐
                        │               ▼      │
                        └──RECOVGRP──────groupname───┘
```
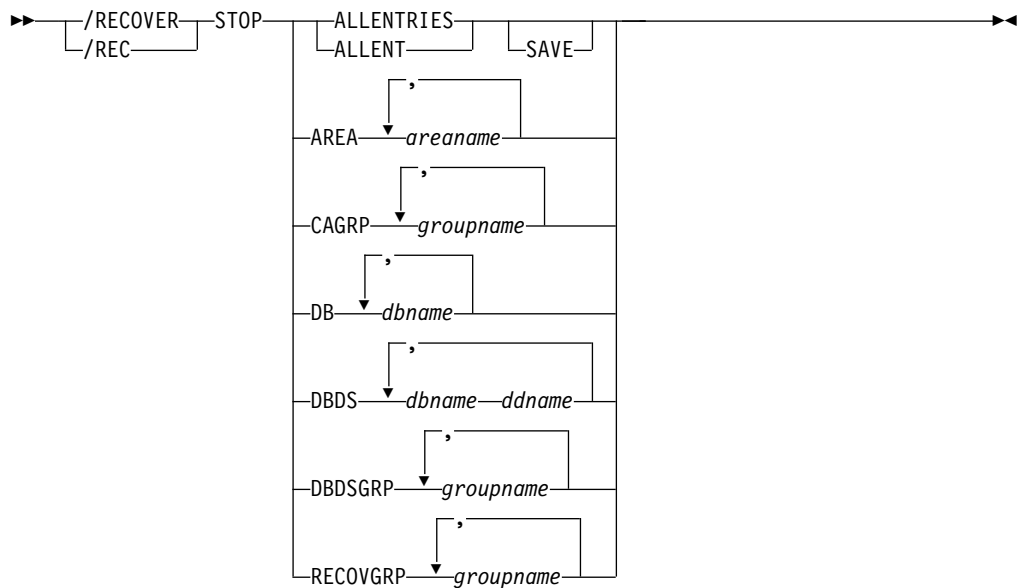
Keywords

The following keywords are valid for the /RECOVER STOP command:

ALLENTRIES
> Specifies that recovery is to be aborted for all database data sets and areas (all entries) in the recovery list.

> **SAVE**
>> Specifies that the recovery list is not to be deleted when recovery is stopped. This parameter is only allowed with the ALLENT parameter after recovery has been initiated with the /RECOVER START command.

AREA
> Specifies that recovery processing is to be stopped for the specified Fast Path areas.

> *areaname*
>> Specifies a unique Fast Path area.

CAGROUP
> Specifies that recovery processing is to be stopped for the database data sets and areas making up the specified change accumulation groups as defined in the RECON data sets.

> *groupname*
>> Specifies the unique name of the group whose database data sets and areas are to have recovery processing be stopped.

DB Specifies that recovery processing is to be stopped for the full-function database data sets or Fast Path areas that make up the specified databases.

> *dbname*
>> Specifies the database whose database data sets or areas are to be added to a recovery list.

DBDS

Specifies that recovery processing is to be stopped for the specified full-function database data sets.

dbname

Specifies the database whose database data sets or areas are to be added to a recovery list.

ddname

The 8 character identifier associated with the data set name and data set characteristics. *dbname* and *ddname* must be specified together if DBDS is specified on the /RECOVER REMOVE command.

DBDSGRP

Specifies that recovery processing is to be stopped for the database data sets and areas making up the specified DBDS groups as defined in the RECON data sets.

groupname

Specifies the unique name of the group whose database data sets and areas are to have recovery processing be stopped.

RECOVGRP

Specifies that this group is a recovery group. A recovery group is a group of full-function databases or DEDB areas that are considered to be related. All DBDSs that make up the full-function databases and all the DEDB areas are removed from the recovery list.

groupname

Specifies the unique name of the group whose database data sets and areas are to have recovery processing be stopped.

Usage notes

The command can only be issued for a recovery list that has had /RECOVER START issued against it. If the /RECOVER STOP is issued before the /RECOVER START command, it is rejected. If it is issued after the /RECOVER START command, recovery is stopped for all database data sets in the recovery list. After a /RECOVER STOP command successfully processes, subsequent /RECOVER STOP commands are rejected.

If /RECOVER STOP ALLENT is issued, all recovery processing for the affected recovery list halts, and the existing recovery list is deleted.

Examples

The `COMMAND IN PROGRESS` message is issued for /RECOVER commands but is not shown in the following examples.

Example 1 for /RECOVER STOP command

In this example, a /RECOVER STOP command is issued to stop recovery for the entire recovery list.

```
/REC STOP ALLENT
DFS4299I FRD6032I THE FOLLOWING ENTRIES WILL HAVE RECOVERY STOPPED:
DFS4299I FRD6003I DBNAME1 DDNAME1
DFS4299I FRD6003I DBNAME2 DDNAME2
DFS4299I FRD6003I DBAREA1 DDAREA1
DFS4299I FRD6003I DBNAME3 DDNAME3
DFS4299I FRD6003I DBAREA4 DDAREA4
```

```
DFS4299I FRD6003I DBNAME5 DDNAME5
DFS4299I FRD6003I DBAREA6 DDAREA6
DFS4299I FRD6003I DBAREA7 DDAREA7
DFS4299I FRD6003I DBAREA8 DDAREA8
DFS4299I FRD6033I ALL ENTRIES IN RECOVERY LIST, ARE BEING STOPPED
```

Example 2 for /RECOVER STOP command

In this example, a /RECOVER STOP ALLENT SAVE is issued after a /RECOVER
START command.

```
/REC STOP ALLENT SAVE
DFS4299I FRD6032I THE FOLLOWING ENTRIES WILL HAVE RECOVERY STOPPED:
DFS4299I FRD6003I DBNAME1 DDNAME1
```

Example 3 for /RECOVER STOP command

In this example, a /RECOVER STOP ALLENT command is issued with no
recovery in progress.

```
/REC STOP ALLENT
DFS4299I FRD6031E UNABLE TO STOP ALLENT: RECOVERY NOT IN PROGRESS
```

Related reference:

"/RECOVER START command" on page 760

"/RECOVER TERMINATE command"

/RECOVER TERMINATE command

Use the RECOVER TERMINATE command to delete all lists in BEING BUILT
status and to terminate the DRS address space.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes" on page 769
- "Example" on page 769

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 246. Valid environments for the /RECOVER TERMINATE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RECOVER	X	X	
TERMINATE	X	X	

Syntax

```
►►──┬─/RECOVER─┬──TERMINATE───────────────────────────────────────►◄
    └─/REC─────┘
```

Usage notes

If a recovery is in progress, the recovery will ignore the /RECOVER TERMINATE command and continue to process. When it completes, another /RECOVER TERMINATE command is required. It will not take effect automatically. If you would like to force DRF down while a recovery is running, you must issue the /RECOVER STOP ALLENT command first. This will stop the recovery, and then you can enter the /RECOVER TERMINATE command to terminate the DRF address space.

Example

In this example, a /RECOVER TERMINATE command is issued with no recovery in progress.

```
/RECOVER TERMINATE
DFS4299I FRD4202I DATABASE RECOVERY DATA MANAGER TERMINATION COMPLETE
```

Related reference:

Chapter 12. REFRESH USEREXIT command

Use the type-2 REFRESH USEREXIT command to refresh the user exit modules that are defined in the USER_EXITS section of the DFSDFxxx member. When the command is processed, IMS rereads the DFSDFxxx member and processes the USER_EXITS section.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 773
- "Output fields" on page 774
- "Return, reason, and completion codes" on page 774
- "Examples" on page 776

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the REFRESH command and keywords can be issued.

Table 247. Valid environments for the REFRESH USEREXIT command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
REFRESH USEREXIT	X	X	X
TYPE	X	X	X
MEMBER	X	X	X

Syntax

```
>>--REFRESH USEREXIT--TYPE(----+---------+-----)-------------------------><
                           |   |   ,   |        |
                           v   +-------+        |
                           +---exit_type--+     |
                                          |     |
                                          +-----+
                                    +--MEMBER(suffix)--+
```

Keywords

The following keyword is valid for the REFRESH USEREXIT command:

TYPE(*exit_type*)
Specifies the user exit type or types that you want to be refreshed. You can specify a single user exit type or a list of user exit types separated by commas. The valid user exit types are the following:

AOIE
Type-2 Automated Operator User Exit

BSEX
Build Security Environment user exit

ICQSEVNT
IMS CQS Event user exit

ICQSSTEV
IMS CQS Structure Event user exit

| **IMSMON**
| IMS Monitor user exit

INITTERM
Initialization/Termination user exit

LOGEDIT
Log Edit user exit

LOGWRT
Logger user exit

NDMX
Non-Discardable Messages user exit

OTMAIOED
OTMA Input/Output Edit user exit

OTMAYPRX
OTMA Destination Resolution user exit

OTMARTUX
OTMA Resume TPIPE Security user exit

PPUE
Partner Product user exit

RASE
Resource Access Security user exit

RESTART
Restart user exit

If an exit type is specified more than once, it is refreshed only once and no error message is issued.

MEMBER(*suffix*)
Specifies a 1- to 3-alphanumeric character value that represents the suffix to a DFSDFxxx member name. The identified DFSDFxxx member is read in and parsed for the EXITDEF statements for the requested user exit types. If the keyword is not specified, the DFSDFxxx member specified for system initialization is used as the default. The DFSDFxxx member specified on system initialization can be determined from message DFS1929I on the system console or log. The command fails if the MEMBER keyword is specified with an invalid xxx value or if the DFSDFxxx member is not found.

This keyword is optional. Normally, the DFSDFxxx member specified on system initialization should be the only one available. The purpose of this keyword is to provide the capability of having the command point to an alternate DFSDFxxx member (different from the system default) for test purposes. Outside of testing, IMS expects the MEMBER keyword to be omitted.

Note: These changes are not saved across a restart. If you refresh your user exits from a DFSDFxxx member that is not the member specified on IMS initialization and you do not update your initialization DFSDFxxx member, the user exit changes will be lost when IMS restarts.

If the MEMBER parameter is specified without any value (MEMBER()), it is treated as if MEMBER is not specified. The DFSDFxxx member specified for initialization will be used.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The output of this command is defined in XML and is available to automation programs that communicate with OM.

When the REFRESH USEREXIT command is entered, IMS performs the following steps:

1. Reads the DFSDFxxx member and process the USER_EXITS section of the DFSDFxxx member.
2. Loads the user exit modules specified in the USER_EXITS section for the exit types specified in the command.
3. Updates the internal IMS control block with pointers to the new user exit modules. Any subsequent calls to the user exit modules will now call the new modules.
4. When the processing has completed in the old exit modules, the old modules will be deleted.

IMS loads the new user exit modules before deleting the old modules. If an error occurs during this process (for example, a module could not be loaded), IMS fails the command for the particular user exit type and leaves the current modules of the user exit type in effect. All modules of the specified user exit type must be loaded successfully for the command to complete successfully.

If your IMSplex has multiple IMS systems, you can refresh the user exit modules in all of the IMS systems in the IMSplex by using the default routing for the command. This sends the REFRESH USEREXIT command to all of the IMS systems in the IMSplex and the user exit modules are refreshed in each of the IMS systems. If the refresh fails on one or more IMS systems, you must resolve the problem that caused the command to fail and reenter the command.

You can use the REFRESH USEREXIT command to add a user exit type to IMS, delete a user exit type, or migrate an existing user exit to be dynamically refreshable:

- To add a user exit type, insert EXITDEF statements into the USER_EXITS section of the DFSDFxxx member for the user exit type you want to add and then enter the REFRESH USEREXIT command for that exit type. If the user exit has an initialization function, the user exit is called with the initialization function before the user exit is made available to IMS.
- To delete a user exit type, remove the EXITDEF statements from the USER_EXITS section of the DFSDFxxx member for the user exit type you want to delete and then enter the REFRESH USEREXIT command for that exit type.
- To migrate an existing user exit, specify the exit routines on the EXITDEF parameter in the USER_EXITS section of the DFSDFxxx member and then enter the REFRESH USEREXIT command for that exit type.

Refreshes are performed at the exit type level, so when an exit routine of a certain exit type is modified, every exit routine of that exit type is refreshed.

Certain messages contain the short form of the command to which the message refers. Because the REFRESH USEREXIT command does not have a short form, the first four characters of the command (REFR) are specified in the message. These four characters are also used to define the command security.

Output fields

The following table shows the REFRESH USEREXIT output fields. The columns in the table are:

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning
Provides a brief description of the output field.

Table 248. Output fields for the REFRESH command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	N/A	Completion code for the line of output. The completion code is always returned.
CCTXT	CCText	N/A	N/A	Completion code text that briefly explains the meaning of the non-zero completion code.
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. Member name is always returned.
TYPE	ExitType	TYPE	LCL	User exit type requested by the REFRESH USEREXIT command. User exit type is always returned.
NAME	ModName	N/A	LCL	Exit routine name that was loaded by this REFRESH USEREXIT command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the REFRESH USEREXIT command are standard for all commands entered through the OM API. For a list of the codes and their meaning, see CSLOMCMD: command request (System Programming APIs).

The following table contains the return, reason, and completion codes for the REFRESH USEREXIT command. Included in the tables is a brief explanation of the codes.

Table 249. Return and reason codes for the REFRESH USEREXIT command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The REFRESH USEREXIT command completed successfully.
X'00000004'	X'00001024'	The REFRESH USEREXIT command completed successfully and a message was issued.
X'00000008'	X'00002014'	The DFSDFxxx suffix specified in the MEMBER parameter contains an invalid character.
X'0000000C'	X'00003000'	The REFRESH USEREXIT command was successful for at least one user exit type. The REFRESH command was not successful for one or more user exit types. The completion code indicates the reason for the error with the user exit type. The completion codes that can be returned by the REFRESH command are listed in Table 250.
X'0000000C'	X'00003004'	The REFRESH USEREXIT command was not successful for any of the user exit types specified. The completion code indicates the reason for the error with the user exit type. The completion codes that can be returned by the REFRESH command are listed in Table 250.
X'00000014'	X'00005004'	The REFRESH USEREXIT command processing terminated, because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The REFRESH USEREXIT command processing terminated because of an internal error.

Errors that are unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

The following table contains completion codes that can be returned on a REFRESH USEREXIT command.

Table 250. Completion codes for the REFRESH USEREXIT command

Completion code	Meaning
0	The REFRESH USEREXIT command completed successfully for the user exit type.
92	An Error was encountered while processing the REFRESH USEREXIT command. Check the messages that are returned with the command output for additional information about the error.
148	The specified user exit type was successfully deleted and a message was issued.
14E	The specified user exit type is deleted and will no longer be called.

Table 250. Completion codes for the REFRESH USEREXIT command (continued)

Completion code	Meaning
14F	The specified user exit type could not be added because there was no EXITDEF for the user exit type in the DFSDFxxx member or because no log record types existed in the DFSVSMxx member (message DFS4586E).

Examples

The following are examples of the REFRESH USEREXIT command:

Example 1 for REFRESH command

TSO SPOC input:
```
REFRESH USEREXIT TYPE(ICQSEVNT)
```

TSO SPOC output:
```
ExitType ModName  MbrName   CC
ICQSEVNT DFSCQEX0 SYS3       0
ICQSEVNT DFSCQEX1 SYS3       0
ICQSEVNT DFSCQEX2 SYS3       0
```

OM API input:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2012.061 20:33:01.174845</statime>
<stotime>2012.061 20:33:01.330060</stotime>
<staseq>C9344A682C43D7AA</staseq>
<stoseq>C9344A685228CD2C</stoseq>
<rqsttkn1>USRT002 10123301</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>REFR</verb>
<kwd>USEREXIT        </kwd>
<input>REFRESH USEREXIT TYPE(ICQSEVNT) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TYPE" llbl="ExitType" scope="LCL" key="YES" len="8"
 dtype="CHAR" align="left" />
<hdr slbl="NAME" llbl="ModName" scope="LCL" key="YES" len="8"
 dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" key="NO" len="4"
 dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" key="NO" len="4" dtype="INT"
 align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>TYPE(ICQSEVNT) NAME(DFSCQEX1) MBR(SYS3) CC(   0) </rsp>
<rsp>TYPE(ICQSEVNT) NAME(DFSCQEX2) MBR(SYS3) CC(   0) </rsp>
<rsp>TYPE(ICQSEVNT) NAME(DFSCQEX0) MBR(SYS3) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: ICQSEVNT exit types are refreshed.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

USER_EXITS section of the DFSDFxxx member (System Definition)

Chapter 13. /RELEASE command

The /RELEASE command resumes a conversation that was previously saved by means of the /HOLD command.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Example" on page 780

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command and keyword can be issued.

Table 251. Valid environments for the /RELEASE command and keyword

Command / Keyword	DB/DC	DBCTL	DCCTL
/RELEASE	X		X
CONVERSATION	X		X

Syntax

```
►►─┬─/RELEASE─┬─CONVERSATION──conv#─────────────────────────────────►◄
   └─/REL─────┘
```

Keywords

The following keywords are valid for the /RELEASE command:

CONVERSATION
　　Specifies the 4-digit identification (including leading zeros) of the conversation to be resumed; CONVERSATION conv# is the 4-digit identification conv# that was provided when the conversation was previously held.

Usage notes

The last message sent to the terminal before /HOLD was entered is sent to the terminal again.

/RELEASE is not valid from an LU 6.2 device. LU 6.2 communications cannot release a conversation, whether started by itself or by another communications protocol.

If global resource information is kept in Resource Manager, /RELEASE updates the conversation globally in Resource Manager. If global resource information is not kept in Resource Manager, /RELEASE updates the conversation locally.

Example

Entry ET:
```
/RELEASE CONVERSATION 0001
```

Response ET:

IMS does not respond to this command except to resend the last message.

Explanation: Conversation 0001 has been released and can be resumed by the terminal operator.

Chapter 14. /RESET command

The /RESET command eliminates the preset mode established by the /SET command.

Subsections:
- "Environment"
- "Syntax"
- "Example"

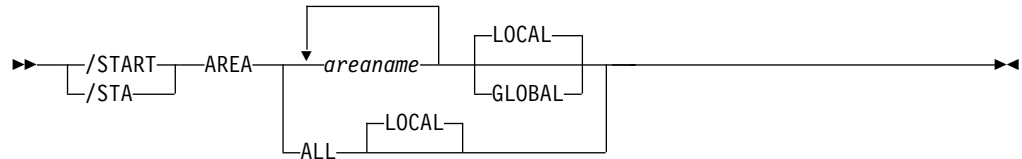## Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command can be issued.

Table 252. Valid environments for the /RESET command

Command	DB/DC	DBCTL	DCCTL
/RESET	X		X

Syntax

```
►►──┬─/RESET─┬──────────────────────────────────────────────►◄
    └─/RES───┘
```
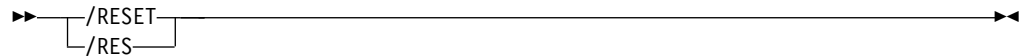
Example

Entry ET:

```
/RESET
```

Response ET:

```
DFS058I  RESET COMMAND COMPLETED
```

Explanation: The preset mode is no longer in effect.

Chapter 15. /RMxxxxxx commands

The /RMxxxxxx commands are multisegment commands that call functions of IMS Database Recovery Control (DBRC). Each /RMxxxxxx command is an online version of the corresponding batch DBRC command.

The following table lists the DBRC commands that are supported online and describes the utility function of each command:

Table 253. Functions of the DBRC commands supported online

Command	Utility function
/RMCHANGE	Changes or modifies information in the RECON data set
/RMDELETE	Deletes information in the RECON data set
/RMGENJCL	Generates JCL for: • IMS Change Accumulation utility • IMS Log Archive utility • IMS Log Recovery utility • IMS Database Image Copy utility • Database Image Copy 2 • IMS Online Database Image Copy utility • Database Recovery utility • User-defined output
/RMINIT	Creates records in the DBRC RECON data set
/RMLIST	Lists information contained in the RECON data set
/RMNOTIFY	Notifies DBRC about additional information, which gets recorded in the RECON data set.

These commands enable the IMS master terminal operator or an authorized terminal operator to run certain DBRC utility functions online. For any /RMxxxxxx commands other than /RMLIST, output is limited to what can be put in a 4 KB buffer. For the /RMLIST command that is issued from non-OM API (for example, IMS terminals or master terminals), output is limited to 32 KB. There is no limit to the output generated and returned by the /RMLIST command issued from OM API.

Subsections:

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command and keyword can be issued.

Table 254. Valid environments for the /RMxxxxxx command and keyword

Command / Keyword	DB/DC	DBCTL	DCCTL
/RMxxxxxx	X	X	X
LTERM	X		X

Syntax
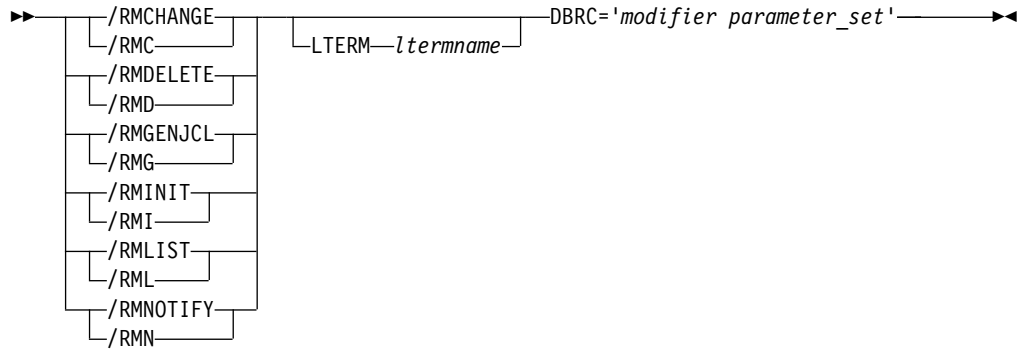
Keywords

The following keywords are valid for the /RMxxxxxx command:

LTERM *ltermname*
>   Specifies the logical terminal designated for output. If you omit the LTERM keyword, the output destination is the input terminal.
>
>   **Recommendation:** Because some of the DBRC commands generate a large amount of output, especially the /RMGENJCL and /RMLIST commands, direct the output to a printer.

DBRC='*modifier parameter_set*'
>   Specifies the DBRC modifier for the function specified and the parameters that will be passed to DBRC. The DBRC= parameter is required on all /RMxxxxxx commands.
>
>   *modifier*
>>   The DBRC modifier for the function specified. The following table lists the DBRC modifiers and the /RMxxxxxx commands with which the modifiers can be issued.

Table 255. DBRC modifiers for the /RMxxxxxx commands

Modifier	/RMxxxxxx commands					
	CHANGE	DELETE	GENJCL	INIT	LIST	NOTIFY
ADS	X	X		X		
ALLOC		X				X
ARCHIVE			X			
BKOUT	X	X			X	X
CA	X	X	X	X		X
CAGRP	X	X		X	X	
CLOSE			X			

Table 255. DBRC modifiers for the /RMxxxxxx commands (continued)

Modifier	/RMxxxxxx commands					
	CHANGE	**DELETE**	**GENJCL**	**INIT**	**LIST**	**NOTIFY**
DB	X	X		X	X	
DBDS	X	X		X	X	
DBDSGRP	X	X		X	X	
GSG		X		X	X	
HISTORY					X	
IC	X	X	X	X		X
LOG		X			X	
OIC			X			
PART				X		
PRILOG	X					X
RECON	X				X	
RECOV		X	X			X
REORG		X				X
SECLOG	X					X
SG	X	X		X		
SUBSYS	X	X			X	X
UIC	X	X				X
USER			X			

> *parameter_set*
>> Represents the required and optional parameters that will be passed to
>> DBRC. For an explanation of the values that can be specified, see the
>> corresponding command. For example, if you specify /RMLIST
>> LTERM(*ltermname*) DBRC='DBDS *parameter_set*', see the description of the
>> LIST.DBDS command to understand what parameters can be used in place
>> of *parameter_set*.

Table 256. /RMxxxxx commands and corresponding DBRC commands

Command	Corresponding DBRC command
/RMCHANGE	For the parameters that can be issued with the /RMCHANGE command, see CHANGE commands (Commands).
/RMDELETE	For the parameters that can be issued with the /RMDELETE command, see DELETE commands (Commands).
/RMGENJCL	For the parameters that can be issued with the /RMGENJCL command, see GENJCL commands (Commands).
/RMINIT	For the parameters that can be issued with the /RMINIT command, see INIT commands (Commands).
/RMLIST	For the parameters that can be issued with the /RMLIST command, see LIST commands (Commands).
/RMNOTIFY	For the parameters that can be issued with the /RMNOTIFY command, see NOTIFY commands (Commands).

Usage notes

All /RMxxxxxx formats require an EOM indication to denote end-of-message. An EOS indication must be included for all segments, if any, that precede the last segment. If comments are included with the /RMxxxxxx commands, they must be enclosed in asterisks.

If a failure other than the loss of both RECON data sets occurs while DBRC is processing an online command, DBRC makes the command unavailable for the remaining time the IMS online region is running. After determining and correcting the cause of the original failure, the command can be made available again by resubmitting the online command with the RESET parameter specified in the parameter set. It is the verb, rather than the modifier, level of the command that DBRC makes unavailable. That is, if a DBRC INIT.DB command fails, DBRC makes *all* INIT commands unavailable. DBRC sends an error message to the originating terminal when the command fails. You can still issue the failing command from other IMS online regions.

Note: DBRC does not make GENJCL commands unavailable, because the GENJCL.ARCHIVE command is needed for automatic archiving.

These commands can be issued to an IMSplex using the Batch SPOC utility.

DBRC does not remember command failures across IMS restarts because it assumes that you will correct the error before restarting IMS.

Examples

The following are examples of the /RMxxxxxx commands:

Examples for /RMCHANGE command

Entry ET (with comments):
```
/RMCHANGE DBRC='DB DBD(DIVNTZ04) SHARELVL(3)'. *COMMENT*.
```

Response ET:
```
CHANGE.DB DBD(DIVNTZ04) SHARELVL(3)
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMC COMMAND COMPLETED
```

Entry ET:
```
/RMCHANGE DBRC='DBDS DBD(DIVNTZ04) DDN(DBHVSAM1) ICON'.
```

Response ET:
```
CHANGE.DBDS DBD(DIVNTZ04) DDN(DIVNTZ04) ICON
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMC COMMAND COMPLETED
```

Entry ET:
```
/RMC DBRC='DBDS DBD(DEDBJN21) AREA(DB21AR7) VSO PREOPEN'.
```

Response ET:

```
DFS000I CHANGE.DBDS DBD(DEDBJN21) AREA(DB21AR7) VSO PREOPEN
DFS000I DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
```

Example for /RMDELETE command

Entry ET (with comments):

```
/RMDELETE DBRC='DB DBD(DIVNTZ04)'. *COMMENT*.
```

Response ET:

```
DELETE.DB DBD(DIVNTZ04)
DSP0203I COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMD COMMAND COMPLETED
```

Example for /RMGENJCL command

Entry ET (with comments):

```
/RMGENJCL LTERM SMITH DBRC='IC DBD(HDAMVSAM) DDN(DD1) LIST'.
```

Response ET:

```
DSP058I RMG COMMAND COMPLETED
```

Response LTERM SMITH:

```
GENJCL.IC DBD(HDAMVSAM) DDN(DD1)
//IC135607   JOB
//IC        EXEC PGM=DFSRRC00,PARM='ULU,DFSUDMP0',REGION=800K
//*
//*     THIS JCL ORIGINATES FROM THE USER'S 'JCLPDS' LIBRARY.
//*     %KEYWORDS ARE REPLACED BY THE GENJCL FUNCTION OF
//*     THE IMS DATABASE RECOVERY CONTROL FEATURE.
//*
//*                JCL FOR IMAGE COPY.
//*
//SYSPRINT  DD SYSOUT=A
//RECON1    DD DSN=POCONO1,DISP=SHR
//RECON2    DD DSN=POCONO2,DISP=SHR
//IMS       DD DSN=IMS.DBDLIB,DISP=SHR
//DD1    DD DSN=HDAMVASM,DCB=BUFNO=10,DISP=OLD
//DATAOUT1  DD DSN=IMS.HDAMVSAM.DD1.IC.ICDD1,UNIT=3400,
//            VOL=(PRIVATE,,,1,SER=(*****)),
//            LABEL=(1,SL),
//            DISP=(NEW,KEEP),DCB=BUFNO=10
//DFSVSAMP  DD *
1024,2
4096,4
//SYSIN     DD *
D1 HDAMVSAM DD1       DATAOUT1
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMG COMMAND COMPLETED
```

Example for /RMINIT command

Entry ET (with comments):

```
/RMINIT DBRC='DB DBD(DIVNTZ04) SHARELVL(3)'. *COMMENT*.
```

Response ET:

```
INIT.DB DBD(DIVNTZ04) SHARELVL(3)
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMI COMMAND COMPLETED
```

Examples for /RMLIST command

Entry ET (with comments):

```
/RMLIST DBRC='DB DBD(DIVNTZ04)'. *LAST COMMENT*.
```

Response ET for full-function databases:

```
        IMS VERSION 14 RELEASE 1 DATA BASE RECOVERY CONTROL        PAGE 0001
  LIST.DB DBD(DIVNTZ04)
14.336 13:19:01.337255                LISTING OF RECON             PAGE 0002
--------------------------------------------------------------------------------
 DB
  DBD=DIVNTZ04                                     DMB#=4      TYPE=IMS
  SHARE LEVEL=0              GSGNAME=**NULL**     USID=0000000001
  AUTHORIZED USID=0000000000   RECEIVE USID=0000000000 HARD USID=0000000000
  RECEIVE NEEDED USID=0000000000
  DBRCVGRP=**NULL**
  FLAGS:                              COUNTERS:
    BACKOUT NEEDED       =OFF           RECOVERY NEEDED COUNT   =0
    READ ONLY            =OFF           IMAGE COPY NEEDED COUNT =0
    PROHIBIT AUTHORIZATION=OFF          AUTHORIZED SUBSYSTEMS   =0
    RECOVERABLE          =YES           HELD AUTHORIZATION STATE=0
                                        EEQE COUNT              =0
    TRACKING SUSPENDED   =NO            RECEIVE REQUIRED COUNT  =0
    OFR REQUIRED         =NO
    REORG INTENT         =NO
    QUIESCE IN PROGRESS  =NO
    QUIESCE HELD         =NO
--------------------------------------------------------------------------------
DSP0180I  NUMBER OF RECORDS LISTED IS        1
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 14.336 13:19:01.891672
        IMS VERSION 14 RELEASE 1 DATA BASE RECOVERY CONTROL        PAGE 0003
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
```

Response ET for Fast Path databases:

```
        IMS VERSION 14 RELEASE 1 DATA BASE RECOVERY CONTROL        PAGE 0001
  LIST.DB DBD(DEDBJN22)
14.336 12:10:07.154448                LISTING OF RECON             PAGE 0002
--------------------------------------------------------------------------------
 DB
  DBD=DEDBJN22                                     DMB#=4      TYPE=FP
  SHARE LEVEL=3              RANDOMIZER=**NULL**
  FLAGS:                              COUNTERS:
                                       RECOVERY NEEDED COUNT   =0
                                       IMAGE COPY NEEDED COUNT =0
    PROHIBIT AUTHORIZATION=OFF          AUTHORIZED AREAS        =0
    RECOVERABLE          =YES           EEQE COUNT              =0
    FULLSEG DEFAULT      =NO
--------------------------------------------------------------------------------
DSP0180I  NUMBER OF RECORDS LISTED IS        1
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 14.336 12:10:07.484796
        IMS VERSION 14 RELEASE 1 DATA BASE RECOVERY CONTROL        PAGE 0003
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
```

Entry ET:

  /RMLIST DBRC='DB DBD(DIVNTZ02) DBDS'

Response ET for full-function databases:

```
          IMS VERSION 14 RELEASE 1 DATA BASE RECOVERY CONTROL         PAGE 0001
LIST.DB DBD(DIVNTZ02) DBDS
14.336 13:58:07.077014                 LISTING OF RECON                PAGE 0002
--------------------------------------------------------------------------------
 DB
 DBD=DIVNTZ02              IRLMID=*NULL        DMB#=1       TYPE=IMS
 SHARE LEVEL=3            GSGNAME=**NULL**     USID=0000000002
 AUTHORIZED USID=0000000002  RECEIVE USID=0000000002 HARD USID=0000000002
 RECEIVE NEEDED USID=0000000000
 DBRCVGRP=**NULL**
 FLAGS:                            COUNTERS:
   BACKOUT NEEDED       =OFF         RECOVERY NEEDED COUNT   =0
   READ ONLY            =OFF         IMAGE COPY NEEDED COUNT =0
   PROHIBIT AUTHORIZATION=OFF        AUTHORIZED SUBSYSTEMS   =0
   RECOVERABLE          =YES         HELD AUTHORIZATION STATE=0
                                     EEQE COUNT              =0
   TRACKING SUSPENDED   =NO          RECEIVE REQUIRED COUNT  =0
   OFR REQUIRED         =NO
   REORG INTENT         =NO
   QUIESCE IN PROGRESS  =NO
   QUIESCE HELD         =NO
--------------------------------------------------------------------------------
14.336 13:58:07.077014                 LISTING OF RECON                PAGE 0003
--------------------------------------------------------------------------------
 DBDS
 DSN=IMSTESTL.DIVNTZ02.FJXXS01K                           TYPE=IMS
 DBD=DIVNTZ02  DDN=DBHVSAM1 DSID=001 DBORG=HISAM  DSORG=VSAM
 CAGRP=**NULL**  GENMAX=2    IC AVAIL=0    IC USED=1     DSSN=00000001
 NOREUSE        RECOVPD=0
 DEFLTJCL=**NULL**  ICJCL=ICJCL     OICJCL=OICJCL    RECOVJCL=RECOVJCL
 RECVJCL=ICRCVJCL
 FLAGS:                            COUNTERS:
   IC NEEDED       =OFF
   RECOV NEEDED    =OFF
   RECEIVE NEEDED =OFF              EEQE COUNT              =0
--------------------------------------------------------------------------------

 IMAGE
 RUN     = 14.336 13:57:56.381657        *  RECORD COUNT =1
 STOP    = 00.000 00:00:00.000000           SMSOFFLC    USID=0000000002

 IC1
  DSN=IMSTESTL.DBFC1.DIVNTZ02.DBHVSAM1


14.336 13:58:07.077014                 LISTING OF RECON                PAGE 0004
--------------------------------------------------------------------------------
 DBDS
 DSN=IMSTESTL.DIVNTZ02.FJXXS01E                           TYPE=IMS
 DBD=DIVNTZ02  DDN=DBHVSAM2 DSID=002 DBORG=HISAM  DSORG=VSAM
 CAGRP=**NULL**  GENMAX=2    IC AVAIL=0    IC USED=1     DSSN=00000000
 NOREUSE        RECOVPD=0
 DEFLTJCL=**NULL**  ICJCL=ICJCL     OICJCL=OICJCL    RECOVJCL=RECOVJCL
 RECVJCL=ICRCVJCL
 FLAGS:                            COUNTERS:
   IC NEEDED       =OFF
   RECOV NEEDED    =OFF
   RECEIVE NEEDED =OFF              EEQE COUNT              =0
--------------------------------------------------------------------------------

 IMAGE
 RUN     = 14.336 13:57:56.381657        *  RECORD COUNT =1
```

```
     STOP    = 00.000 00:00:00.000000              SMSOFFLC    USID=0000000002

   IC1
    DSN=IMSTESTL.DB1.FC1.DIVNTZ02.DBHVSAM2


 DSP0180I   NUMBER OF RECORDS LISTED IS        5
 DSP0203I   COMMAND COMPLETED WITH CONDITION CODE 00
 DSP0220I   COMMAND COMPLETION TIME 14.336 13:58:07.329858
          IMS VERSION 14 RELEASE 1 DATA BASE RECOVERY CONTROL        PAGE 0005
 LIST.DB DBD(DIVNTZ02)
 14.336 13:58:07.077014                   LISTING OF RECON             PAGE 0006
 --------------------------------------------------------------------------------
  DB
   DBD=DIVNTZ02              IRLMID=*NULL        DMB#=1      TYPE=IMS
   SHARE LEVEL=3            GSGNAME=**NULL**     USID=0000000002
   AUTHORIZED USID=0000000002   RECEIVE USID=0000000002 HARD USID=0000000002
   RECEIVE NEEDED USID=0000000000
   DBRCVGRP=**NULL**
   FLAGS:                            COUNTERS:
    BACKOUT NEEDED       =OFF         RECOVERY NEEDED COUNT   =0
    READ ONLY            =OFF         IMAGE COPY NEEDED COUNT =0
    PROHIBIT AUTHORIZATION=OFF        AUTHORIZED SUBSYSTEMS   =0
    RECOVERABLE          =YES         HELD AUTHORIZATION STATE=0
                                      EEQE COUNT              =0
    TRACKING SUSPENDED   =NO          RECEIVE REQUIRED COUNT  =0
    OFR REQUIRED         =NO
    REORG INTENT         =NO
    QUIESCE IN PROGRESS  =NO
    QUIESCE HELD         =NO
 --------------------------------------------------------------------------------
 DSP0180I   NUMBER OF RECORDS LISTED IS        1
 DSP0203I   COMMAND COMPLETED WITH CONDITION CODE 00
 DSP0220I   COMMAND COMPLETION TIME 14.336 13:58:07.333337
```

Response ET for Fast Path databases:

```
          IMS VERSION 14 RELEASE 1 DATA BASE RECOVERY CONTROL        PAGE 0001
 LIST.DB DBD(DEDBJN22) DBDS
 14.336 13:58:07.077014                   LISTING OF RECON             PAGE 0002
 --------------------------------------------------------------------------------
  DB
   DBD=DEDBJN22                             DMB#=4      TYPE=FP
   SHARE LEVEL=3            RANDOMIZER=**NULL**
   FLAGS:                            COUNTERS:
                                      RECOVERY NEEDED COUNT   =0
                                      IMAGE COPY NEEDED COUNT =0
    PROHIBIT AUTHORIZATION=OFF        AUTHORIZED AREAS        =0
    RECOVERABLE          =YES         EEQE COUNT              =0
    FULLSEG DEFAULT      =NO          ALTER COUNT             =0
 --------------------------------------------------------------------------------
 14.336 13:58:07.077014                   LISTING OF RECON             PAGE 0003
 --------------------------------------------------------------------------------
  DBDS
   DBD=DEDBJN22  AREA=DB22AR0                          TYPE=FP
   SHARE LEVEL=3           DSID=00001 DBORG=DEDB   DSORG=VSAM
   GSGNAME=**NULL**            USID=0000000002
   AUTHORIZED USID=0000000002   RECEIVE USID=0000000002 HARD USID=0000000002
   RECEIVE NEEDED USID=0000000000
   CAGRP=**NULL**  GENMAX=2    IC AVAIL=0    IC USED=0    DSSN=00000001
   NOREUSE        RECOVPD=0       NOVSO  NOPREOPEN  NOPRELOAD  NOFULLSG
   CFSTR1=**NULL**         CFSTR2=**NULL**            NOLKASID  NOMAS
   DEFLTJCL=**NULL**  ICJCL=ICJCL    RECVJCL=ICRCVJCL  RECOVJCL=RECOVJCL
   DBRCVGRP=**NULL**
   FLAGS:                            COUNTERS:
    PROHIBIT AUTHORIZATION=OFF       AUTHORIZED SUBSYSTEMS   =0
                                     HELD AUTHORIZATION STATE=0
```

```
        IC NEEDED            =OFF       ADS AVAIL #            =1
        IC RECOMMENDED       =ON
        RECOV NEEDED         =OFF       REGISTERED ADS #      =1
                                        EEQE COUNT            =0
        RECEIVE NEEDED       =OFF       SHADOW ADS AVAIL #    =0
        OFR REQUIRED         =NO        REGISTERED SHADOW ADS # =0
        TRACKING SUSPENDED   =NO
        HSSP CIC IN PROGRESS =NO
        QUIESCE IN PROGRESS  =NO
        QUIESCE HELD         =NO
        ALTER IN PROGRESS    =NO

      ADS LIST:
                                                            CREATE
        -ADS DDN--ADS DSN-                              -STAT- -RUNNING-
        DB22AR0  IMSTESTL.DB22AR0                       AVAIL    NO
    --------------------------------------------------------------------------------

      ALLOC
       ALLOC  =14.336 13:56:04.484403      *  ALLOC LRID =0000000000000000
       DSSN=0000000001 USID=0000000002 START = 14.336 13:55:34.300003
       DEALLOC =14.336 13:57:37.105130        DEALLOC LRID =0000000000000000


      REORG
       RUN    = 14.336 13:54:39.460557      *    USID = 0000000001
       REORG# = 00000
    14.336 13:58:07.077014                 LISTING OF RECON              PAGE 0004
    --------------------------------------------------------------------------------
      DBDS
       DBD=DEDBJN22  AREA=DB22AR1                                        TYPE=FP
       SHARE LEVEL=3           DSID=00002 DBORG=DEDB   DSORG=VSAM
       GSGNAME=**NULL**           USID=0000000002
       AUTHORIZED USID=0000000002  RECEIVE USID=0000000002 HARD USID=0000000002
       RECEIVE NEEDED USID=0000000000
       CAGRP=**NULL**  GENMAX=2    IC AVAIL=0    IC USED=0    DSSN=00000001
       NOREUSE       RECOVPD=0     NOVSO  NOPREOPEN  NOPRELOAD  NOFULLSG
       CFSTR1=**NULL**        CFSTR2=**NULL**          NOLKASID  NOMAS
       DEFLTJCL=**NULL**  ICJCL=ICJCL    RECVJCL=ICRCVJCL  RECOVJCL=RECOVJCL
       DBRCVGRP=**NULL**
       FLAGS:                             COUNTERS:
         PROHIBIT AUTHORIZATION=OFF         AUTHORIZED SUBSYSTEMS   =0
                                            HELD AUTHORIZATION STATE=0
         IC NEEDED            =OFF          ADS AVAIL #            =1
         IC RECOMMENDED       =ON
         RECOV NEEDED         =OFF          REGISTERED ADS #      =1
                                            EEQE COUNT            =0
         RECEIVE NEEDED       =OFF          SHADOW ADS AVAIL #    =0
         OFR REQUIRED         =NO           REGISTERED SHADOW ADS # =0
         TRACKING SUSPENDED   =NO
         HSSP CIC IN PROGRESS =NO
         QUIESCE IN PROGRESS  =NO
         QUIESCE HELD         =NO
         ALTER IN PROGRESS    =NO

      ADS LIST:
                                                            CREATE
        -ADS DDN--ADS DSN-                              -STAT- -RUNNING-
        DB22AR1  IMSTESTL.DB22AR1                       AVAIL    NO
    --------------------------------------------------------------------------------

      ALLOC
       ALLOC  =14.336 13:56:04.561039      *  ALLOC LRID =0000000000000000
       DSSN=0000000001 USID=0000000002 START = 14.336 13:55:34.300003
       DEALLOC =14.336 13:57:37.105130        DEALLOC LRID =0000000000000000
```

```
 REORG
 RUN    = 14.336 13:54:39.633913        *    USID = 0000000001
 REORG# = 00000
14.336 13:58:07.077014                LISTING OF RECON                 PAGE 0005
--------------------------------------------------------------------------------
 DBDS
 DBD=DEDBJN22  AREA=DB22AR2                                     TYPE=FP
 SHARE LEVEL=3              DSID=00003 DBORG=DEDB   DSORG=VSAM
 GSGNAME=**NULL**             USID=0000000001
 AUTHORIZED USID=0000000000   RECEIVE USID=0000000000 HARD USID=0000000000
 RECEIVE NEEDED USID=0000000000
 CAGRP=**NULL** GENMAX=2    IC AVAIL=0     IC USED=0     DSSN=00000000
 NOREUSE         RECOVPD=0      NOVSO  NOPREOPEN  NOPRELOAD  NOFULLSG
 CFSTR1=**NULL**         CFSTR2=**NULL**             NOLKASID  NOMAS
 DEFLTJCL=**NULL**  ICJCL=ICJCL    RECVJCL=ICRCVJCL  RECOVJCL=RECOVJCL
 DBRCVGRP=**NULL**
 FLAGS:                          COUNTERS:
   PROHIBIT AUTHORIZATION=OFF        AUTHORIZED SUBSYSTEMS   =0
                                     HELD AUTHORIZATION STATE=0
   IC NEEDED            =OFF         ADS AVAIL #            =1
   IC RECOMMENDED       =ON
   RECOV NEEDED         =OFF         REGISTERED ADS #        =1
                                     EEQE COUNT              =0
   RECEIVE NEEDED       =OFF         SHADOW ADS AVAIL #      =0
   OFR REQUIRED         =NO          REGISTERED SHADOW ADS # =0
   TRACKING SUSPENDED   =NO
   HSSP CIC IN PROGRESS =NO
   QUIESCE IN PROGRESS  =NO
   QUIESCE HELD         =NO
   ALTER IN PROGRESS    =NO

 ADS LIST:
                                                          CREATE
   -ADS DDN--ADS DSN-                               -STAT- -RUNNING-
   DB22AR2  IMSTESTL.DB22AR2                        AVAIL    NO
--------------------------------------------------------------------------------

 REORG
 RUN    = 14.336 13:54:39.782464        *    USID = 0000000001
 REORG# = 00000
```

Entry ET (With comments):

```
 /RML DBRC='DBDS DBD(DEDBJN21) AREA(DB21AR0)'. *VSO AREA*.
```

Response ET:

```
 LIST.DBDS DBD(DEDBJN21) AREA(DB21AR0)
--------------------------------------------------------------------------------
 DBDS
 DBD=DEDBJN21  AREA=DB21AR0                                     TYPE=FP
 SHARE LEVEL=1              DSID=00001 DBORG=DEDB   DSORG=VSAM
 GSGNAME=**NULL**             USID=0000000002
 AUTHORIZED USID=0000000002   RECEIVE USID=0000000002 HARD USID=0000000002
 RECEIVE NEEDED USID=0000000000
 CAGRP=**NULL** GENMAX=2    IC AVAIL=0     IC USED=0     DSSN=00000001
 NOREUSE         RECOVPD=0      NOVSO  NOPREOPEN  NOPRELOAD  NOFULLSG
 CFSTR1=**NULL**         CFSTR2=**NULL**             NOLKASID  NOMAS
 DEFLTJCL=**NULL**  ICJCL=ICJCL    RECVJCL=ICRCVJCL  RECOVJCL=RECOVJCL
 DBRCVGRP=**NULL**
 FLAGS:                          COUNTERS:
   PROHIBIT AUTHORIZATION=OFF        AUTHORIZED SUBSYSTEMS   =1
                                     HELD AUTHORIZATION STATE=7
   IC NEEDED            =OFF         ADS AVAIL #            =1
   IC RECOMMENDED       =ON
   RECOV NEEDED         =OFF         REGISTERED ADS #        =1
                                     EEQE COUNT              =0
   RECEIVE NEEDED       =OFF         SHADOW ADS AVAIL #      =0
```

```
              OFR REQUIRED          =NO          REGISTERED SHADOW ADS # =0
              TRACKING SUSPENDED    =NO
              HSSP CIC IN PROGRESS  =NO
              QUIESCE IN PROGRESS   =NO
              QUIESCE HELD          =NO
              ALTER IN PROGRESS     =NO

        ADS LIST:
                                                              CREATE
          -ADS DDN--ADS DSN-                          -STAT- -RUNNING-
          DB21AR0   DB21AR0                           AVAIL    NO

        ASSOCIATED SUBSYSTEM INFORMATION:
                              ENCODED
          -SSID-   -ACCESS INTENT-  -STATE-  -SS ROLE-
          SYS3       EXCLUSIVE        7      ACTIVE
    ------------------------------------------------------------------------------
      ALLOC
       ALLOC  =14.335 14:10:21.444644      *  ALLOC LRID =0000000000000000
       DSSN=0000000001 USID=0000000002 START = 14.335 14:03:14.819522


      REORG
       RUN     = 14.335 13:59:41.090369       *    USID = 0000000001
    DSP0180I  NUMBER OF RECORDS LISTED IS     3
    DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
    DSP0220I  COMMAND COMPLETION TIME 14.335 14:15:56.641143
    DSP0211I  COMMAND PROCESSING COMPLETE
    DSP0211I  HIGHEST CONDITION CODE = 00
```

Entry ET (With comments):

```
  /RML DBRC='DBDS DBD(DEDBJN21) AREA(DB21AR6)'. *DEDB AREA*.
```

Response ET:

```
 LIST.DBDS DBD(DEDBJN21) AREA(DB21AR6)
 ------------------------------------------------------------------------------
 DBDS
  DBD=DEDBJN21  AREA=DB21AR6                                   TYPE=FP
  SHARE LEVEL=1           DSID=00006 DBORG=DEDB   DSORG=VSAM
  GSGNAME=**NULL**            USID=0000000002
  AUTHORIZED USID=0000000002  RECEIVE USID=0000000002 HARD USID=0000000002
  RECEIVE NEEDED USID=0000000000
  CAGRP=**NULL**  GENMAX=2    IC AVAIL=0     IC USED=0     DSSN=00000001
  NOREUSE         RECOVPD=0     NOVSO  NOPREOPEN  NOPRELOAD  NOFULLSG
  CFSTR1=**NULL**         CFSTR2=**NULL**            NOLKASID  NOMAS
  DEFLTJCL=**NULL**  ICJCL=ICJCL    RECVJCL=ICRCVJCL  RECOVJCL=RECOVJCL
  DBRCVGRP=**NULL**
  FLAGS:                            COUNTERS:
    PROHIBIT AUTHORIZATION=OFF        AUTHORIZED SUBSYSTEMS  =0
                                      HELD AUTHORIZATION STATE=0
    IC NEEDED           =OFF          ADS AVAIL #           =0
    RECOV NEEDED        =ON           REGISTERED ADS #      =1
                                      EEQE COUNT            =0
    OFR REQUIRED        =NO
    TRACKING SUSPENDED  =NO

  ADS LIST:
                                                           CREATE
      -ADS DDN--ADS DSN-                            -STAT- -RUNNING-
      DB21AR6   DB21AR6                             UNAVAIL   NO

  ASSOCIATED SUBSYSTEM INFORMATION:

  ALLOC
   ALLOC  =14.335 14:10:22.530986      *  ALLOC LRID =0000000000000000
```

```
      DSSN=0000000001 USID=0000000002 START = 14.335 14:03:14.819522


 REORG
  RUN     = 14.335 13:59:42.372727          *    USID = 0000000001
  REORG#  = 00000
DSP0180I  NUMBER OF RECORDS LISTED IS      4
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 14.335 14:15:56.641143
DSP0211I  COMMAND PROCESSING COMPLETE
DSP0211I  HIGHEST CONDITION CODE = 00
```

Example for /RMNOTIFY command

Entry ET (with comments):

```
/RMNOTIFY DBRC='SUBSYS SSID(IMSB) IRLMID(IRLM1) NORMAL'.
```

Response ET:

```
NOTIFY.SUBSYS SSID(IMSB) IRLMID(IRLM1) NORMAL
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME
DSP0211I  COMMAND PROCESSING COMPLETE.
DSP0211I  HIGHEST CONDITION CODE = 00
DSP0058I  RMN COMMAND COMPLETED
```

Related reference:

➡ CHANGE commands (Commands)

➡ DELETE commands (Commands)

➡ GENJCL commands (Commands)

➡ INIT commands (Commands)

➡ LIST commands (Commands)

➡ NOTIFY commands (Commands)

Chapter 16. /RSTART command

The /RSTART command starts lines, lines and physical terminals, logical links, nodes, and users when you do not want to reset all associated conditions such as a conversation or special operating mode.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 796
- "Usage notes" on page 798
- "Equivalent IMS type-2 commands" on page 798
- "Examples" on page 798

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 257. Valid environments for the /RSTART command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RSTART	X		X
CONTINUOUS	X		X
LINE	X		X
LINK	X		X
LOPEN	X		X
MODE	X		X
MSPLINK	X		X
NODE	X		X
PTERM	X		X
USER	X		X

Syntax

```
►►─┬─/RSTART─┬─┬─LINE─┬─┬─line#─┬────────┬─LOPEN─┬──────────────────►◄
   └─/RST────┘ │      │ └─ALL───┘        └───────┘
              │      │
              │      └─line#─PTERM─┬─pterm#─┬──────┐
              │                    └─ALL────┘
              │
              ├─LINK─┬─link#─┬──────┬─MODE─modename─┐
              │      └─ALL───┘
              │
              ├─MSPLINK─┬─msplinkname─┬──────────────┐
              │         └─ALL─────────┘
              │
              ├─NODE─┬─┬─nodename──┬──────────────────────┐
              │      │ └─nodename*─┘
              │      ├─nodename─USER─┬─username─┬─────┐
              │      │               └─ALL──────┘
              │      └─ALL────────────────────────┘
              │
              └─USER─┬─┬─username──┬──────────────────┐
                     │ └─username*─┘
                     └─ALL──────────┘
```

Keywords

The following keywords are valid for the /RSTART command:

LINE
Specifies the line or line/physical terminal to be started with all terminals on the line in the same mode of operation they were in when they were stopped. The /RSTART LINE command allows all input, output, and queuing to start on the line and take the line out of response mode if the line was in this mode. The /RSTART LINE PTERM command does not reset line response mode, but does reset the non-VTAM-attached 3270 terminal response mode and looptest mode.

LOPEN
LOPEN enables stopped and idle remote non-VTAM lines. Enter the /RSTART LINE LOPEN command before any /RSTART LINE PTERM command to avoid having a line that is stopped and idle reset before it can be enabled again. If the line is not stopped or process stopped, and idle, or if enabling is not applicable, the LOPEN keyword is ignored and processing continues.

/RSTART LINE and /RSTART NODE cannot reset terminal response mode if Fast Path is active for a specified physical terminal or node. The /DEQUEUE command must be entered to discard Fast Path output before using the /RSTART command.

LINK
Specifies the logical links to be started, either individually or all at once.

For TCP/IP and VTAM links, the /RSTART LINK command can be issued in either one of the two partner systems to start communication. For CTC or MTM links, communication does not begin until the /RSTART LINK command is entered in both partner IMS systems.The /RSTART LINK command is rejected unless the link is in a stopped and idled status and the assigned physical link is open, as shown in the /DISPLAY command.

MODE

The MODE keyword allows you to specify mode table entries to be used when activating an IMS VTAM MSC session. Use of the MODE keyword with non-VTAM links is invalid. If non-VTAM links are referred to specifically in the /RSTART LINK command with the MODE keyword, they will be marked in error.

MSPLINK

Resets the PSTOPPED status of MSC TCP/IP or VTAM links to enable logons. If the TCP/IP link is used with TCP/IP generic resources, this keyword also resets the PSTOPGEN status.

NODE

Specifies the VTAM node for which input, output, and queuing will start. The /RSTART NODE nodename USER username command restarts the ISC half-session allocated to *username* for *nodename*. The USER keyword is valid only with the NODE keyword and restarts the specified half-session. If the USER keyword is omitted, all half-sessions of the specified node are affected.

Restrictions for using NODE and USER parameters together:
- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- The /RSTART NODE USER commands are valid for ISC, LUP, and 3600 nodes only.

The NODE parameter can be generic if the USER keyword is not specified. The generic parameter specifies nodes that already exist. If the node was created temporarily to retain status data and the status conditions have been reset, then the node is deleted at the next simple checkpoint.

If global resource information is not kept in Resource Manager, the /RSTART NODE command allows a node to logon to the local IMS, without resetting local status. If global resource information is kept in Resource Manager, the /RSTART NODE command allows a node to logon to any IMS in the IMSplex, without resetting global node status kept in Resource Manager. If the node no longer has significant status, it is deleted by Resource Manager.

USER

Specifies the USER for which input, output, and queuing are to start. This command starts the USER without resetting conditions such as conversation mode, exclusive mode, and test mode. The /RSTART USER command applies only to dynamic users.

The USER parameter can be generic where the generic parameter specifies already existing users.

If the user structure is temporary and was created solely to retain status that is now reset, the temporary user is deleted at the next simple checkpoint.

If global resource information is not kept in Resource Manager, the /RSTART USER command allows a user to sign on to the local IMS. If global resource information is kept in Resource Manager, the /RSTART USER command allows a user to sign on to any IMS in the IMSplex. If the user no longer has significant status in Resource Manager, it is deleted.

Usage notes

The /RSTART command checks the validity of all parameters entered by the terminal operator. If an error is detected on parameters that are independent of one another, only the invalid parameters are flagged as being in error and the /RSTART command processes the rest of the parameters.

The /RSTART command can be used to reset conditions previously established by the /START, /STOP, /PSTOP, /PURGE, /MONITOR, /COMPT, or /RCOMPT command.

This command can be issued to an IMSplex using the Batch SPOC utility.

Equivalent IMS type-2 commands

The following table shows variations of the /RSTART command and the IMS type-2 commands that perform similar functions.

Table 258. Type-2 equivalents for the /RSTART command

Task	/RSTART command	Similar IMS type-2 command
Resets MSC TCP/IP or VTAM links to enable logons.	/RSTART MSPLINK *msplinkname* | ALL	UPDATE MSPLINK NAME(*msplinkname* | *) START(LOGON)

Examples

The following are examples of the /RSTART command:

Example 1 for /RSTART command

Entry ET:
```
/RSTART LINE 4,5,6,7,8,9,10,11
```

Response ET:
```
DFS058I  RSTART COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL RSTARTED
```

Explanation: LINES 4,5,6,7,8,9,10, and 11 are started.

Example 2 for /RSTART command

Entry ET:
```
/RSTART LINE 4 5 6 700
```

Response ET:
```
DFS058I  RSTART COMMAND COMPLETED EXCEPT LINE 700
```

Example 3 for /RSTART command

Entry ET:
```
/RSTART LINE 4 PTERM 1, 2
```

Response ET:
```
DFS058I  RSTART COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL RSTARTED
```

Explanation: LINE 4 PTERM 1 and 2 are started.

Example 4 for /RSTART command

Entry ET:
```
/RSTART LINK ALL
```

Response ET:
```
DFS058I  RSTART COMMAND COMPLETED
```

Explanation: All of the logical links are started. For CTC and MTM links, communication across the link does not begin until the partner in the remote system is started with the /RSTART LINK command.

Response ET:
```
DFS2168I  CONNECTION ESTABLISHED ON LINK 2
```

Explanation: The connection for communication between two IMS systems is established. The partner link is started with a /RSTART LINK command. After each connection, the message DFS2168 is returned.

Example 5 for /RSTART command

Entry ET:
```
/RSTART LINK 2
```

Response ET:
```
DFS058I  RSTART COMMAND COMPLETED
```

Explanation: Logical link 2 is started.

Response ET:
```
DFS2168I  CONNECTION ESTABLISHED ON LINK 2
```

Explanation: The two IMS systems are connected.

Example 6 for /RSTART command and a TCP/IP link

Entry ET:
```
/RSTART LINK 23
```

Response ET:
```
DFS2168I 17:37:39 CONNECTION ESTABLISHED ON LINK 0023
```

Response ET at the partner system:

```
DFS2160I 17:37:39 LINK 028 STARTED BY PARTNER TB NODE IMS1
DFS2168I 17:37:39 CONNECTION ESTABLISHED ON LINK 0028
```

Explanation: In the Response ET at the partner system, IMS1 is the IMS ID of the partner system in which the /RSTART command was issued.

Example 7 for /RSTART command and a VTAM link

Entry ET:

```
/RSTART LINK 10
```

Response ET:

```
DFS058I 17:34:36 RSTART COMMAND COMPLETED
DFS2168I 17:34:36 CONNECTION ESTABLISHED ON LINK 0010
```

Response ET at the partner system:

```
DFS2160I 17:34:36 LINK 013 STARTED BY PARTNER AK NODE L6APPL3
DFS2168I 17:34:36 CONNECTION ESTABLISHED ON LINK 0013
```

Explanation: In the Response ET at the partner system, L6APPL3 is the APPLID of the partner system in which the /RSTART command was issued.

Example 8 for /RSTART command

Entry ET:

```
/RSTART NODE EAST
```

Response ET:

```
DFS058I  RSTART COMMAND COMPLETED
```

Explanation: The node named EAST is started.

Chapter 17. /RTAKEOVER command

The /RTAKEOVER command requests a remote takeover of an active IMS subsystem by a tracking subsystem at a secondary site.

The remote takeover causes the tracking subsystem to finish processing and shut down.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 803

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 259. Valid environments for the /RTAKEOVER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/RTAKEOVER	X	X	X
DUMPQ	X	X	X
FREEZE	X	X	X
NOREVERSE	X	X	X
UNPLAN	X	X	X

Syntax

/RTAKEOVER for an active subsystem

```
►►──┬─/RTAKEOVER─┬──┬─FREEZE─┬──────────────────────────────────►◄
    └─/RTA───────┘  └─DUMPQ──┘
```

/RTAKEOVER for a tracking subsystem

```
►►──┬─/RTAKEOVER─┬──UNPLAN──┬──────────────┬─────────────────────►◄
    └─/RTA───────┘          └─NOREVERSE──┘
```
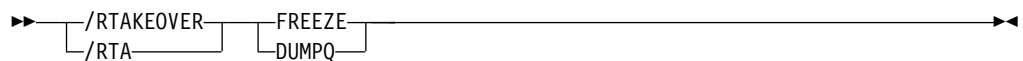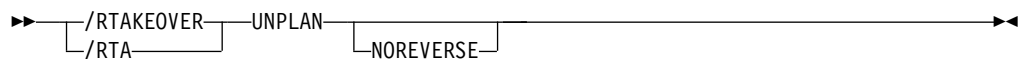
Usage notes

If entered on an active subsystem, /RTAKEOVER also causes the active subsystem to shut down. Once the remote tracking subsystem has shut down and remote takeover has successfully completed, one or more subsystems may be brought up in an active role at the secondary site and started with standard IMS restart commands.

Unless you specify the NOREVERSE keyword, a remote takeover can be reversed, even after the /RTAKEOVER command has been issued. The NORTA parameter on the CHANGE.SG command can be used to reverse a remote takeover.

This command can be issued to an IMSplex using the Batch SPOC utility.

/RTAKEOVER for an active subsystem

/RTAKEOVER is issued on the active IMS subsystem to initiate a planned remote takeover and must be specified with either the FREEZE keyword or the DUMPQ keyword.

/RTAKEOVER FREEZE indicates that a /CHECKPOINT FREEZE type of shutdown is performed before the planned takeover occurs.

/RTAKEOVER DUMPQ indicates that a /CHECKPOINT DUMPQ type of shutdown is performed before the planned takeover occurs. This form of takeover provides that all relevant log records reach the tracking subsystem such that no data is lost. This form of takeover allows the capability to rebuild the message queues during the new active start.

/RTAKEOVER must be entered for at least one IMS subsystem in the global service group (GSG) at the active site for which takeover is to occur. The other IMS subsystems at the active site must also be shut down, either with the /RTAKEOVER command, or some form of the /CHECKPOINT command that shuts the system down.

Once the active sends all of its log data sets, the active IMS subsystem shuts down. When all of the active subsystems in the global service group have shut down, the tracking subsystem then completes tracking work, stops online forward recovery (OFR), changes the role of the service group (tracking to active) in the RECON data set, and shuts down.

/RTAKEOVER for a tracking subsystem

/RTAKEOVER UNPLAN is issued on the tracking IMS subsystem to initiate an unplanned remote takeover after the active site fails unexpectedly. /RTAKEOVER UNPLAN causes the tracking subsystem to complete tracking work, stops online forward recovery (OFR), changes the role of the service group (tracking to active) in the RECON data set, and shuts down.

Recommendation: Specify the NOREVERSE keyword to enable the tracking IMS subsystem to save and process all data it has received from the active site, regardless of whether that data was committed on the active IMS subsystem.

Although you can still reverse a remote takeover even if you specify NOREVERSE, you should not reverse it; in this case, you receive message DFS4122A when you restart the tracking subsystem.

If you do not specify NOREVERSE, the tracking IMS subsystem discards any uncommitted data it has received from the active subsystem.

Examples

The following are examples of the /RTAKEOVER command:

Example for /RTAKEOVER command at active site

Entry ET:
```
/RTA FREEZE
```

Response ET:
```
DFS2939I REMOTE SITE PLANNED TAKEOVER IN PROGRESS SYS3
DFS2719I MSDB CHECKPOINT WRITTEN TO MSDBCP2  SYS3 DFS994I
        *CHKPT 94308/160026**FREEZE*  SYS3
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBB  FORMATA  MODSTAT ID:  2 SYS3
DFS3804I LATEST RESTART CHKPT: 94308/160026, LATEST BUILDQ CHKPT: 94308/154950 SYS3
DFS4036I CONVERSATION ENDING WITH SERVICE GROUP STLSITE2 SYS3
DFS3257I ONLINE LOG CLOSED ON DFSOLP01 SYS3
DFS2484I JOBNAME=JT160031 GENERATED BY LOG AUTOMATIC ARCHIVING SYS3
DFS092I  IMS LOG TERMINATED    SYS3
DFS4036I CONVERSATION ENDED  WITH SERVICE GROUP STLSITE2 SYS3
DFS4024I  STOP SERVGRP PROCESSING (INTERNAL) COMPLETE SYS3
DFS2091I IMS TIMER SERVICE SHUTDOWN COMPLETE SYS3
DFS0617I RDS BUFFERS HAVE BEEN SUCCESSFULLY PURGED SYS3
```

Response received at the Tracking system:
```
DFS2932I DATABASE UPDATES PRIOR TO SYSTEM SYS3 TAKEOVER HAVE BEEN
 ROUTED SYS3
```

Example for /RTAKEOVER DUMPQ at active site

Entry ET:
```
/RTA DUMPQ
```

Response ET:
```
DFS2939I REMOTE SITE PLANNED TAKEOVER IN PROGRESS SYS3
DFS2719I MSDB CHECKPOINT WRITTEN TO MSDBCP2  SYS3
DFS994I  *CHKPT 94308/165340**DUMPQ**  SYS3
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBB  FORMATA  MODSTAT ID:  2 SYS3
DFS3804I LATEST RESTART CHKPT: 94308/165340, LATEST BUILDQ CHKPT: 94308/165340 SYS3
DFS4036I CONVERSATION ENDING WITH SERVICE GROUP STLSITE2 SYS3
DFS3257I ONLINE LOG CLOSED ON DFSOLP01 SYS3
DFS2484I JOBNAME=JT165345 GENERATED BY LOG AUTOMATIC ARCHIVING SYS3
DFS092I  IMS LOG TERMINATED    SYS3
DFS4036I CONVERSATION ENDED  WITH SERVICE GROUP STLSITE2 SYS3
DFS4024I  STOP SERVGRP PROCESSING (INTERNAL) COMPLETE SYS3
DFS2091I IMS TIMER SERVICE SHUTDOWN COMPLETE SYS3
DFS0617I RDS BUFFERS HAVE BEEN SUCCESSFULLY PURGED SYS3
```

Response received at the Tracking system:
```
DFS2932I DATABASE UPDATES PRIOR TO SYSTEM SYS3 TAKEOVER HAVE BEEN
ROUTED SYS3
```

Example for /RTAKEOVER UNPLAN at remote site

Entry ET:
```
/RTA UNPLAN
```

Response ET:

```
DFS4123I UNPLANNED TAKEOVER IN PROGRESS
DFS2913I CONVERSATION WITH IMS SYS3 TERMINATED: TRK SYS SHUTDOWN
DFS2913I CONVERSATION WITH IMS IMS2 TERMINATED: TRK SYS SHUTDOWN
DFS2500I DATASET  IMZ00007 SUCCESSFULLY DEALLOCATED
DFS2500I DATASET  IMZ00015 SUCCESSFULLY DEALLOCATED
DFS2934I LOG TRUNCATION STARTED FOR IMS: IMS2
DFS2934I LOG TRUNCATION STARTED FOR IMS: SYS3
DFS2943I THERE ARE NO DATA SETS TO TRUNCATE FOR SYS3
DFS2908I DATABASE UPDATES COMPLETE FOR REMOTE TAKEOVER PROCESSING
DFS2500I DATASET  IMZ00020 SUCCESSFULLY CREATED
DFS2500I DATASET  IMZ00021 SUCCESSFULLY ALLOCATED
DFS2500I DATASET  IMZ00021 SUCCESSFULLY DEALLOCATED
DFS2935I TRACKING LOG DATA SETS HAVE BEEN TRUNCATED AT 000000000000209C:
DFS2936I IMSTESTL.RSR.SLDS1.N0000012
DFS2500I DATASET  IMZ00016 SUCCESSFULLY DEALLOCATED
DFS2500I DATASET  IMZ00020 SUCCESSFULLY DELETED
DFS4126I TAKEOVER COMPLETE
DFS994I *CHKPT 94310/160240**FREEZE*
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:    3
DFS3804I LATEST RESTART CHKPT: 94310/160240, LATEST BUILDQ CHKPT: 94310/155301
DFS3257I ONLINE LOG CLOSED ON DFSOLP01
DFS2484I JOBNAME=JT160245 GENERATED BY LOG AUTOMATIC ARCHIVING
DFS092I IMS LOG TERMINATED
DFS2091I IMS TIMER SERVICE SHUTDOWN COMPLETE
DFS0617I RDS BUFFERS HAVE BEEN SUCCESSFULLY PURGED
```

Explanation: An unplanned takeover is successfully initiated for a tracking subsystem that was tracking 2 active subsystems (SYS3 and IMS2).

Chapter 18. /SECURE command

The /SECURE command is used to control the RACF security level. It is used for administrative control of the IMS environment and as an emergency operations control command to throttle RACF activity without requiring an IMS shutdown.

This command can be issued to an IMSplex using the Batch SPOC utility.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 806
- "Examples" on page 808

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 260. Valid environments for the /SECURE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/SECURE	X		X
APPC	X		X
OTMA	X		X

Note: You can use the SECURE OTMA command in non-OTMA environments to specify the security settings for the DFSYICAL member. The DFSYICAL member is an internal member that is used to process DL/I ICAL calls for synchronous program switch, even when OTMA is not enabled. If the DFSYICAL member does not exist when the SECURE OTMA command is issued, it is created automatically and the specified security configuration is applied to it.

Syntax

```
►►──┬─/SECURE─┬──APPC──┬─CHECK───┬──────────────────────────────►
    └─/SEC────┘        ├─FULL────┤
                       ├─NONE────┤
                       └─PROFILE─┘
```

```
                  ►►──┬─OTMA─┬─ACEEAGE─aging_value──────────────────────────────────┬──►◄
                      │      │                      └─TMEMBER─tmembername─┘          │
                      │      ├─CHECK───────────────────────────────────────────────┤
                      │      │        └─TMEMBER─tmembername─┘                        │
                      │      ├─FULL────────────────────────────────────────────────┤
                      │      │       └─TMEMBER─tmembername─┘                         │
                      │      ├─JOIN────────────────────────────────────────────────┤
                      │      │       └─TMEMBER─tmembername─┘                         │
                      │      ├─NONE────────────────────────────────────────────────┤
                      │      │       └─TMEMBER─tmembername─┘                         │
                      │      ├─PROFILE─────────────────────────────────────────────┤
                      │      │         └─TMEMBER─tmembername─┘                       │
                      │      └─REFRESH──┬───────────────────────┬──────────────────┘
                      │                 ├─TMEMBER─tmembername─┤
                      │                 └─USER─userid─────────┘
```
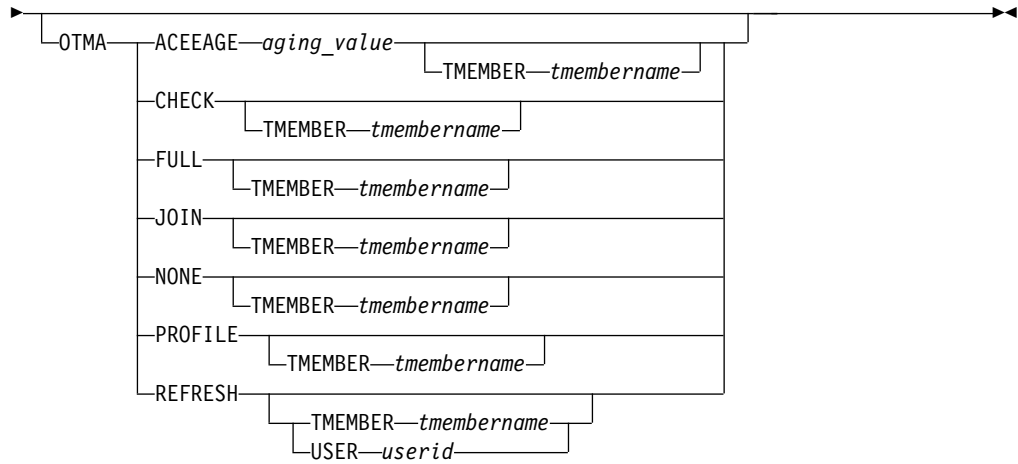
Keywords

The following keywords are valid for the /SECURE command:

ACEEAGE *aging_value* TMEMBER *tmembername*

Specifies an aging value for OTMA accessor environment elements (ACEEs), where *aging_value* is a 1- to 5-digit integer in the range 0 - 86,400® seconds. 86,400 seconds is equivalent to 1 day.

If you specify a value in the range 1 - 300, OTMA uses a value of 300 seconds because OTMA requires a value of at least 300 to enable ACEE refreshes. If you specify a 5-digit integer that is greater than 86,400, the value is automatically reset to 86,400. If you specify an integer that is a 6 digits or greater than 6 digits, the value is rejected and an error message is issued.

If you specify 0 for the **ACEEAGE** parameter and you do not use the **TMEMBER** parameter to specify an OTMA client, ACEE caching is disabled by OTMA even if OTMA security is set to FULL or CHECK. That is, new ACEEs are not cached by OTMA and ACEEs that are already cached are expired and removed from online memory in subsequent ACEE refreshes.

If you specify 0 for the **ACEEAGE** parameter and you use the **TMEMBER** parameter to specify an OTMA client, ACEE caching is disabled by OTMA for the specified OTMA client. That is, new ACEEs for the OTMA client are not cached by OTMA and ACEEs that are already cached for the client are expired and removed from online memory in subsequent ACEE refreshes.

If you specify a value in the range 1 - 86,400 for the **ACEEAGE** parameter and you do not use the **TMEMBER** parameter to specify an OTMA client, the aging value applies globally to all OTMA clients and overrides all other aging values that are passed by OTMA clients.

If you specify a value in the range 1 - 86,400 for the **ACEEAGE** parameter and you specify an OTMA client by using the **TMEMBER** parameter, the aging value applies to the specified OTMA client. If an aging value is passed by the OTMA client at client-bid time, the value is overridden by the value that you specify by using the **ACEEAGE** parameter. If you previously issued the /SECURE OTMA **ACEEAGE** 0 command, the /SECURE OTMA **ACEEAGE** *aging_value* **TMEMBER** *tmembername* command is rejected. For the /SECURE OTMA **ACEEAGE** *aging_value* **TMEMBER** *tmembername* command to be valid if the /SECURE OTMA**ACEEAGE** 0 command was previously issued, first reissue the /SECURE

| OTMA **ACEEAGE** command and specify a value in the range 1 - 86,400, and then
| reissue the /SECURE OTMA **ACEEAGE** *aging_value* **TMEMBER** *tmembername*
| command.

APPC

When used with the CHECK, FULL, NONE, or PROFILE parameters. APPC
controls the RACF security level for input from LU 6.2 devices. The /DISPLAY
APPC command can be used to show the security level that is currently in
effect. At IMS startup, the security default is FULL.

CHECK

Causes existing RACF calls to be made. IMS commands are checked using
the RACF resource class of CIMS. IMS transactions are checked using
TIMS. Disables z/OS System Authorization Facility security for IMS
allocate PSBs (APSBs).

FULL

Causes the same processing as the CHECK parameter but uses additional
RACF calls to create the security environment for dependent regions and
enables z/OS System Authorization Facility security for IMS APSBs for all
CPI Communications driven application programs.

NONE

Does not call RACF within IMS for security verification. RACF security
verification in APPC/MVS is not affected. Disables z/OS System
Authorization Facility security for IMS APSBs.

PROFILE

Causes the values in the TP profile for each transaction to be used. If the
TP profile is not defined for a transaction, or if the TP profile does not
specify a RACF security level, then the default security is CHECK.

OTMA

Is used with the CHECK, FULL, JOIN, NONE, or PROFILE parameters to
control the RACF security level for input from IMS Open Transaction Manager
Access (OTMA) clients. The /DISPLAY OTMA command can be used to show
the security level that is currently in effect. After an IMS cold start, the security
default is FULL if the IMS startup parameter OTMASE= is not used.

CHECK TMEMBER *tmembername*

Causes existing RACF calls to be made for input from the specified OTMA
client.

FULL TMEMBER *tmembername*

Causes the same processing as the CHECK parameter for input from the
specified OTMA client, but uses additional RACF calls to create the
security environment for dependent regions.

| **JOIN TMEMBER** *tmembername*
| Causes existing RACF calls to be made on only the OTMA client bid
| requests from OTMA clients. No transaction or command security checking
| is performed on individual messages.

NONE TMEMBER *tmembername*

Specifies that there is no RACF security checking within IMS for the input
from the specified OTMA client.

PROFILE TMEMBER *tmembername*

Specifies that the values in the Security Data section of the OTMA message
prefix of each transaction are used to check security for input from the
specified OTMA client.

REFRESH

  OTMA caches the ACEE for a user ID to reduce the amount of RACF I/O. As a result, a refresh for the cached ACEE is needed after the RACF database is updated. Issuing the /SEC OTMA REFRESH command without the TMEMBER option performs the ACEE refresh for all user IDs for all the OTMA clients. However, the actual ACEE refresh occurs when the next OTMA message for the user ID is received. This is designed to prevent all the RACF ACEE refreshes from happening at one time.

  When USER is specified, OTMA refreshes across all TMEMBERs only ACEEs that include the specified user profile.

USER *userid*

  An option to refresh only the specified user ID for all OTMA TMEMBERs. *userid* is the 1-8 character name of RACF User Profile to be refreshed.

Examples

The following are examples of the /SECURE command:

Example 1 for /SECURE command

Entry ET:

```
  /DIS APPC
```

Response ET:

```
IMSLU     #APPC-CONV SECURITY STATUS   DESIRED
IMSLUNME          0 PROFILE  ENABLED  ENABLED
*91242/163820*
```

Explanation: Enter /DISPLAY APPC to see which security checking option is in effect.

Entry ET:

```
  /SECURE APPC FULL
```

Response ET:

```
DFS058I SECURE COMMAND COMPLETED
```

Example 2 for /SECURE command

Entry ET:

```
  /DIS OTMA
```

Response ET:

```
GROUP/MEMBER XCF-STATUS USER-STATUS    SECURITY TIB INPT SMEM
            DRUEXIT   T/O  ACEEAGE
XCFGRP1
-IMS1       ACTIVE    SERVER          FULL       8000
-IMS1         N/A     0
-HWS1       ACTIVE    ACCEPT TRAFFIC  FULL   0 5000
-HWS1        HWSYDRU0 239  3600
-HWS2       ACTIVE    ACCEPT TRAFFIC  CHECK  0 5000
-HWS2        HWSYDRU0 239  7200
-HWS3       ACTIVE    ACCEPT TRAFFIC  NONE   0 5000
-HWS3        HWSYDRU0 239  0
*09121/172200*   IMS1
```

Explanation: Enter /DISPLAY OTMA to view the security setting of each OTMA tmember.

Entry ET:
```
/SECURE OTMA FULL
```

Response ET:
```
DFS058I SECURE COMMAND COMPLETED
```

Related reference:

Parameter descriptions for IMS procedures (System Definition)

Chapter 19. /SET command

The /SET command establishes the destination of all messages entered into this terminal to another terminal or to a particular transaction code.

If the terminal is in conversation, the /SET command also sets the destination of only the next message to the specified transaction.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 812
- "Examples" on page 813

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 261. Valid environments for the /SET command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/SET	X		X
CONVERSATION	X		X
LTERM	X		X
TRANSACTION	X		X

Syntax

```
►►──/SET──┬─CONVERSATION──tranname─┬──────────────────────────────►◄
          │                        └─(password)─┘                 │
          ├─LTERM──ltermname───────────────────┤
          └─TRANSACTION──tranname──┬────────────┤
                                   └─(password)─┘
```
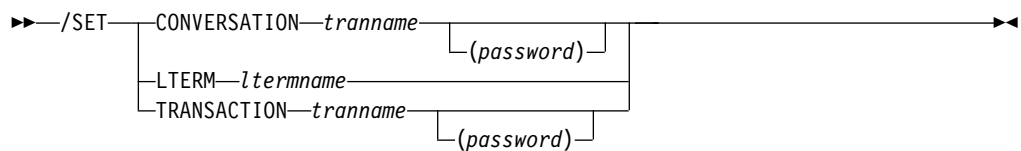
Keywords

The following keywords are valid for the /SET command:

CONVERSATION
> Directs the next input message to a selected transaction. The terminal being used must be in a conversation that is waiting for an input message. For example, the response message must have been received.
>
> In an IMSplex, if global resource information is kept in Resource Manager, the /SET command sets a transaction for the next input message both globally and locally. If global resource information is not kept in Resource Manager, the /SET command sets the transaction just locally.

LTERM

> Specifies the logical terminal that is the destination of all messages entered into this terminal.
>
> The mode established by /SET LTERM is called preset mode. If the preset mode is established from a 3270 and user-defined formats are not being used, input message must be entered from a cleared screen. Preset mode can be reset by:
>
> ```
> /CLSDST NODE
> /SIGN
> /STOP LINE PTERM
> /STOP NODE command
> /STOP USER command
> /RESET
> /STOP NODE
> /STOP LINE
> /STOP USER
> ```
>
> Once a destination is preset, the terminal operator cannot enter the destination (logical terminal name) as the first part of the message.
>
> In a multiple systems configuration, the name of a remote logical terminal can be specified. If the preset destination is to be deleted (/RESET) or changed (/SET), the command must be entered from some other valid component.

TRANSACTION

> Specifies the transaction code that is the destination of all messages entered into this terminal. The mode established by /SET TRANSACTION is called preset mode. If the preset mode is established from a 3270 and user-defined formats are not being used, input messages must be entered from a cleared screen. Preset mode can be reset by:
>
> ```
> /CLSDST NODE
> /SIGN
> /START LINE PTERM
> /RESET
> /STOP NODE
> /STOP LINE
> /STOP USER
> ```
>
> Once a destination is preset, the terminal operator cannot enter the destination (transaction code) as the first part of the message. In a multiple systems configuration, the name of a remote transaction can be specified. The terminal cannot be in conversation.

Usage notes

The status fields of /DISPLAY LINE PTERM, /DISPLAY NODE, or /DISPLAY USER indicate if a physical terminal, node, or user is in preset destination mode (PRST), and display the destination transaction code or logical terminal name.

A transaction name can be defined with password protection in SAF for the CONVERSATION and TRANSACTION keywords. If the resource is not defined to SAF, or is defined and is authorized to the user, the command is processed. If the resource is defined to SAF but not authorized for use, the command is rejected with a DFS2469W message.

A period that is specified in the password brackets is not treated as a delimiter.

Examples

The following are examples of the /SET command:

Example 1 for /SET command

Entry ET:
```
/SET CONVERSATION CONVTRAN(password)
```

Response ET:
```
DFS058I  SET COMMAND COMPLETED
```

Explanation: Any message entered from this terminal is sent to conversation CONVTRAN.

Example 2 for /SET command

Entry ET:
```
/SET LTERM CNTRL
```

Response ET:
```
DFS058I  SET COMMAND COMPLETED
```

Explanation: Any message entered from this terminal is sent to LTERM CNTRL.

Example 3 for /SET command

Entry ET:
```
/SET TRANSACTION IMS(password)
```

Response ET:
```
DFS058I  SET COMMAND COMPLETED
```

Explanation: Any message entered from this terminal is sent to transaction IMS.

Chapter 20. /SIGN command

The /SIGN command is used to sign on and sign off at terminals attached to IMS.

This command enables IMS to identify who is using the terminal and to determine if you are authorized to enter the transaction or command.

Subsections:

- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 819
- "Examples" on page 820

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command can be issued.

*Table 262. Valid environments for the /SIGN command*

Command	DB/DC	DBCTL	DCCTL
/SIGN	X		X

Syntax

A:
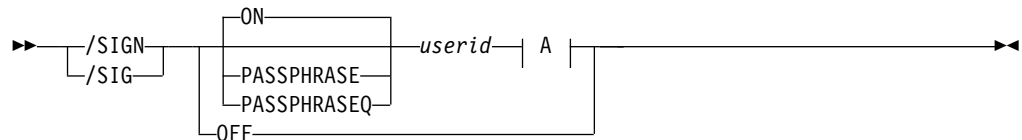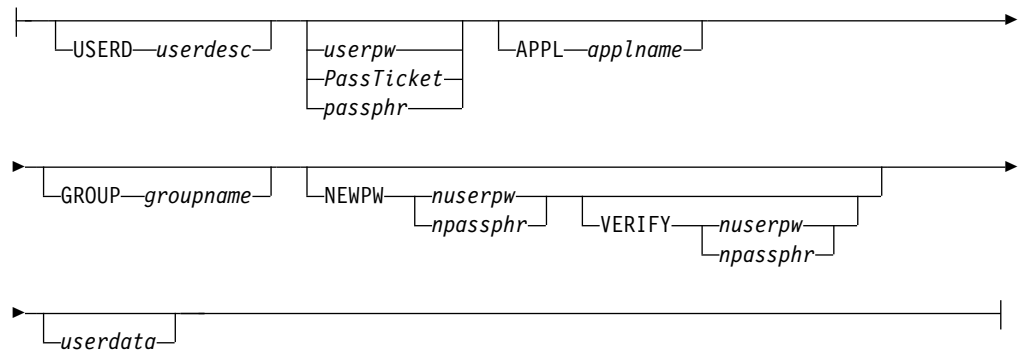
Keywords

The following keywords are valid for the /SIGN command:

ON /SIGN ON must be issued for any physical terminal or user ID requiring a signon, or the transaction entered is rejected.

From terminals that require signon, commands other than /SIGN or /RCLSDST are rejected if transaction authorization is requested. Static terminals requiring a signon also have enhanced command authorization with RACF or an equivalent product if RCF=S or RCF=A is specified at system startup.

At terminals not requiring signon, transactions are passed to RACF, an equivalent security product, or a user exit for authorization checking. If /SIGN ON is entered at a terminal not requiring a signon, the signon is processed as if the terminal required a signon. That is, the terminal is placed in a signed on status with the user ID until a /SIGN OFF or another /SIGN ON command is entered.

After any IMS restart or terminal disconnect, the remote terminal operator is required to sign on again using the /SIGN ON command. A terminal can be disconnected by:
- A switched line disconnect
- A VTAM CLSDST
- A line shutdown
- The /IDLE command
- Auto logoff

Signon status is also reset by the /START LINE, /START LINE PTERM, and /START NODE commands and auto signoff.

The remote terminal operator must wait at a static physical terminal for confidential responses, because responses queued for a given physical terminal are sent even if the physical terminal is signed off. If the remote terminal operator must be absent, the /LOCK command can be used to prevent output from being received. Confidential output sent to a dynamic user is queued to the user instead of to the physical terminal when the user has signed off. A successful signon of an existing user turns off the DEADQ status for the user, if that status exists.

For the user exit routine DFSCSGN0, the user ID and userdata parameter values are defined by the installation.

PASSPHRASE
The /SIGN PASSPHRASE command is equivalent to the /SIGN ON command except that it uses RACF password phrases instead of passwords for a signon. A RACF password phrase can be up to 100 bytes. IMS uses 100 bytes as the password phrase and removes leading and trailing blanks, if any, before passing it to RACF.

RACF password phrases are used for password, NEWPW, and VERIFY. RACF does not allow a mixture of passwords and password phrases. For example, if PASSPHRASE is specified on the /SIGN command, you must specify password phrases for all the other keywords such as NEWPW and VERIFY.

A blank is necessary after PASSPHRASE. There must be a blank between the 100-character password phrase and the next keyword. A period within the 100 character does not end the /SIGN command. If there is no additional keyword after the password phrase, the password phrase does not need to have trailing blanks. If there are additional keywords after the password phrase, the password phrase needs to include trailing blanks for a total of 100 characters. If a password phrase is less than 9 bytes, IMS will pass the password phrase as

a password to RACF. The PASSPHRASE keyword is most likely used on MFS panels, which fill the password phrase with trailing blanks.

RACF password phrases are always mixed case. It is not necessary to turn on mixed-case password for password phrases. The IMS system's default MFS panels do not support password phrases.

PASSPHRASEQ

The /SIGN PASSPHRASEQ command is equivalent to the /SIGN ON command except that it uses RACF password phrases instead of passwords for a signon. A password phrase must start with a single quotation mark (') and end with a single quotation mark. If you want to include a single quotation mark in a password phrase, you must specify two single quotation marks (''). For example,

```
'This is "my" passphrase.'
```

IMS removes the single quotation mark at the beginning and ending of the password phrase and also removes one single quotation mark if there are two single quotation marks following each other. PASSPHRASEQ must have at least one blank before the beginning single quotation mark. A password phrase can be up to 100 characters. If a password phrase is less than 9 characters, IMS will pass it as a password to RACF. RACF does not allow a mixture of passwords and password phrases. For example, if PASSPHRASEQ is specified on the /SIGN command, you must specify password phrases for all the other keywords such as NEWPW and VERIFY.

RACF password phrases are always mixed case. It is not necessary to turn on mixed-case password for password phrases. The IMS system's default MFS panels do not support password phrases.

OFF

The /SIGN OFF command is used to complete a session on a terminal that required a signon. Static terminals in conversational mode cannot be signed off without first issuing an /EXIT or /HOLD command.

Another method of signing off a terminal is to reenter the /SIGN ON command. This method initiates a new signon at the terminal without having to enter the /SIGN OFF command.

The /SIGN OFF command resets status that is not significant such as preset mode, test mode, lock lterm, pstop lterm, and purge lterm.

/SIGN OFF for ETO users will also take other actions depending on the recovery settings for the user:

RCVYCONV=NO

/SIGN OFF causes any IMS conversations (active and held) for an ETO user to be terminated. Any conversational message that is queued or being processed has its output response message delivered asynchronously.

RCVYFP=NO

/SIGN OFF causes Fast Path status and messages for an ETO user to be discarded.

RCVYRESP=NO

/SIGN OFF resets full-function response mode.

If global resource information is kept in Resource Manager, /SIGN OFF deletes the user ID from Resource Manager (if single user signon enforced) and resets

status globally. If the user has no status, /SIGN OFF deletes the user and associated lterms from Resource Manager.

You can specify the following keywords and parameters with the ON, PASSPHRASE, or PASSPHRASEQ keyword:

APPL

A keyword that notifies IMS that the following character string should be the application name used by IMS when IMS makes the SAF call to verify the user. The default application name used by IMS is the IMSID. The IMSID can be overridden by the SAPPLID= parameter in the IMS PROCLIB member DFSDCxxx. If the signon specifies a PassTicket instead of a password, the APPL parameter should specify the application name used when the PassTicket was created. The creator of the PassTicket can specify any value to identify an IMS subsystem.

If RACF is used, APPL= should specify the name of the RACF PTKTDATA profile for IMS as defined to RACF by the creator of the PassTicket. If the name of the PTKTDATA profile is the same as the IMSID, the APPL keyword is not needed.

GROUP

Is an optional keyword indicating a group name of 8 characters or fewer that is associated with the user ID.

NEWPW

Is an optional keyword that indicates a new user password or a new password phrase that replaces the current user password or password phrase specified in *userpw*. Passwords can be mixed case or uppercase depending on what is specified on the PSWDC keyword in the DFSPBxxx IMS.PROCLIB member. RACF password phrases are always mixed case.

nuserpw

Is a new password of 8 characters or fewer that is associated with the user identification.

npassphr

Is a 9- to 100-character password phrase that is associated with the user identification. If PASSPHRASE is specified, the password phrase must be up to 100 characters. If PASSPHRASEQ is specified, the password phrase must be enclosed in single quotation marks. If a password phrase contains one or more single quotation marks, two single quotation marks must be specified for each single quotation mark.

USERD

Is a user descriptor name. This user descriptor name is used in the signon. The *userdesc* parameter must be a user ID, node name, or DFSUSER.

userdata

Is user identification information that has been defined to IMS with the (RACF), equivalent security product or the user exit routine, DFSCSGN0. For RACF, this information consists of the following:

```
userpw  GROUP groupname   NEWPW nuserpw
```

userid

Is a user identification of 8 characters or fewer.

userpw | PassTicket | passphr

Specifies user identification in one of the following formats:

userpw
> Is a password of 8 characters or fewer that is associated with the user
> identification. Passwords can be mixed case or uppercase depending on
> what is specified on the PSWDC keyword in the DFSPBxxx IMS.PROCLIB
> member. If support for special characters is enabled in RACF, IMS supports
> RACF passwords that contain special characters.

PassTicket
> A one-time password that is generated by a requesting product or function.
> The *PassTicket* is an alternative to the RACF password. Using a *PassTicket*
> removes the need to send RACF passwords across the network in clear
> text.

passphr
> Is a 9- to 100-character password phrase that is associated with the user
> identification. If PASSPHRASE is specified, the password phrase must be
> up to 100 characters. If PASSPHRASEQ is specified, the password phrase
> must be enclosed in single quotation marks. If a password phrase contains
> one or more quotation marks, two single quotation marks must be
> specified for each single quotation mark.

VERIFY
> Is an optional keyword that requests IMS to verify the new password entered.
> IMS verifies the new password before passing it to RACF or to the IMS signon
> exit routines. This keyword can also be used as an alternative to reentering the
> password on the DFS3656 panel.
>
> **Restriction:** You can use this keyword only when responding to an IMS
> DFS3656A message and as an alternative to reentering the password on the
> DFS3656 panel.

nuserpw
> Is a new password of 8 characters or fewer that is associated with the user
> identification.

npassphr
> Is a 9- to 100-character password phrase that is associated with the user
> identification. If PASSPHRASE is specified, the password phrase must be
> up to 100 characters. If PASSPHRASEQ is specified, the password phrase
> must be enclosed in single quotation marks. If a password phrase contains
> one or more single quotation marks, two single quotation marks must be
> specified for each single quotation mark.

Usage notes

When SGN=G, Z, or M is specified, the user can sign on multiple times to both
STATIC and ETO terminals when the structure name is different from the user ID.

For a static terminal, or a dynamic terminal that has the same SPQBname as the
node name, a user will not be allowed to sign on unless all conversations are held,
or the user is authorized to use the transaction for the active conversation.

If there is an active conversation for a static terminal, and the user is not
authorized to use its transaction, the user can enter a /HOLD command prior to
signing on to put all of the conversations in a held state. The user will then be
allowed to sign on.

If there is an active conversation for a dynamic terminal that has the SPQBname the same as the node name, only a user that is authorized to use the transaction of the active conversation will be allowed to sign on. The /HOLD command is not allowed prior to signing on for a dynamic terminal.

If there is an active conversation for a dynamic terminal that has the SPQBname the same as the USERID, the conversation will be associated with that user at signoff. That same user can sign on to any dynamic terminal and continue the conversation if they are still authorized to use the conversational transaction. Any new user that signs on to the dynamic terminal will not be in a conversation unless they are continuing a conversation from a previous signon or starting a new conversation by entering an authorized conversational transaction.

The status fields of /DISPLAY NODE and /DISPLAY LINE PTERM indicate whether a terminal is signed on with the word SIGN.

You can use password phrases for user identification.

A period is normally used as the delimiter at the end of IMS commands. When support for special characters is enabled in RACF, a period becomes a valid character in the RACF password. Therefore, when a password is specified at the end of the /SIGN command, you must insert a space before the period that you are using as the end-of-command delimiter. If a space is not added before the period that you are using as the end-of-command delimiter and support for special characters is enabled in RACF, the period is treated as part of the password and not as a delimiter.

One or more equal signs (=) can normally be used as a valid delimiter between a keyword and the keyword value on the /SIGN command. However, when support for special characters is enabled in RACF, an equal sign becomes a valid character in the RACF password. If you use the equal sign as a delimiter when support for special characters is enabled in RACF, you can specify only one delimiter immediately preceding the new password on the **NEWPW** keyword and the new password on the **VERIFY** keyword. Any equal sign that follows the delimiter is interpreted as the first character of the password.

For IMS 14 to support special characters in RACF passwords when special characters is also enabled in RACF, install APAR/PTFs PI48111/UI34376/UI34377. Also, ensure that all IMS systems that require support for RACF special characters are migrated to IMS 14.

Examples

The following are examples of the /SIGN command:

Example 1 for /SIGN *command*

Entry ET:
```
DFS3649A /SIGN COMMAND REQUIRED FOR IMS

DATE: 11/03/92     TIME: 14:39:33

NODE NAME: DT327001

USERID: IMSUS01

PASSWORD: IMSPW01
```

```
USER DESCRIPTOR:
GROUP NAME:
NEW PASSWORD:

    OUTPUT SECURITY AVAILABLE
```

Response ET:

```
DFS3650I SESSION STATUS FOR IMS

DATE: 11/03/92     TIME: 14:41:48
NODE NAME:          DT327001
USERID:             IMSUS01
PRESET DESTINATION:

CURRENT SESSION STATUS:

    OUTPUT SECURITY AVAILABLE
```

Explanation: The user with user ID IMSUS01 and password IMSPW01 has successfully signed on to a dynamic terminal. The signon is done with the panel (DFS3649A).

Example 2 for /SIGN *command*

Entry ET:

```
/SIGN IMSUS02 IMSPW02
```

Response ET:

```
DFS3650I SESSION STATUS FOR IMS

DATE: 11/03/92     TIME: 14:41:48
NODE NAME:          DT327001
USERID:             IMSUS02
PRESET DESTINATION:

CURRENT SESSION STATUS:

    OUTPUT SECURITY AVAILABLE
```

Explanation: The user with user ID IMSUS02 and password IMSPW02 has successfully signed on to a dynamic terminal. The signon is done with the /SIGN command.

Example 3 for /SIGN *command*

Entry ET:

```
/SIGN IMSUS03 IMSPW03
```

Response ET:

```
DFS3650I SESSION STATUS FOR IMS

DATE: 11/03/92     TIME: 14:45:53
NODE NAME:          L3270A
USERID:             IMSUS03
PRESET DESTINATION:

CURRENT SESSION STATUS:

NO OUTPUT SECURITY AVAILABLE
```

Explanation: The user with user ID IMSUS03 and password IMSPW03 has successfully signed on to a static terminal.

Example 4 for /SIGN *command*

Entry ET:

```
/SIGN PASSPHRASEQ IMSUS03 'this is my ''password'' now'
```

Response ET:

```
DFS3650I SESSION STATUS FOR IMS

DATE: 06/07/13      TIME: 15:26:42
NODE NAME:          L3270A
USERID:             IMSUS03
PRESET DESTINATION:

CURRENT SESSION STATUS:

NO OUTPUT SECURITY AVAILABLE
```

Explanation: The user with user ID IMSUS03 and password phrase `this is my 'password' now` has successfully signed on to a static terminal.

Example 5 for /SIGN *command*

Entry ET:

```
/SIGN PASSPHRASE IMSUS03 this is my 'password' now.
```

Response ET:

```
DFS3650I SESSION STATUS FOR IMS

DATE: 06/07/13      TIME: 15:36:42
NODE NAME:          L3270A
USERID:             IMSUS03
PRESET DESTINATION:

CURRENT SESSION STATUS:

NO OUTPUT SECURITY AVAILABLE
```

Explanation: The user with user ID IMSUS03 and password phrase `this is my 'password' now.` has successfully signed on to a static terminal. Note that the period is part of the password phrase and that no trailing blanks have been added.

Related concepts:

[➡] z/OS: RACF secured signon PassTicket

Related tasks:

[➡] z/OS: Using the secured signon function

Related reference:

[➡] z/OS: RACF password authentication options

Chapter 21. /SMCOPY command

The /SMCOPY command is used by the master terminal operator to control the printing of certain output to the secondary master terminal.

This command can be issued to an IMSplex using the Batch SPOC utility.

Subsections:
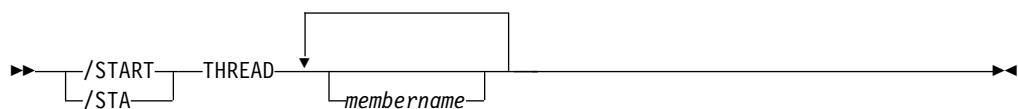- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 824
- "Example" on page 824

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 263. Valid environments for the /SMCOPY command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/SMCOPY	X		X
MASTER	X		X
MSG	X		X
TERMINAL	X		X

Syntax

```
>>--+-/SMCOPY-+--+-->-+-MASTER---+-ON--+--+-------------------------><
    +-/SMC----+  |      |         +-OFF-+  |
                 |      |                  |
                 |      +-TERMINAL-+-ON--+-+
                 |      |          +-OFF-+ |
                 |      |                  |
                 |      +-MSG------+-ON--+-+
                 |                 +-OFF-+
                 +-----,<----------------+
```
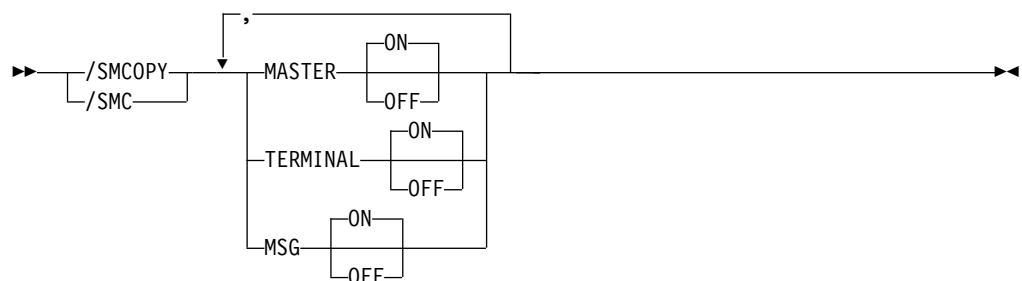
Keywords

The following keywords are valid for the /SMCOPY command:

MASTER
Turns ON or OFF the printing of the above subset of IMS commands and command responses when issued from the master terminal.

TERMINAL

Turns ON or OFF the printing of the above subset of IMS commands and command responses when issued from terminals other than the master terminal.

Input coming through Operations Manager is not reflected in the secondary master. This also applies to input coming from MCS/E-MCS terminals.

MSG

ON All system messages will be logged to the secondary master. This is the default.

Before logging a message to the secondary master, if the installation type-2 exit routine indicated it needs to be called for messages, the exit routine is called first. The exit routine then determines if the message is logged to the secondary master.

OFF

IMS does not send system messages to the secondary master.

Usage notes

System definition establishes whether the commands and their responses will be printed on the secondary master and the origin of the printed command (master terminal, remote terminal, or both). /SMCOPY provides online control of the printing established by system definition.

Example

A system definition has established that copies of the above subset of commands and command responses, when issued from any terminal, will be printed on the secondary master terminal (operand COPYLOG=ALL was specified on the COMM macro).

Entry ET:

```
/SMCOPY TERMINAL OFF
```

Response ET:

```
DFS058I  SMCOPY COMMAND COMPLETED
```

Explanation: The secondary master terminal does not receive copies of IMS commands and command responses issued from remote terminals. Commands and responses issued from the master terminal are still received.

Related reference:

➭ IMS type-1 commands logged to the secondary master terminal (Commands)

Chapter 22. /SSR command

The /SSR command is a multisegment command that allows the IMS operator to enter an external subsystem command as part of the command input. (The external system is not a CCTL subsystem.)

Subsection:
- "Environment"
- "Syntax"
- "Usage notes"
- "Example"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command can be issued.

Table 264. Valid environments for the /SSR command

Command	DB/DC	DBCTL	DCCTL
/SSR	X	X	X

Syntax

```
►►──/SSR──text────────────────────────────────────────────────►◄
```

Usage notes

Routing is the only function IMS performs. The command is processed by the external subsystem and the response (from the external subsystem, not CCTL) is routed back to the entering terminal.

All /SSR formats require an EOM indication to denote end-of-message. An EOS indication must be included for all segments that precede the last segment.

text is the alphanumeric external subsystem command.

Example

Entry ET:
```
/SSR ;START DATABASE (DSN8D22P)
```

Response ET:
```
DFS058I  SSR COMMAND COMPLETED

DSN9022I ; DSNTDDIS 'START DATABASE' NORMAL COMPLETION
```

Explanation: The START DATABASE command is successfully routed to the DB2 subsystem for processing.

Chapter 23. /START commands

The /START commands make IMS resources available for reference and use.

/START also checks the validity of all parameters entered by the terminal operator. If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and the /START command processes the rest of the parameters. For example,

```
/START LINE 4 6 200
DFS058 START COMMAND COMPLETED EXCEPT LINE 200
```

signifies parameter value 200 is not a valid line number.

When a resource becomes available, the system parameters used for this initialization of IMS are displayed in message DFS1929I. The system parameters are also written to the job log.

These commands can be issued to an IMSplex using the Batch SPOC utility.

/START commands are:

- "/START APPC command" on page 828
- "/START AREA command" on page 828
- "/START AUTOARCH command" on page 832
- "/START CLASS command" on page 833
- "/START DATAGRP command" on page 833
- "/START DB command" on page 837
- "/START DC command" on page 846
- "/START DESC command" on page 846
- "/START ISOLOG command" on page 847
- "/START LINE command" on page 848
- "/START LTERM command" on page 850
- "/START LUNAME command" on page 851
- "/START MADSIOT command" on page 852
- "/START MSNAME command" on page 853
- "/START NODE command" on page 854
- "/START OLDS command" on page 856
- "/START OTMA command" on page 857
- "/START PGM command" on page 858
- "/START REGION command" on page 859

- "/START RTC command" on page 862
- "/START SB command" on page 863
- "/START SERVGRP command" on page 864
- "/START SLDSREAD command" on page 865
- "/START SUBSYS command" on page 866
- "/START SURV command" on page 867
- "/START THREAD command" on page 868
- "/START TMEM command" on page 869
- "/START TRAN command" on page 872
- "/START TRKARCH command" on page 875
- "/START USER command" on page 875
- "/START VGR command" on page 877
- "/START WADS command" on page 878
- "/START XRCTRACK command" on page 879

/START APPC command

Use the /START APPC command to instruct IMS to activate the connection to
APPC/z/OS and to start accepting transaction schedule requests from
APPC/z/OS.

This command reverses the effect of a /PURGE APPC command or a /STOP
APPC(CANCEL) command.

The /START APPC command sets the desired status to ENABLED. The current
status is initially set to STARTING. When APPC/z/OS responds to the start
request, the status changes to either ENABLED or FAILED.

Subsections:
- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 265. Valid environments for the /START APPC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
APPC	X		X

Syntax

```
►►──┬─/START─┬──APPC──────────────────────────────────────────────►◄
    └─/STA───┘
```

/START AREA command

Use the /START AREA command to specify the specific areas of DEDBs to be
allocated. For z/OS, you can use the /START AREA command to reallocate DEDB
areas.

Subsections:
- "Environment"
- "Syntax" on page 829
- "Keywords" on page 829
- "Usage notes" on page 830
- "Equivalent IMS type-2 commands" on page 831
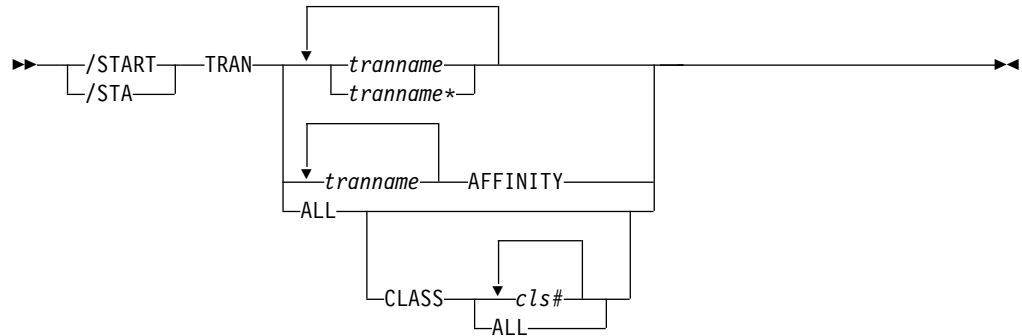- "Examples" on page 831

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 266. Valid environments for the /START AREA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
AREA	X		X
GLOBAL	X	X	
LOCAL	X		

Syntax

```
>>──┬─/START─┬──AREA──┬─┬──areaname──┬──┬─LOCAL──┬──────────────────────><
    └─/STA───┘        │ └────────────┘  └─GLOBAL─┘
                      │                            
                      └─ALL──┬─LOCAL─┬────────────┘
                             └───────┘
```

Keywords

The following keywords are valid for the /START AREA command:

GLOBAL

> The GLOBAL keyword applies when an IRLM is active. GLOBAL specifies that the command applies to all subsystems sharing the area. The GLOBAL keyword and the ALL parameter are mutually exclusive. The /START command is rejected if both ALL and GLOBAL are specified. The GLOBAL keyword requires that IRLM be active. The command will be rejected if IRLM is not active.
>
> If the area is registered in the RECON data set, the /STA AREA GLOBAL command will reset the prohibit authorization flag to off (PROHIBIT AUTH = OFF).
>
> The GLOBAL command is processed by the IMS system where the command was initiated. This system will process the command locally and then request IRLM NOTIFY to route and process the command on sharing IMS systems.
>
> If global DB or AREA status is maintained, the global status maintained in RM is also updated. The global status is set to STOA.
>
> If the command is entered from OM API, the global status is updated by the command master IMS. If the command is not entered from OM API, the IMS that initiated the GLOBAL command updates the global status in RM.
>
> You must specify if the global area status must be maintained in RM. You can specify this during IMS initialization in either the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSAREA(Y)). You can also change it dynamically using the UPDATE IMS SET(PLEXPARM(GSTSAREA(Y))) command. If you do not specify that global database status is to be maintained, the GLOBAL keyword is processed as in prior releases, and the global status is not updated.
>
> If global status in RM is successfully updated, message DFS0988I for RSRCTYPE=AREA is issued. If global status is not successfully updated, message DFS3308I is issued, indicating RM failure, and no command response

lines are generated. Any RM error is traced to the OCMD trace table. Users can issue a QRY AREA STATUS(GLOBAL) command to obtain the global status of the resources in RM.

The X'594C' log record for databases is updated to include both global status and global command time stamp.

The GLOBAL keyword is not supported on an RSR tracking subsystem.

If the GLOBAL keyword on a command is entered from an OM API, the command is processed only by the command master IMS. The command master IMS will make DBRC calls to update the RECON with GLOBAL status. It will also request IRLM NOTIFY to route and process the command on sharing IMS systems, and then process the command locally. All other non-master IMS systems ignore the /START command with the GLOBAL keyword.

Messages produced on the NOTIFIED systems will appear only on the system console and will not be routed back to the OM API which originally entered the command.

If multiple IMS systems have been explicitly specified in the route list, the master IMS system will process the command as described previously. However, the non-master IMS systems, to which OM routes the command, will reject the command with the following return and reason code listed in the following table:

Table 267. Return and reason code for GLOBAL keyword issued from the OM API

Return code	Reason code	Meaning
X'00000004'	X'00001000'	The command contained the GLOBAL keyword and was routed to more than one IMS system in the IMSPLEX. The non-master IMS systems will reject this command when OM routes the command to them. The master IMS system will process this command and use IRLM NOTIFY to route and process the command on the non-master IMS systems. See the discussion under the GLOBAL keyword.

LOCAL
Specifies that the command only applies to the IMS subsystem in which the command is entered. This command does not affect any other subsystem sharing the area.

LOCAL is the default.

Usage notes

For areas on an RSR tracking subsystem, /START AREA is used to resume tracking for those areas that were stopped by a previous /DBRECOVERY command or by errors found during tracking subsystem processing. /START AREA also starts online forward recovery (OFR) for those areas that are not current with mainline tracking.

For virtual storage option (VSO) areas that have been defined with the PREOPEN option, /START AREA causes the areas to be preopened. If the VSO area is defined with the PRELOAD option, /START AREA causes the area to be opened and loaded into the z/OS data space.

Restriction: This command only applies to the IMS subsystem on which it is entered; it does not preload or preopen areas on other IMS subsystems in the sysplex that share the area.

The /START AREA command has no effect on VSO areas that are in virtual storage when the command is issued.

The output of the /START AREA command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. For commands that specify GLOBAL, only the command master returns the asynchronous messages to OM. When a command is processed with the LOCAL keyword, all IMS systems are able to return the asynchronous messages to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS0011I, DFS140I, DFS0488I, DFS0666I, DFS1407I, DFS2980E, DFS2981E, DFS3320I, DFS3325I, DFS3342I, DFS3720I, DFS3824I

/START AREA ALL causes message DFS0488 to be issued for every area that is not started successfully, but you do not see a DFS0488 message for every area that does start successfully. You do see a final DFS0488 message which indicates the end of command processing.

While the database is being quiesced, this command cannot be processed successfully.

Equivalent IMS type-2 commands

The following table shows variations of the /START AREA command and the IMS type-2 commands that perform similar functions.

Table 268. Type-2 equivalents for the /START AREA command

Task	/START AREA command	Similar IMS type-2 command
Starts the area.	/START AREA *areaname*	UPDATE AREA NAME(*areaname*) START(ACCESS)

Examples

The following is an example of the /START AREA command:

Entry ET:

```
  /START AREA DB1AREA0 DB1AREA1
```

Response ET:

```
DFS058I   START COMMAND IN PROGRESS
DFS0488I  START COMMAND COMPLETED.   AREA=DB1AREA0
DFS0488I  START COMMAND COMPLETED.   AREA=DB1AREA1
```

Explanation: DEDB areas DB1AREA0 and DB1AREA1 are started.

Related concepts:

➡ Maintaining global information for databases, DEDB areas, and transactions (System Administration)

Related reference:

/START AUTOARCH command

Use the /START AUTOARCH command to set the value to change the automatic archiving option selected at system initialization or to set the value to start automatic archiving after a previous /STOP AUTOARCH command.

Subsections:
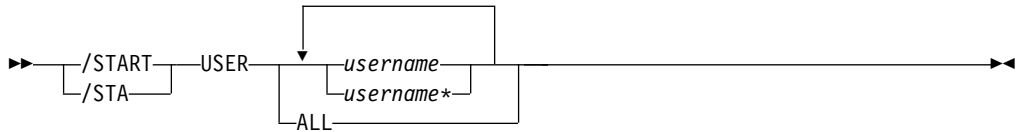- "Environment"
- "Syntax"
- "Keywords"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 269. Valid environments for the /START AUTOARCH command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
AUTOARCH	X	X	X

Syntax

```
►►──┬─/START─┬──AUTOARCH─────────────────────────────────────────►◄
    └─/STA───┘          └─#olds─┘
```
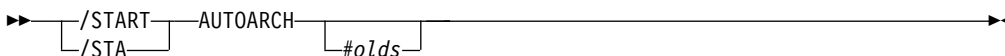
Keywords

The following keywords are valid for the /START AUTOARCH command:

#olds

Specifies the number of OLDSs that are to be filled before the /DBRC GENJCL ARCHIVE command is generated. It is optional and defaults to either the value specified at system initialization or to one. If /DBR NOFEOV or /DBD NOFEOV is issued before *nn* OLDSs are filled, the number of OLDSs currently filled will be archived.

Examples

The following is an example of the /START AUTOARCH command:

Entry ET:
```
/START AUTOARCH 4
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Explanation: Automatic archiving will be initiated after 4 OLDS data sets are filled.

/START CLASS command

Use the /START CLASS command to specify transaction class, allowing scheduling of application programs to begin.

Message regions must have appropriate classes assigned to them before scheduling will proceed.

Subsections:
- "Environment"
- "Syntax"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 270. Valid environments for the /START CLASS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
CLASS	X		X

Syntax

Examples

The following is an example of the /START CLASS command:

Entry ET:
```
 /START CLASS ALL
```

Response ET:
```
 DFS058I  START COMMAND COMPLETED
```

Explanation: All classes of transactions are made available for scheduling into message processing regions.

/START DATAGRP command

Use the /START DATAGRP command to specify groups of DL/I databases and Fast Path DEDBs to be allocated.

Data groups are logical groupings of databases and areas; they enable simplified command processing for databases. You define a data group in the RECON data set by using the INIT.DBDSGRP command with parameters GRPNAME and DBGRP.

DATAGRP is valid on active and RSR tracking subsystems.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 836
- "Equivalent IMS type-2 commands" on page 836

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 271. Valid environments for the /START DATAGRP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
ACCESS	X	X	
DATAGRP	X	X	
LOCAL	X	X	

Syntax

A:

B:

Keywords

The following keywords are valid for the /START DATAGRP command:

ACCESS

Overrides the access intent for a database. The original database access is specified during IMS system definition.

Changing the access intent for a DEDB is allowed only when no PSBs are scheduled that access any areas in the DEDB. You might have to stop PSBs and regions that have wait-for-input (WFI) transactions scheduled before you can change a DEDB's access intent.

Changing the access intent of a database with the /START DB ACCESS= command causes any VSO areas of the database to be removed from virtual storage. Because the access intent of a DEDB cannot be changed while any of its areas are authorized to an IMS subsystem, IMS closes any open areas before processing the command. If a VSO area is closed as a result of the command, it is also removed from the data space.

The /START AREA command must be used to reactivate the VSO options (VSO and PREOPEN) and PRELOAD option for the area. If an area is opened as a result of an access request rather than by the /START AREA command, it is opened as a non-VSO area.

Non-VSO areas with the PREOPEN option are closed as a result of the /START DB ACCESS= command. These areas are reopened either at the next access request for the area or by the /START AREA command.

The GLOBAL and ACCESS keywords are mutually exclusive. The /START command is rejected if both keywords are specified.

The meanings of the ACCESS parameter values are:

RO   Specifies that the named database is available for read-only processing on this IMS subsystem.

The only programs which can use the database on this subsystem are those which have a PCB processing option of GO (`PROCOPT=GO`). Programs which access the data using the GO processing option might see uncommitted data, since a sharing IMS subsystem could be updating the database, which is opened for input only.

RD   Specifies that the named database is available for read-only processing on this IMS subsystem.

Programs with update intent can be scheduled, but cannot update the database. ACCESS=RD differs from ACCESS=RO in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UP   Specifies that the named database is for update as well as read processing in the IMS subsystem.

EX   Specifies that the named database is to be used exclusively by this IMS subsystem.

This exclusive access is guaranteed only when the database is registered to DBRC.

NODBALLOC | DBALLOC
Indicates whether the databases within the data group are to be allocated. NODBALLOC is the default.

NODBALLOC
Indicates that the databases within the data group are not to be allocated. The databases will be allocated when they are scheduled. This command does not affect any other subsystem sharing the database.

DBALLOC
Indicates that the databases within the data group are to be allocated.

LOCAL
Specifies that the /START command applies only to the IMS subsystem in which the command is entered.

Usage notes

For databases and areas on an RSR tracking subsystem, /START DATAGRP is used to resume tracking for those areas that were stopped by a previous /DBRECOVERY command or by errors found during tracking subsystem processing. /START DATAGRP also starts online forward recovery (OFR) for those databases and areas that are not current with mainline tracking.

After processing for a /START DATAGRP completes, a DFS0488 message is issued indicating the end of processing. A DFS0488 message is also issued for every database or area that does not start successfully.

If the data group contains both full function and Fast Path databases, a DFS0488 message might be issued indicating the /START DATAGRP command completed successfully before any messages are issued that indicate a Fast Path area did not start successfully. This situation is caused by the asynchronous processing of Fast Path databases.

For virtual storage option (VSO) areas that have been defined with the PREOPEN option, /START AREA causes the areas to be preopened. If the VSO area is defined with the PRELOAD option, /START AREA causes the area to be opened and loaded into the z/OS data space.

A data group is defined in the RECON data set using the INIT.DBDSGRP command with the parameters GRPNAME and DBGRP or DBDSGRP. The DATAGRP keyword on the /START command can specify either a DBDS group or a database group name.

If the ACCESS keyword is specified on the /START DATAGRP command along with the DBDS group name, the ACCESS keyword is not applied to the Fast Path DEDB databases associated with the Fast Path DEDB areas in the DBDS group. If the intent is to use the ACCESS keyword for Fast Path DEDB databases, the DATAGRP parameter must be a database group name that does not contain area names.

Recommendation: Although you can use DBDS groups as well as database groups for this command, you should use database groups whenever possible to eliminate the overhead of converting the DBDS group to a database group.

Equivalent IMS type-2 commands

The following table shows variations of the /START DATAGRP command and the IMS type-2 commands that perform similar functions.

Table 272. Type-2 equivalents for the /START DATAGRP command

Task	/START DATAGRP command	Similar IMS type-2 command
Starts the data group.	/START DATAGRP *datagrpname*	UPDATE DATAGRP NAME(*datagrpname*) START(ACCESS)

Related reference:
"UPDATE DATAGRP command" on page 1021

/START DB command

Use the /START DB command to specify the DBD name. The /START DB command permits access from transactions that read or update databases.

The /START DB command can be used to allocate or reallocate all databases other than DEDBs. The /START AREA command must be entered to allocate or deallocate DEDB areas.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 838
- "Usage notes" on page 841
- "Equivalent IMS type-2 commands" on page 844
- "Examples" on page 844

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 273. Valid environments for the /START DB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
ACCESS	X	X	
DB	X	X	
DBALLOC	X	X	
GLOBAL	X	X	
LOCAL	X	X	
NOBACKOUT	X	X	
NODBALLOC	X	X	
NOOPEN	X	X	
OPEN	X	X	

Syntax

A:

```
├────────────────────────────────────────────────────────────────┤
         └─ACCESS=─┬─RO─┬─┘
                   ├─RD─┤
                   ├─UP─┤
                   └─EX─┘
```

B:

```
          ┌─NODBALLOC─┐
├─────────┼───────────┼──────────────────────────────────────────┤
          └─DBALLOC───┘
```

C:

```
                      ┌─DBALLOC───┐  ┌─NOOPEN─┐
├──────────────────────┼───────────┼──┼────────┼──────────────────┤
          └─NOBACKOUT─┘ └─NODBALLOC─┘  └─OPEN───┘
```
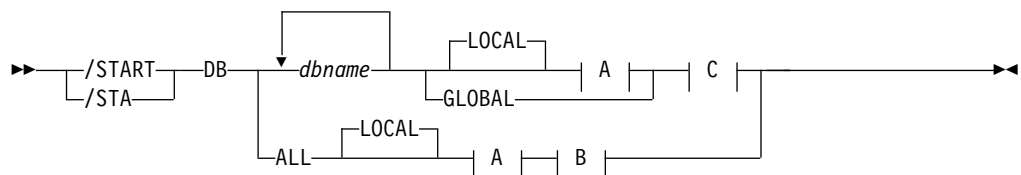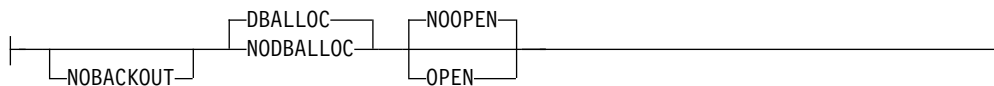
Keywords

The following keywords are valid for the /START DB command:

ACCESS

Specifies the type of access that is intended for the named database. This keyword overrides the database access that is specified at system definition. Changing the ACCESS parameter of a DEDB is allowed only when all the AREAS in the DEDB are not authorized by the subsystem.

Changing the access intent of a database with the /START DB ACCESS= command causes any VSO areas of the database to be removed from virtual storage. Because the access intent of a DEDB cannot be changed while any of its areas are authorized to an IMS subsystem, IMS closes any open areas before it processes the command. If a VSO area is closed as a result of the command, it is also removed from the data space.

The /START AREA command must be used to reactivate the VSO options (VSO and PREOPEN) and PRELOAD option for the area. If an area is opened as a result of an access request rather than by the /START AREA command, it is opened as a non-VSO area.

Non-VSO areas with the PREOPEN option are closed as a result of the /START DB ACCESS= command. These areas are reopened either at the next access request for the area or by the /START AREA command.

The GLOBAL and ACCESS keywords are mutually exclusive. The /START command is rejected if both keywords are specified.

Issuing a /START DB command with the ACCESS parameter is not valid for an MSBD.

The meanings of the ACCESS parameter values are:

RO Specifies that the named database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data by using the GO

processing option might see uncommitted data, because a sharing IMS subsystem could be updating the database, which is opened for input only.

RD Specifies that the named database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. ACCESS of RD differs from ACCESS of RO in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UP Specifies that the named database is for update and read processing in the IMS subsystem.

EX Specifies that the named database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC.

DBALLOC
Indicates that the specified databases are to be allocated. DBALLOC is the default except for /START DB ALL commands.

NODBALLOC
Indicates that the specified databases are not to be allocated. The databases will be allocated when they are scheduled. NODBALLOC is the default for /START DB ALL commands.

GLOBAL
Specifies that the command applies to all subsystems that are sharing the database. GLOBAL requires that IRLM be active. The command is rejected if IRLM is not active.

If the database is registered in the RECON data set, the /START DB GLOBAL command will reset the prohibit authorization flag to off (PROHIBIT AUTH = OFF).

The /START DB GLOBAL command is processed by the IMS system where the command was initiated. This system will process the command locally and then request IRLM NOTIFY to route and process the command on sharing IMS systems.

If global DB status is maintained, the global status that is maintained in RM is also updated. The global status is set to STA.

If the command is entered from OM API, the global status is updated by the command master IMS. If the command is not entered from OM API, the IMS that initiated the GLOBAL command updates the global status in RM.

You must specify whether the global database status must be maintained in RM. You can specify this during IMS initialization in the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSDB(Y)). You can also change it dynamically using the UPD IMS SET(PLEXPARM(GSTSDB(Y)) command. If you do not specify that global database status is to be maintained, the GLOBAL keyword is processed as in prior releases, and the global status is not updated.

If global status in RM is successfully updated, message DFS0988I for RSRCTYPE=DB is issued. If global status is not successfully updated, message DFS3308I is issued, indicating RM failure, and no command response lines are generated. Any RM error is traced to the OCMD trace table. Users can issue a QRY DB STATUS(GLOBAL) command to obtain the global status of the resources in RM.

The X'4C' log record for databases is updated to include both global status and global command time stamp. It includes the DEDB name for the area.The GLOBAL keyword and the ALL parameter are mutually exclusive. The /START command is rejected if both ALL and GLOBAL are specified.

The GLOBAL keyword is not supported on an RSR tracking subsystem.

If the GLOBAL keyword on a command is entered from an OM API, the command should only be routed to one IMS system in the IMSplex. The IMS that receives the command from OM will make DBRC calls to update the RECON with GLOBAL status. It will also request IRLM NOTIFY to route and process the command on sharing IMS systems, and then process the command locally.

Messages that are produced on the NOTIFIED systems will be displayed only on the system console and will not be routed back to the OM API that originally entered the command.

If multiple IMS systems have been explicitly specified in the route list, the master IMS system will process the command as described previously. However, the non-master IMS systems, to which OM routes the command, will reject the command with the following return and reason code listed in the following table:

Table 274. Return and reason code for GLOBAL keyword issued from the OM API

Return code	Reason code	Meaning
X'00000004'	X'00001000'	The command contained the GLOBAL keyword and was routed to more than one IMS system in the IMSplex. The non-master IMS systems will reject this command when OM routes the command to them. The master IMS system will process this command and use IRLM NOTIFY to route and process the command on the non-master IMS systems. See the discussion under the GLOBAL keyword.

LOCAL

Specifies that the command applies only to the IMS subsystem in which the command is entered. This command does not affect any other subsystem sharing the database.

LOCAL is the default.

NOBACKOUT

Suppresses backout restart for a database not registered in DBRC. If there was a prior dynamic backout or emergency restart backout failure, then a /START, command will attempt to perform the backout again. However, if the log data required to perform the backout has been archived, the backout must be performed by executing the Batch Backout utility.

If the database is registered in DBRC and is using share control, then DBRC is informed when batch backout is successfully executed, and the failing backout will not be attempted again when the /START command is issued.

If the database is registered in DBRC and is using recovery control, DBRC is not informed when batch backout is successfully executed. You must specify the NOBACKOUT keyword to inform IMS that it does not have to attempt to execute the failed backout again.

NOBACKOUT is not valid with the ALL parameter.

OPEN | NOOPEN

Indicates that the named databases should be opened or should not be opened as part of the /START DB processing. NOOPEN is the default except when the database has EEQEs or the database was previously authorized but not allocated. If the database has EEQEs or was previously authorized but not allocated, then the database will be opened until the NOOPEN keyword is specified. Operators need to be aware of the results of issuing this command. Check the return code in message DFS0488I.

Specifying /START DB OPTION(OPEN) can cause the randomizer routine or the selection partition routine to load.

Restriction: The OPEN parameter is not supported:
- On a HALDB master. The command will result in message DFS0488I RC=57, but if only one HALDB master database was specified in the command, the online partition structures of the HALDB master database will be rebuilt if needed. No rebuild will be attempted if more than one database name is listed in the command.
- In an RSR environment.
- On an XRF alternate.
- With keywords ALL, GLOBAL, or NODBALLOC.

Usage notes

For a DEDB, the /START DB command also causes any unloaded randomizer, which is specified in the DBD source, to be reloaded.

You can use the UPD DB START(ACCESS) AREA(*) command to allocate the areas of a DEDB.

When you issue the CREATE DB command for a DEDB, you must specify RESIDENT(Y). DEDBs are always resident. The residency default for the CREATE DB command is non-resident.

After a DEDB is created, you must start it before you can use it in a similar manner as when you add a DEDB or change a DEDB using the online change (OLC) process. After an OLC process that either adds a DEDB or changes a DEDB, the DEDB and the areas must be started. The /START DB command or the UPD DB command will start the database and load the randomizer. The /START AREA command or the UPD AREA command will start the individual areas.

When the name that is specified is for a partition, the action that is taken to allocate data sets varies. The action varies depending on the status of the master database and whether the DMB for the master database is already loaded. If a partition has the /DBRECOVERY command called against it, then the partition cannot be allocated by the /START command even if the DBALLOC keyword is specified. The partition can be allocated by the /START command if the OPEN keyword is used or if the database has EEQEs. The partition will get allocated at first reference if the partition cannot be allocated by the /START command, the OPEN keyword is not used, and the database does not have EEQEs. This applies to partitioned PHDAM and PHIDAM database types.

PHDAM or PHIDAM partitions that had the /DBRECOVERY command issued against them, cannot be allocated with the /START DB DBALLOC command. However, if the partition databases have EEQEs, or the OPEN keyword is used with the /START DB command, or the databases were previously authorized but

not allocated, the partitions can be allocated. If you do not use the OPEN keyword, the PSINDEX partition is allocated and the PHDAM and PHIDAM partitions are not allocated until they are authorized.

When a /START DB command is issued for all transactions whose processing program has access to a successfully started database, the USTOPPED attribute will be reset and any messages on the suspend queue for that transaction will be transferred to the normal queue.

If one or more of the named databases requires backout or recovery, and the database is registered in DBRC, the database that require backout or recovery is dropped from the command and the remainder of the databases continue processing. If the database is not registered in DBRC, specify the NOBACKOUT keyword to inform IMS that it does not have to attempt to execute the failed backout again.

To start a HIDAM database, both the index and the data area DBD names must be specified. If a backout failure occurred for this database, the /START command causes the backout to be attempted again.

If a DL/I database is in use by a batch message processing (BMP) region when the /START DB command is issued, a DFS0565I message is returned to the master terminal, the command is ignored for the database indicated in the message, and processing continues for any other databases that are specified in the command. The master terminal operators must wait until the BMP program ends before they reissue the /START DB command to start the database indicated in the DFS0565I message. However, if allocation of a database fails during the scheduling or processing of a BMP region, the /START DB command for that database is processed while the BMP region is still active, and a DFS0565I message is not issued.

For databases on an RSR tracking subsystem, /START DB is used to resume tracking for those databases that were stopped by a tracking subsystem processing. The /START DB command also starts online forward recovery (OFR) for those databases that are not current with mainline tracking.

The /START DB ALL command causes message DFS0488I to be issued. All databases that were defined during the system definition process will be started if possible.

The output of the /START DB command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate to the database type and the command completion.
- Full-function database messages: DFS030I, DFS132, DFS160, DFS216, DFS0402I, DFS0488I, DFS0740I, DFS1407, DFS2026, DFS3317I, DFS3318I, DFS3320I, DFS3325I, DFS3465I, DFS3466I
- Fast Path database messages: DFS140I, DFS666, DFS3062

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB except for the /START DB command and the UPDATE DB START(ACCESS) command. A HALDB partition is

not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

For HALDB databases, IMS tracks partition statuses and master database statuses separately. For example, a partition can be stopped, but its master database can be started. Alternatively, the partition can be started, but its master database can be stopped. Before opening, authorizing, or scheduling a partition, IMS always examines the status of the partition and the master database. If either the partition or the master database has a status that prevents the action, IMS does not perform the action.

Each partition has the access limitations of both itself and its master database. For example, if the master database has an access intent of read (READ) and one of its partitions has an access intent of update (UPD), the partition cannot be updated. Alternatively, if the master database has an access intent of update (UPD) and one of its partitions has an access intent of read (READ), the partition cannot be updated. Similar considerations apply to other statuses that affect access limitations, such as being stopped or locked.

Exception: If the HALDB master database has update access (UPD), the partitions can have an access intent of exclusive (EXCL), exceeding the access of the master.

Commands that are issued with a partition name affect only the status of the partition. Commands that are issued against the master database affect only the status of the master database. Therefore, a start of a master database does not update the status of its partitions. If the partitions are stopped, they remain stopped. When a HALDB partition is explicitly stopped, it must be explicitly started again. The type-1 commands with the keyword ALL, type-2 commands with NAME(*), and commands against a HALDB master do not change the STOPPED (shown as STOACC, STOSCHD, or STOUPDS on QUERY DB) and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is UPDATE DB STOP(ACCESS) on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. However, the stopped status is only set in the master database. If a QUERY DB command is issued, only the HALDB master displays a status of STOACC (each HALDB partition does not display STOACC unless it was itself stopped). If a UPDATE DB STOP(ACCESS) command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

Restrictions:
- The /START DB command cannot be processed against a HALDB partition on an IMS system while HALDB Online Reorganization (OLR) is running against that partition on the same IMS system.
- The /START DB ACCESS=UP command cannot be issued against a HALDB master while OLR is reorganizing any of its partitions.
- While the database is being quiesced, this command cannot be processed successfully.

The /START DB command is not allowed for a database that is marked bad with the `NOTINIT-48-REPOCHGLIST` reason code because the IMS change list processing is not complete for the database or the change list processing failed.

Equivalent IMS type-2 commands

The following table shows variations of the /START DB command and the IMS type-2 commands that perform similar functions.

Table 275. Type-2 equivalents for the /START DB command

Task	/START DB command	Similar IMS type-2 command
Starts a database and change access intent of the database.	/START DB ACCESS	UPDATE DB START(ACCESS) SET(ACCTYPE())
Starts a database.	/START DB *dbname*	UPDATE DB NAME(*dbname*) START(ACCESS)

Examples

The following are examples of the /START DB command:

Example 1

TSO SPOC input:
```
STA DB BANKATMS BANKTERM BANKLDGR BE3ORDER
```

TSO SPOC output:
```
SYS3      DFS0488I  STA COMMAND COMPLETED. DBN= BANKATMS RC=04
SYS3      DFS0488I  STA COMMAND COMPLETED. DBN= BANKTERM RC=04
SYS3      DFS0488I  STA COMMAND COMPLETED. DBN= BANKLDGR RC=04
SYS3      DFS0488I  STA COMMAND COMPLETED. DBN= BE3ORDER RC=08
IMS3      DFS0488I  STA COMMAND COMPLETED. DBN= BANKATMS RC=04
IMS3      DFS0488I  STA COMMAND COMPLETED. DBN= BANKTERM RC=04
IMS3      DFS0488I  STA COMMAND COMPLETED. DBN= BANKLDGR RC=04
IMS3      DFS0488I  STA COMMAND COMPLETED. DBN= BE3ORDER RC=08
```

OM API input:
```
CMD (STA DB BANKATMS BANKTERM BANKLDGR BE3ORDER )
```

OM API output:
```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.197 21:59:29.210362</statime>
<stotime>2002.197 21:59:30.213238</stotime>
<staseq>B7EFC01B367FAE02</staseq>
<stoseq>B7EFC01C2B576D8F</stoseq>
<rqsttkn1>USRT005 10145929</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
</ctl>
<cmderr>
<mbr name="SYS3    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>00000014</rc>
<rsn>00005050</rsn>
<rsntext>Command processing error</rsntext>
</mbr>
<mbr name="IMS3    ">
```

```
<typ>IMS      </typ>
<styp>DBDC     </styp>
<rc>00000014</rc>
<rsn>00005050</rsn>
<rsntext>Command processing error</rsntext>
</mbr>
</cmderr>
<cmd>
<master>SYS3    </master>
<userid>USRT005 </userid>
<verb>STA </verb>
<kwd>DB            </kwd>
<input>/STA DB BANKATMS BANKTERM BANKLDGR BE3ORDER   </input>
</cmd>
<msgdata>
<mbr name="SYS3     ">
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKATMS RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKTERM RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKLDGR RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BE3ORDER RC=08</msg>
</mbr>
<mbr name="IMS3     ">
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKATMS RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKTERM RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BANKLDGR RC=04</msg>
<msg>DFS0488I  STA COMMAND COMPLETED. DBN= BE3ORDER RC=08</msg>
</mbr>
</msgdata>
</imsout>
```

Explanation: The START DB command is routed from OM to the two active IMS systems - SYS3 and IMS3. The response from both IMS systems is returned to OM. The databases BANKATMS, BANKTERM, BANKLDGR, and BE3ORDER are started at both IMS systems.

Example 2

Entry ET:
```
 /START DB TREEFARM
```

Response ET:
```
DFS058I  (time stamp) START COMMAND IN PROGRESS
DFS0488I START COMMAND COMPLETED. DBN=TREEFARM RC=0.
```

Explanation: Database TREEFARM is started.

Related concepts:

Maintaining global information for databases, DEDB areas, and transactions (System Administration)

Related reference:

"UPDATE DB command" on page 1038

Related information:

DFS2406I (Messages and Codes)

DFS2838I (Messages and Codes)

/START DC command

Use the /START DC command to open the VTAM ACBs (if MNPS for XRF is used, then both the MNPS and APPLID ACBs are opened) if they are not already open, to enable logons to IMS, and to enable optional transaction manager functions such as IMS generic resource support and IMS persistent sessions support for RNR.

- IMS generic resource support: The defined VTAM generic resource group is joined with GRSNAME in the IMS or DCC PROCLIB members.
- IMS persistent sessions support for RNR: Session activity that was suspended because of a major outage is resumed or terminated, as appropriate, if the RNR option was specified in the DFSDCxxx PROCLIB member.

If the /START DC command is issued on an XRF alternate system that is using MNPS, the command will only open the APPLID ACB. The MNPS ACB is not opened until XRF takeover processing.

Subsections:
- "Environment"
- "Syntax"
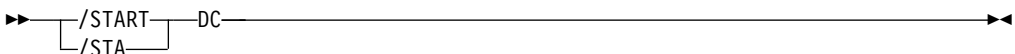
Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 276. Valid environments for the /START DC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
DC	X		X

Syntax

```
►►──┬─/START─┬──DC───────────────────────────────────────────────►◄
    └─/STA───┘
```

/START DESC command

Use the /START DESC command to define the LU62 descriptors from DFS62xxx PROCLIB member to IMS.

The full PROCLIB member name needs to be specified. The PROCLIB member must start with DFS62.

Subsections:
- "Environment"
-

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 277. Valid environments for the /START DESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
DESC	X		X

Syntax

```
►►──┬─/START─┬──DESC──proclibmember──────────────────────────────────────────►◄
    └─/STA───┘
```

/START ISOLOG command

The /START ISOLOG command indicates that the RSR tracking subsystem is to initiate a request for isolated log data from the isolated log sender at the active site.

/START ISOLOG can only be entered on an RSR tracking subsystem.

/START ISOLOG may be needed when:
- The active system failed and was not restarted, and the OLDS at the time of failure was closed using the Log Recovery utility.
- Batch jobs or utilities running at the active were unable to send logs.
- Permanent error status is shown for gaps in the output of a /DISPLAY TRACKING STATUS command. If the log problem at the active has been corrected, use /START ISOLOG to initiate retry.

Successful completion of syntax checking of the /START ISOLOG command results in the DFS058 START COMMAND COMPLETED message, although processing of the command continues asynchronously.

Subsections:
- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 278. Valid environments for the /START ISOLOG command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
ISOLOG	X	X	X

Syntax

```
►►──┬─/START─┬──ISOLOG─────────────────────────────────────────────────────────►◄
    └─/STA───┘
```

/START LINE command

The /START LINE command makes communication lines that are idle and in a stopped or process stopped state available for use. It also terminates any conversations that are active on the line.

Subsections:
- "Environment"
- "Syntax"
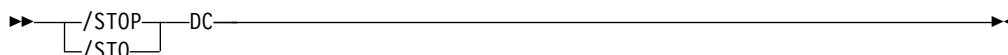- "Usage notes"
- "Examples" on page 849

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 279. Valid environments for the /START LINE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
LINE	X		X
PTERM	X		X

Syntax
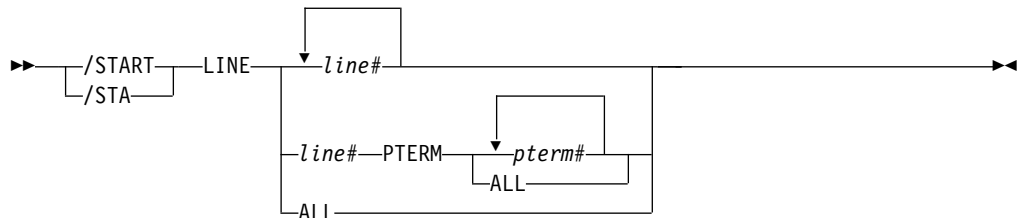
Usage notes

All terminals are removed from looptest mode, MFSTEST mode, or exclusive mode, and any terminals in signon status are reset. The /START LINE command without the PTERM keyword enables the line again (resulting in a non-VTAM LOPEN macro).

If the line must be enabled again, /START LINE must be entered prior to any /START LINE PTERM command.

The /START LINE PTERM command makes one or more physical terminals available for use. The physical terminals are removed from response mode, test mode, looptest mode, MFSTEST mode, preset destination mode, or exclusive mode, and resets terminals in signon status. If IMS encounters a stopped and idle line when processing /START LINE PTERM, it restarts the line. Any inoperable components are marked as operable.

To activate I/O operations for a LINE, LINE PTERM, or NODE without altering the status of the associated/specified terminals, use /RSTART instead of /START.

If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and processing continues for the rest of the parameters. This happens for /START LINE if:

- The specified line is already started or is not idle.
- Any terminals on the line had conversations that could not be canceled (because an application program is scheduled).
- The specified line could not be started because of IMS internal processing.
- The DD statement is missing from the IMS execution JCL.

The /START LINE command no longer resets preset mode, test mode, and response mode since these statuses are no longer significant and therefore are not kept after a /START LINE command or restart.

Examples

The following are examples of the /START LINE command:

Example 1 for /START LINE command

Entry ET:
```
/START LINE 4 PTERM 1, 2
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL STARTED
```

Explanation: Physical terminals 1 and 2 on line 4 are started.

Example 2 for /START LINE command

Entry ET:
```
/START LINE 4,5,6,7,8,9,10,11
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL STARTED
```

Explanation: Lines 4, 5, 6, 7, 8, 9, 10, and 11 are started.

Example 3 for /START LINE command

Entry ET:
```
/START LINE 4 5 6 700
```

Response ET:
```
DFS058I  START COMMAND COMPLETED EXCEPT LINE 5 700
```

Response RT:
```
DFS059I  TERMINAL STARTED
```

Explanation: Lines 4 and 6 are started. The /DISPLAY LINE command can be used to determine why line 5 did not start successfully. (700 is an invalid line number.)

/START LTERM command

Use the /START LTERM command to specify the logical terminals to be started and to reset the QLOCK state. (QLOCK indicates that the LTERM is locked from sending any further output or from receiving input that can create additional output for the same LTERM until the state is reset by a specific request received on the session.) /START LTERM is rejected for remote logical terminals.

Subsections:
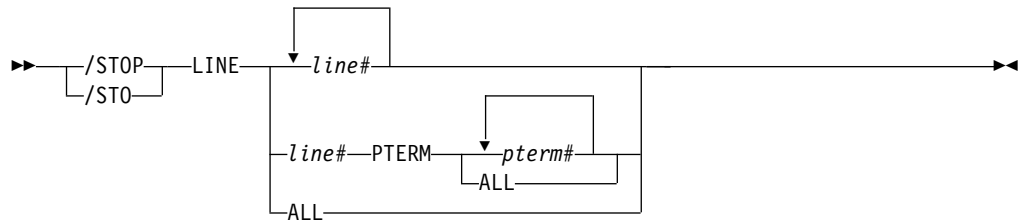- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 280. Valid environments for the /START LTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
LTERM	X		X

Syntax
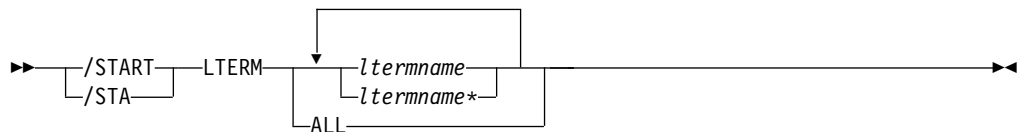
Usage notes

The LTERM keyword is only effective for existing LTERMs.

The LTERM parameter can be generic where the generic parameter specifies LTERMs that already exist.

If global resource information is kept in Resource Manager, the /START LTERM command allows messages to be queued to the LTERM from anywhere in the IMSplex and the change is reflected both in Resource Manager and in the local IMS system.

Examples

The following is an example of the /START LTERM command:

Entry ET:
```
/START LTERM APPLE, TREE, FRUIT
```

Response ET:

```
DFS058I  START COMMAND COMPLETED
```

Response RT:

```
DFS059I  TERMINAL STARTED
```

Explanation: Logical terminals APPLE, TREE, and FRUIT are started.

/START LUNAME command

Use the START LUNAME command to specify the LU name that is to be started.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 852

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 281. Valid environments for the START LUNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
INPUT	X		X
LUNAME	X		X
OUTPUT	X		X

Syntax
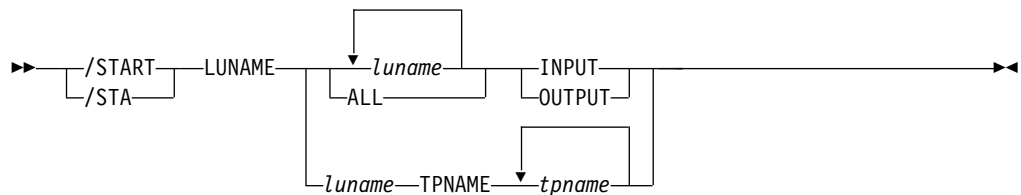
Keywords

The following keywords are valid for the /START LUNAME command:

INPUT
 Specifying the keyword INPUT starts an luname for any input and synchronous outbound activities. Specifying the parameter ALL with the keyword INPUT causes all future LU 6.2 inbound and synchronous output activities to be started as well.

OUTPUT
 Specifying the keyword OUTPUT starts an luname for asynchronous outbound

activities. Specifying the parameter ALL with the keyword OUTPUT causes all future LU 6.2 outbound asynchronous activities to be started as well.

Usage notes

Specifying neither INPUT nor OUTPUT is the same as specifying both INPUT and OUTPUT. The LU name is started for any input and both synchronous and asynchronous outbound activities. Specifying the parameter ALL in this case also causes the start of all future LU 6.2 inbound activities, outbound synchronous, and asynchronous activities.

A network-qualified LU name is optional for the LUNAME keyword. If the LU name is not network-qualified and no TP name is specified, all the network-qualified LU names whose LU name matches the LU name specified are also started.

/START LUNAME TPNAME starts a particular tpname of an luname. The keyword OUTPUT is the default for this command.

If the specified resource does not exist, a structure is created to retain the status.

Related reference:

Command keywords and their synonyms (Commands)

/START MADSIOT command

Use the /START MADSIOT command to specify the MADS I/O timing function.

Subsections:
- "Environment"
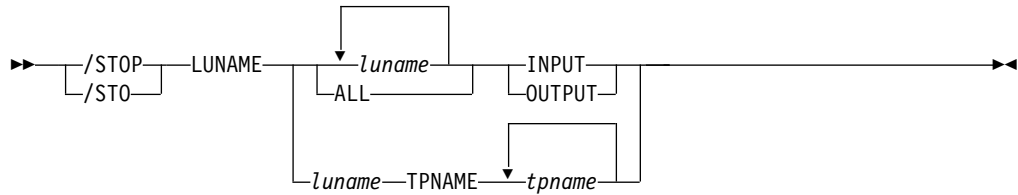- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 282. Valid environments for the /START MADSIOT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
MADSIOT	X	X	

Syntax

```
►►──┬─/START─┬──MADSIOT──────────────────────────────────────────────►◄
    └─/STA───┘
```

Usage notes

The /START MADSIOT command is valid only after the long busy handling function is disabled for a link failure, a structure failure, or a rebuild failure. When

the /START MADSIOT command completes normally, one of the two following messages is returned to the operator's console:

- `DFS1728E START MADSIOT COMMAND FAILED RSN=rrr`
- `DFS1727I MADSIOT TIMING FUNCTION RESUMED SUCCESSFULLY`

The purpose of this command is to enable MADS I/O Timing function. If MADS I/O Timing list structure is not defined in DFSVSMxx, the command will be rejected. If MADS I/O Timing function is already enabled, the command will be ignored. If MADS I/O Timing function is not enabled and all sharing partners successfully connect to MADS I/O Timing list structure on the coupling facility, the command will complete successfully; if any sharing partners fails to connect to MASD I/O Timing list structure, the command will fail.

The output of the /STA MADSIOT command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS0023I, DFS0007I, DFS1270I, DFS1727I, DFS1552A, DFS1728E

/START MSNAME command

Use the /START MSNAME command to specify the logical link path that is to be started. The MSNAME keyword can be generic.

Subsections:

- "Environment"
- "Syntax"
- "Examples" on page 854

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 283. Valid environments for the /START MSNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
MSNAME	X		X

Syntax
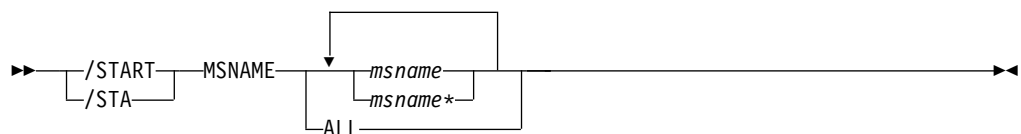
Examples

The following is an example of the /START MSNAME command:

Entry ET:
```
/START MSNAME CHICAGO
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Explanation: A logical link path associated with the name CHICAGO is started.
Related reference:
"UPDATE MSNAME command" on page 1211

/START NODE command

Use the /START NODE command to allow IMS to accept logons from, or initiate logons to, terminals attached to VTAM without actually initiating a session with the terminal. When a stopped node is started by using the /START NODE command, the terminal can log on to IMS or IMS can initiate a session with the terminal by using the /OPNDST NODE command.

The /START NODE command operates only if the node is disconnected, idle, and stopped. In addition to resetting the STOPPED status, the command resets MFSTEST mode, exclusive mode, DEACT status, and conversational mode by terminating the conversations.

Subsections:
- "Environment"
- "Syntax"
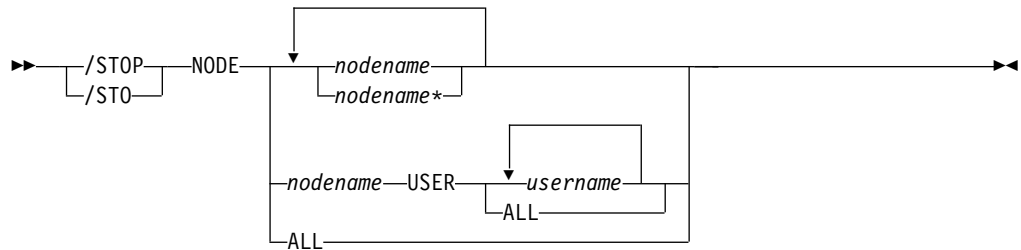- "Usage notes" on page 855
- "Examples" on page 856

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 284. Valid environments for the /START NODE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
NODE	X		X
USER	X		X

Syntax

```
>>--+--/START--+--NODE--+--+-------------+--------------------------------------->><
    +--/STA----+        |  | <--------+  |
                        |  +-nodename--+  |
                        |  +-nodename*-+  |
                        |                 |
                        |                              <---------+
                        +-nodename--USER--+--+-username-+--+-----+
                        |                    +-ALL------+
                        +-ALL--------------+
```
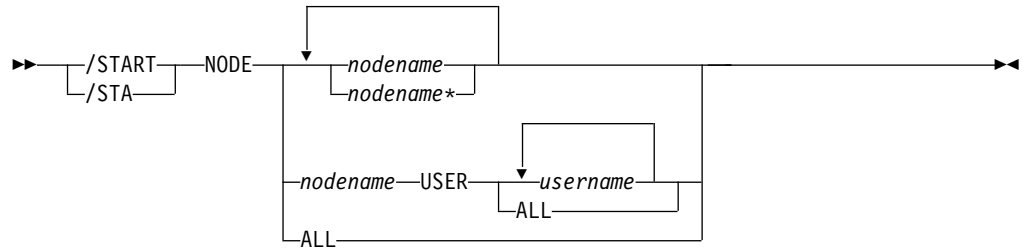
Usage notes

The /START NODE command is valid for existing nodes only (including any temporary nodes present in the system).

There is no need to issue a /DEQUEUE command for Fast Path messages. When the response mode is reset for a NODE or USER destination of Fast Path messages by either a /STOP NODE or a /STOP USER command, the Fast Path messages are discarded when the accompanying /START NODE or /START USER command is issued. In this case, IMS writes a X'67D0' Subtype 11 trace log record.

The /START NODE nodename USER username command applies to ISC sessions only, and it is used to start a half session allocated to USER *username* for NODE *nodename*. The USER keyword when used with the NODE keyword affects the specified half-session. When the USER keyword is omitted, all half-sessions of the specified node are affected.

Restrictions for using NODE and USER parameters together:
- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- /START NODE USER commands are valid for ISC, LUP, and 3600 nodes only.

The NODE parameter can be generic if the USER keyword is not present. The generic parameter specifies nodes that already exist.

The /START NODE command no longer resets test mode and preset mode, because these statuses are no longer significant and therefore no longer carried across logon or restart. MFSTEST mode (at the node level) and exclusive mode are still reset.

If global resource information is kept in Resource Manager, the /START NODE command allows a node to log on to any IMS in the IMSplex and resets MFSTEST mode and exclusive mode. If the node no longer has significant status, it is deleted from Resource Manager.

If a node in conversational mode receives the message, DFS058I START COMMAND COMPLETED EXCEPT when a /START NODE command is issued, it is possible that the conversation is INUSE by some other process. This is a temporary condition; you can reissue the /START NODE command.

Examples

The following is an example of the /START NODE command:

Entry ET:
```
/START NODE HARRY
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Explanation: The physical terminal that is associated with the node HARRY is started.

/START OLDS command

The /START OLDS command indicates that either a previously stopped OLDS is to be started or that IMS is to add a new OLDS log data set.

If a new OLDS is being added, olds# is an OLDS identifier that is defined by the DFSMDA macro specification. If in dual mode, both primary and secondary OLDSs are started. olds# must be 00-99.

When using /START OLDS, an OLDS must be defined in the DFSMDA macro, even if it is allocated in JCL.

Subsections:
- "Environment"
- "Syntax"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 285. Valid environments for the /START OLDS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
OLDS	X	X	X

Syntax

```
►►──┬─/START─┬──OLDS──olds#──────────────────────────────────────────────►◄
    └─/STA───┘
```

Examples

The following is an example of the /START OLDS command:

Entry ET:
```
/START OLDS 09
```

Response ET:

```
DFS058I  START COMMAND IN PROGRESS
```

Explanation: OLDS data set DFSOLP09 (DFSOLS09) will be started for logging.

/START OTMA command

Use the /START OTMA command to cause IMS to join the z/OS cross-system coupling facility (XCF) group for the IMS Open Transaction Manager Access (OTMA).

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 286. Valid environments for the /START OTMA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
OTMA	X		X

Syntax

```
►►─┬─/START─┬─OTMA─┬────────┬─────────────────────────────────────────────►◄
   └─/STA───┘      └─NOCHECK─┘
```
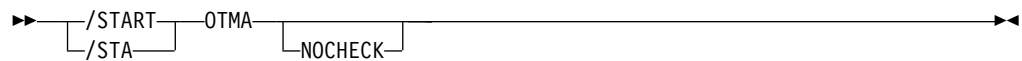
Usage notes

/START OTMA command processing is as follows:
1. IMS joins the XCF group.
2. Following a successful Client-Bid, IMS sends an ACK message to the OTMA client.
3. IMS begins sending all Commit-then-Send (commit mode 0) output messages to the OTMA client.

The NOCHECK option specifies that the command is not recovered during emergency restart for OTMA.

Examples

The following is an example of the /START OTMA command:

Entry ET:

```
/STA OTMA
```

Response ET:

```
DFS2360I 14:02:53 XCF GROUP JOINED SUCCESSFULLY. SYS3
DFS058I 14:02:53 START COMMAND COMPLETED   SYS3
DFS996I *IMS READY*  SYS3
```

/START PGM command

Use the /START PGM command to specify the application program that is to be started.

This command also clears the indicator preventing a program from scheduling when I/O prevention has not completed. The integrity of a GSAM database residing on DASD can be affected if I/O prevention has not been done on a failing active system.

/START PGM does not start a CPI Communications driven transaction program.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands"
- "Examples" on page 859

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 287. Valid environments for the /START PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
PGM	X	X	X

Syntax
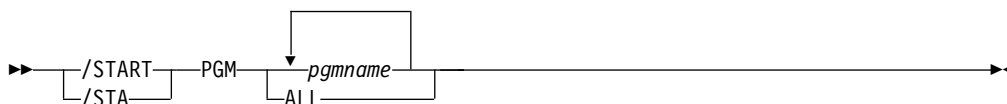
Usage notes

The /START PGM command is not allowed for a program that is marked bad with the NOTINIT-57-REPOCHGLIST reason code because the IMS change list processing is not complete for the program or the change list processing failed.

Equivalent IMS type-2 commands

The following table shows variations of the /START PGM command and the IMS type-2 commands that perform similar functions.

Table 288. Type-2 equivalents for the /START PGM command

Task	/START PGM command	Similar IMS type-2 command
Starts program scheduling.	/START PGM *pgmname*	UPDATE PGM NAME(*pgmname*) START(SCHD)

Examples

The following are examples of the /START PGM command:

Example 1 for /START PGM command

Entry ET:
```
/START PROGRAM ALL
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Explanation: All application programs are started.

Example 2 for /START PGM command

Entry ET:
```
/START PROGRAM APPLETRE
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Explanation: Application program APPLETRE is started.

Related reference:

"UPDATE PGM command" on page 1259

/START REGION command

Use the /START REGION command to specify the set of message processing region JCL to be passed to z/OS. If no member name is specified, the default member name is used

Subsections:
- "Environment"
- "Syntax" on page 860
- "Usage notes" on page 860
- "Examples" on page 861

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 289. Valid environments for the /START REGION command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X

Table 289. Valid environments for the /START REGION command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
LOCAL	X	X	
REGION	X	X	X

Syntax
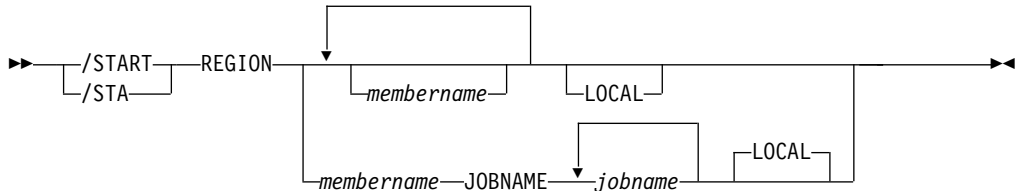
Usage notes

IMS dependent regions of the same type (MPP, BMP, or IFP) can share a PROCLIB member containing the startup JCL for the type of region. Use the JOBNAME or LOCAL keywords of the /START REGION command to enable IMS to set (or override) the IMS ID for the dependent region to match the IMS ID of the IMS that processes the command.

Restriction: The JCL for the region to be started must include the IMSID= execution parameter.

Use the JOBNAME keyword to override the job name on the JOB statement of the default or specified JCL member for a dependent region.

If you specify the LOCAL keyword, IMS overrides the symbolic IMSID parameter in the JCL of the default or specified member. LOCAL is the default if you specify the JOBNAME keyword.

When the LOCAL or JOBNAME keywords are specified on the /START REGION command, the PROCLIB member must be a job that runs a procedure to start the dependent region. The procedure cannot be an instream procedure. For example, suppose that the /START REGION command is entered in one of the following formats:

```
/START REGION member_name LOCAL
/START REGION member_name JOBNAME job_name
/START REGION member_name JOBNAME job_name LOCAL
```

In these instances, *member_name* is a job that runs a procedure to start the dependent region and has the following format:

```
//job_name JOB ... (parameters)...
//     EXEC proc_name,
//       IMSID=xxxx
```

The operator can start more dependent regions than were specified in the IMS system definition or the EXEC parameter, up to 999. A request to start more regions than the system-definition value (but less than or equal to 999) might be rejected if resources are not available.

The /START REGION command is not mirrored on the XRF alternate subsystem. You must enter this command on the alternate subsystem if you want it to affect the alternate subsystem.

The /START REGION command allows the IMS catalog update task to be stopped. The IMS catalog update task is used to make changes to the IMS catalog and activate those changes in the online IMS system. When issuing the /START REGION command against the IMS catalog update task the results to the IMS catalog will be unpredictable when an update is in progress. The IMS catalog update task performs several operations that extend outside the scope of the IMS database, updating the RECONs, creating data sets, creating programs and databases. The /START REGION command prevents the task from executing the abnormal termination logic. It is your responsibility to cleanup.

Examples

The following are examples of the /START REGION command:

Example 1 for /START REGION command

Entry ET:
```
/START REGION
```

Response ET:
```
DFS058I  START COMMAND IN PROGRESS
```

Response ET:
```
DFS551I IFP|MESSAGE|BATCH  REGION XXXXXXXX STARTED. ID=yy TIME=zzzz
CLASSES=xxx,xxx,xxx,xxx
```

Explanation: One message region or batch region (ID=yy) is started at TIME=zzzz. The transactions associated with the classes listed in the response can now be scheduled.

Example 2 for /START REGION command

Entry ET:
```
/START REGION IMSWT000
```

Response ET:
```
DFS058I  START COMMAND IN PROGRESS
```

Explanation: The JCL stored as member IMSWT000 is used to start the spool SYSOUT utility for the data sets associated with the spool line corresponding to the IMSWT000 procedure.

Example 3 for /START REGION command

Entry ET:
```
/START REGION MEMABC
```

Response ET:
```
DFS058I  START COMMAND IN PROGRESS
```

Response ET:

```
DFS551I IFP|MESSAGE|BATCH  REGION XXXXXX STARTED. ID=yy TIME=zzzz
CLASSES=xxx,xxx,xxx,xxx
```

Explanation: The JCL stored as member XXXXXX is used to start a message processing region or batch message processing region with the classes specified by the EXEC statement parameters in MEMABC.

Related reference:

➡ PRDR=name parameter for procedures (System Definition)

/START RTC command

Use the /START RTC command to specify the Fast Path routing codes to be activated and to allow transactions associated with the routing codes to be processed.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands"
- "Examples" on page 863

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 290. Valid environments for the /START RTC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
RTC	X		X

Syntax
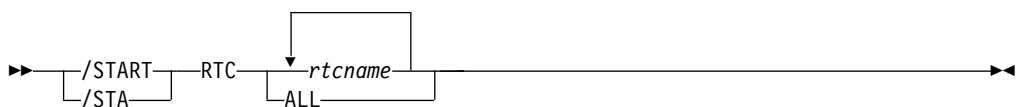
Usage notes

The /START RTC command is not allowed for a routing code that is marked bad with the `NOTINIT-07-REPOCHGLIST` reason code because the IMS change list processing is not complete for the routing code or the change list processing failed.

Equivalent IMS type-2 commands

The following table shows variations of the /START RTC command and the IMS type-2 commands that perform similar functions.

Table 291. Type-2 equivalents for the /START RTC command

Task	/START RTC command	Similar IMS type-2 command
Starts queuing to a Fast Path routing code.	/START RTC *rtcname*	UPDATE RTC NAME(*rtcname*) START(Q)

Examples

The following is an example of the /START RTC command:

Entry ET:
```
/START RTCODE ALL
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Explanation: All the Fast Path routing codes are activated. Transactions associated with these routing codes can now be processed.

Related reference:

"UPDATE RTC command" on page 1308

/START SB command

Use the /START SB command to dynamically allow sequential buffering. This command does not affect sequential buffering applications scheduled before this command was issued.

Subsections:
- "Environment"
- "Syntax"
- "Examples" on page 864

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 292. Valid environments for the /START SB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SB	X	X	

Syntax

```
►►──┬─/START─┬──SB──────────────────────────────────────────────────►◄
    └─/STA───┘
```

Examples

The following is an example of the /START SB command:

Entry ET:
```
/START SB
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Entry ET:
```
/DISPLAY POOL DBAS
```

Response ET:
```
SEQUENTIAL BUFFERING:  STATUS = NOT-STOPPED
MAX     N.A.  FREE  N.A.   CURR   160K  HIGH  320K
DATABASE BUFFER POOL: SIZE   67584
REQ1      0 REQ2    0 READ     0 BISAM     0 WRITES     0
KEYC      0 LCYL    0 PURG     0 OWNRR     0 ERRORS 00/00
DATABASE BUFFER POOL: BSIZE  12288
RRBA     0 RKEY    0 BFALT    0 NREC  0 SYN PTS     0
NMBUFS  29 VRDS    0  FOUND    0 VWTS  0 ERRORS 00/00
DATABASE BUFFER POOL: BSIZE 356352
RRBA     0 RKEY    0 BFALT    0 NREC  0 SYN PTS     0
NMBUFS  29 VRDS    0  FOUND    0 VWTS  0 ERRORS 00/00
*86253/104547*
```

Explanation: Sequential buffering is started.

/START SERVGRP command

Use the /START SERVGRP command to start communications between the entering service group and the service group at the other site in an Remote Site Recovery (RSR) complex.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes" on page 865

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 293. Valid environments for the /START SERVGRP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SERVGRP	X	X	X

Syntax

```
►►──┬─/START─┬──SERVGRP──────────────────────────────────────────►◄
    └─/STA───┘
```

Usage notes

If the subsystem is not currently identified to the Transport Manager Subsystem (TMS), an attempt to identify precedes an attempt to connect to the other subsystem. /START SERVGRP is supported on an active IMS subsystem and an RSR tracking subsystem.

The /START SERVGRP command is not normally needed for an active subsystem because the logger normally attempts to identify to TMS at each OLDS switch to establish connections with the other subsystem. However, the operator might want to trigger this process between OLDS switches; for instance, if a network outage between the active and tracking sites has been repaired and the operator does not want to wait until the next OLDS switch to re-establish communications.

Successful completion of the syntax checking of the /START SERVGRP command results in the DFS058 START COMMAND COMPLETED message, although processing of the command continues asynchronously.

Examples

The following is an example of the /START SERVGRP command:

Entry ET (at the active site):
```
/START SERVGRP
```

Response ET (to the active subsystem):
```
DFS058 START COMMAND COMPLETED
```

Explanation: Communications between the subsystem at the active site and the subsystem at the RSR tracking site are started.

/START SLDSREAD command

The /START SLDSREAD command indicates whether IMS is enabled to retrieve records from both a system log data set (SLDS) and OLDS or OLDS only. The default is that SLDSREAD is enabled.

Subsections:
- "Environment"
- "Syntax" on page 866

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 294. Valid environments for the /START SLDSREAD command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SLDSREAD	X	X	X

Syntax

```
►►──┬─/START─┬──SLDSREAD─────────────────────────────────────────►◄
    └─/STA───┘
```
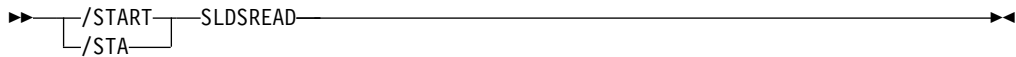
/START SUBSYS command

Use the /START SUBSYS command to specify the external subsystem to which IMS is to connect.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 867

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 295. Valid environments for the /START SUBSYS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SSM	X	X	X
SUBSYS	X	X	X

Syntax

```
►►──┬─/START─┬──SUBSYS──┬─▼─subsysname─┬────────────────────────►◄
    └─/STA───┘          ├─ALL──────────┤
                        └─SSM──ssmname──┘
```
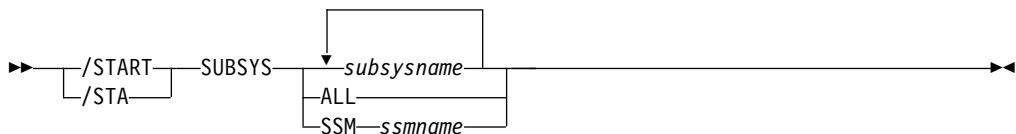
Keywords

The following keywords are valid for the /START SUBSYS command:

SSM

Allows external subsystem connection processing to occur even though the option was not requested when IMS was started.

The syntax and usage of the SSM keyword are the same as the SSM= EXEC parameter that can be specified on the IMS startup JCL. The SSM keyword is followed by a 1 to 4 character identifier. IMS concatenates the SSM identifier to the IMSID to create an SSM PROCLIB member name. The SSM PROCLIB member is then used for external subsystem processing.

The SSM keyword is not valid if either of the following conditions apply:
- The SSM= keyword is specified in the EXEC parameters of the startup JCL.
- The /START SUBSYS SSM command has been previously issued.

Usage notes

This command can also be used to dynamically reconfigure existing subsystem definitions. The installation can start IMS with the subsystem PROCLIB member defining one subsystem. The PROCLIB member can then be changed or added to. The operator can then /STOP the existing subsystem connections or only the one that has changed. By issuing the /START SUBSYS command, IMS will pick up the new or changed definitions and attempt to connect to those subsystems.

The /START SUBSYS ALL command connects IMS to all external subsystems. Also, the SSM keyword can be used with the /START SUBSYS command.

If the subsystem connection was abnormally terminated, IMS puts the connection in a stopped state. In this instance, the /START command must be used to reestablish the connection.

Examples

The following is an example of the /START SUBSYS command:

Entry ET:
```
/START SUBSYS ABC
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Explanation: IMS has established a connection to the requested subsystem. It is likely that an external subsystem (not CCTL) connection message will be received at this time. If this is not the case, the /DISPLAY command can be used.

/START SURV command

Use the /START SURV command in an XRF environment to start the operation of the IMS surveillance function.

Subsections:
- "Environment"
- "Syntax" on page 868
- "Usage notes" on page 868

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 296. Valid environments for the /START SURV command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
SURV	X		X

Syntax

```
>>--+--/START--+--SURV--+--ALL--+-------------------------------------------><
    +--/STA----+        +--LNK--+
                        +--LOG--+
                        +--RDS--+
```

Usage notes

When surveillance is on for a function, potential failures of the active system are detected. Based on information from surveillance, the alternate system either requests a takeover or informs the operator of the potential failure. The following are the surveillance functions to be started:

ALL   Same as specifying LNK, RDS, and LOG.

LNK   ISC link.

LOG   System log.

RDS   Restart data set.

The surveillance function is generally started during IMS system definition by using the /START SURVEILLANCE control statement. ALL is the default.

/START THREAD command

Use the /START THREAD command to specify the set of message processing region JCL to be passed to z/OS. If no member name is specified, the default member name is used.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes" on page 869

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 297. Valid environments for the /START THREAD command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
THREAD	X	X	

Syntax

```
>>--+--/START--+--THREAD--+-----------------+---------------------------><
    +--/STA----+          +---<-----------+ |
                          +--membername---+
```

Usage notes

The /START THREAD command is used only for DEDB utility regions. BMP regions are started by JCL. CCTL threads are started automatically at connection: first to the MINTHREAD value, and later (on demand) to MAXTHREAD value.

/START TMEM command

Use the /START TMEM command to cause IMS to send an Open Transaction Manager Access (OTMA) command to OTMA clients to request that input resume for the specified transaction membername. IMS then resumes sending output to the OTMA client.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Examples" on page 871

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 298. Valid environments for the /START TMEM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
TMEM	X		X
TPIPE	X		X

Syntax

A:
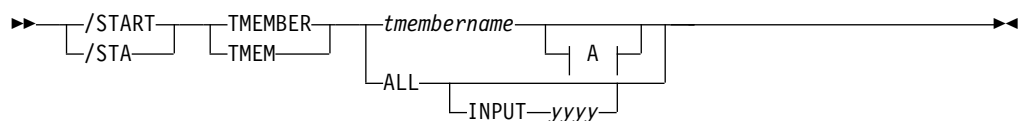
Keywords

The following keywords are valid for the /START TMEM command:

INPUT

Specifies the maximum number of concurrent input messages from the

OTMA member. OTMA monitors the growth of the input messages from the member. A DFS1988W warning message is sent to the system console to indicate that the input messages reached 80% of the limit and the message is issued every 5% thereafter. When the maximum is reached, a DFS1989E error message is sent to the console. Any subsequent OTMA input message from this member will be rejected with the OTMA sense code X'30'.

When the number of input messages (transaction instance blocks (YTIBs)) reaches the maximum, the /DISPLAY command shows FLOOD status under the user-status column. FLOOD status is relieved when the input messages are processed and recede to 50% or less of the maximum value or a /START TMEMBER INPUT command is issued with a higher maximum value specified.

The input value is a 1- to 5-digit decimal integer. Valid values are 200 - 65000 or 0. If the value is 1 - 200, it is treated as 200. If the value is over 65000, a value of 65000 is used. If the value is 0, OTMA deactivates the message flood detection.

If this keyword is specified with the ALL keyword, the following is true:

- The value can be one- to five-digit decimal integer 0 - 99999. If the value is 1 - 99999, it indicates the maximum number of input messages from all the OTMA members that can be waiting at the same time to be processed. To avoid too many OTMA unprocessed or waiting messages or TIBs in the system, this value is used as a global flood threshold to protect IMS from getting storage-related abends. A non-zero value activates the global flood control for OTMA for the system. If you specify a value of 0, OTMA deactivates the global flood monitoring and control. If the value is 1 - 200, it is treated as 200.

- When a non-zero global flood limit is specified, it activates the global flood control. When the global flood limit is reached, OTMA rejects all the new input transactions from all the members, including synchronous program switch requests from the internal OTMA member DFSYICAL. A DFS3429E error message is sent to the console and MTO. A protocol message with the command type set to X'3C' is sent to all the OTMA members with "unavailable for work" status. When OTMA members receive this new status, they can choose to take corrective action when this degraded condition occurs. One example would be the rerouting of all the new transaction requests from one IMS system to another. The /DISPLAY OTMA command of this degraded system also shows "SERVER-FLOOD" in the user status of the OTMA server member.

- The DFS3428W warning message is issued to IMS MTO and system console along with OTMA protocol messages that reflect a warning status to all the OTMA member clients when the global flood limit has reached 80%. And this message is issued every 5% thereafter until the global message limit is reached. This global flood status is relieved when the input messages in the system are processed and the number of total transaction instance blocks (TIB) in the system is reduced to 50% of the global limit or less. A DFS0793I is sent to IMS MTO and system console along with OTMA protocol messages that reflect a good status to all the OTMA members.

- When this command is not issued and there is no global limit in OTMA client descriptor, the default global flood processing takes place in the

system and the default global flood limit is set to 10000 to protect the
system. The default action points of the OTMA global flood process
issue:

- Warning message DFS3428W starting at 80% of the flood limit and
every 5% thereafter.
- Warning message DFS4388W when 100% is reached.
- Relief message DFS0793I when the flood is relieved and back to 50%.

Also, the OTMA protocol message that reflect a warning or good status
is sent to all the OTMA members.

TPIPE

When it is used with the /START command, TPIPE causes IMS to send an
OTMA command to its OTMA clients to request that the input resume for
the specified transaction pipe. IMS then resumes sending output to the
OTMA client. If the member specified is a super member, output is
resumed for the super member's transaction pipe, but no OTMA command
is sent. If the member specified is a regular member whose hold queue
output is managed by a super member, IMS resumes output for the
specified member's transaction pipe and it also resumes output for the
super member's transaction pipe. An OTMA command is sent to the
regular member's OTMA client. Output is only resumed on the IMS that
processes the command. If output cannot be resumed for both the regular
member's transaction pipe and the super member's transaction pipe, it is
not resumed for either transaction pipe. The DFS058I COMMAND
COMPLETED EXCEPT message is issued with the name of the regular
member for which output could not be resumed.

Restriction: If a transaction pipe has a resynchronization pending status,
IMS does not start the transaction pipe.

TIMEOUT

Sets the timeout interval for acknowledgments to commit-then-send (CM0)
and send-then-commit (CM1) output messages.

After OTMA delivers a send-then-commit output message that used
synclevel=confirm or synclevel=syncpt, OTMA expects an acknowledge
(ACK or NAK) message from the OTMA client, such as IMS Connect. If
the acknowledge message is not received by IMS within the specified
timeout value, OTMA will take the timeout action to abort the transaction.
This prevents IMS dependent region from getting the WAITSYNPT or
WAIT/RRS status for long time.

The TIMEOUT keyword also applies to acknowledgments for transaction
messages sent to a remote IMS system by way of an IMS-to-IMS TCP/IP
connection. When the timeout interval expires, the unacknowledged
message is rerouted to the default timeout queue, DFS$$TOQ.

The timeout interval must be 0 - 255 seconds, with a default value of 120
seconds. When 0 is specified, the timeout function is disabled.

Examples

The following are examples of the /START TMEM command:

Example 1 for /START TMEM command

Entry ET:

```
/STA TMEMBER CLIENT1 TPIPE TPIPESY
```

Response ET:

```
DFS058I 15:39:40 START COMMAND COMPLETED    SYS3
DFS996I *IMS READY*  SYS3
```

Example 2 for /START TMEM command

Entry ET:

```
 /START TMEMBER HWS1 INPUT 2000
```

Response ET:

```
DFS058I START COMMAND COMPLETED
```

Explanation: The maximum concurrent input message count for the OTMA
member HWS1 has been set to 2000. Based on the number specified, IMS OTMA
will monitor the growth of the input messages to prevent a message flood
condition.

/START TRAN command

Use the /START TRAN command to specify the transactions to be started.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes" on page 873
- "Equivalent IMS type-2 commands" on page 874
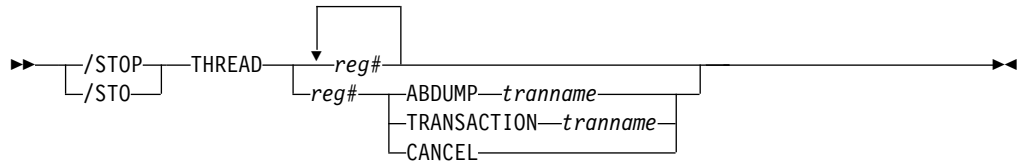- "Examples" on page 874

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 299. Valid environments for the /START TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
AFFINITY	X		X
CLASS	X		X
TRAN	X		X

Syntax

```
         ┌─────────────────────┐
         │    ┌───────────┐     │
►►─┬─/START─┬─TRAN─┼─┬─tranname──┼─┤───────────────────────────►◄
   └─/STA───┘      │ └─tranname*─┘ │
                   │   ┌──────────┐│
                   ├─┬─▼─tranname─┼┴─AFFINITY──┬─┤
                   │ └─ALL────────┘            │
                   │          ┌──────┐         │
                   └─CLASS──┬──▼─cls#─┴─────────┘
                            └─ALL────┘
```

Usage notes

The /START TRAN ALL CLASS *cls#* command causes all transactions associated with the specified class to be started.

If a transaction that has messages on the suspend queue is started, the suspend queue associated with the transaction is automatically transferred to the normal queue.

The TRAN keyword can be generic where the generic parameter specifies transactions that already exist.

You can use the /START TRAN *tranname* AFFINITY command to start a local affinity transaction queue in a shared-queues environment. Starting a local affinity transaction queue in a shared-queues environment causes this IMS system to be notified when transaction messages that have an affinity name of this IMS system's IMSID or RSENAME (XRF) appended to the transaction name are on the shared queues for processing. This affinity registration is in addition to the normal transaction registration (registration with the transaction name appended with blanks).

When a transaction is started with affinity, the affinity registration status cannot be removed and the QUERY TRAN command always shows the affinity status. The following /STOP TRAN *tranname* command therefore stops both normal and affinity registration. The subsequent /START TRAN *tranname* command (with or without the AFFINITY keyword) always performs both normal and affinity registration.

The affinity status is lost across IMS cold starts. If the DFSMSCE0 user message routing exit is used to set local affinity for an input transaction message in a shared-queues environment and the IMS control region is stopped and subsequently cold started, you must issue the /START TRAN AFFINITY command or the UPDATE TRAN NAME(*tranname*) START(SCHD) OPTION(AFFIN) command. These commands register the transaction with affinity status so that the messages will be processed.

At the end of cold start processing, when shared-queues informs are done, the inform for the transaction is done without affinity to the local IMSID. Because local affinity is set by the exit and is not part of the transaction definition, it is not maintained across a cold start. As a result, the message with local affinity cannot be scheduled. The /START TRAN *tranname* AFFINITY command issues an inform for the transaction with affinity to the local IMSID.

The /START TRAN *tranname* AFFINITY command does not support generic transaction names of ALL in the parameter.

In a shared-queues environment, the /START TRAN command will result in IMS registering interest for the transaction, which indicates that the transaction can be scheduled at the IMS. A /START TRAN ALL command does not register transactions that are already registered to CQS.

The /START TRAN command is not allowed for a transaction that is marked bad with the `NOTINIT-15-REPOCHGLIST` reason code because the IMS change list processing is not complete for the transaction or the change list processing failed.

Equivalent IMS type-2 commands

The following table shows variations of the /START TRAN command and the IMS type-2 commands that perform similar functions.

Table 300. Type-2 equivalents for the /START TRAN command

Task	/START TRAN command	Similar IMS type-2 command
Starts a transaction.	/START TRAN	UPDATE TRAN NAME(*tranname*) START(Q,SCHD,SUSPEND)

Examples

The following are examples of the /START TRAN command:

Example 1 for /START TRAN command

Entry ET:
```
/START TRAN ALL CLASS 6
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Explanation: All transactions associated with class 6 are started.

Example 2 for /START TRAN command

Entry ET:
```
/START TRAN PIT, SEED
```

Response ET:
```
DFS058I  START COMMAND COMPLETED
```

Explanation: Transactions PIT and SEED are started.

Example 3 for /START TRAN command

Entry ET:
```
/START TRAN APOL12 AFFINITY
```

Response ET:
```
DFS058I START COMMAND COMPLETED
```

Explanation: Transaction APOL12 is started with affinity.

Related reference:

"UPDATE TRAN command" on page 1326

/START TRKARCH command

Use the /START TRKARCH command to indicate that the RSR tracking subsystem is to initiate a request to start the automatic archiving of the tracking log data sets.

This keyword allows the user to start automatic archive after it has been terminated following archive data set full conditions.

Successful completion of the syntax checking of the /START TRKARCH command results in the DFS058I START COMMAND COMPLETED message, although processing of the command continues asynchronously.

Subsections:
- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 301. Valid environments for the /START TRKARCH command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
TRKARCH	X	X	X

Syntax

```
►►──┬─/START─┬──TRKARCH──────────────────────────────────────────────►◄
    └─/STA───┘
```

/START USER command

Without the NODE keyword, USER specifies the ISC user or the dynamic user to start. The USER parameter can be generic where the generic parameter specifies users that already exist.

Subsections:
- "Environment"
- "Syntax" on page 876
- "Usage notes" on page 876

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 302. Valid environments for the /START USER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
USER	X		X

Syntax

```
>>--+--/START--+--USER--+--->--+--username---+--+------------------------->< 
    +--/STA----+        ^      +--username*--+  |
                        |                       |
                        +----------------------+
                        +--ALL------------------+
```

Usage notes

The /START USER command applies only to users that are stopped and signed off. The /START USER command also terminates any active conversations before starting the user.

The /START USER command removes the user from MFSTEST mode and exclusive mode. If the USER structure is temporary and the status conditions that caused the creation of the structure have been reset, the temporary user is deleted at the next simple checkpoint.

For ISC users, the user is started and made available for allocation.

For dynamic users, the user is started and made available for signon. If the dynamic user was in either Fast Path input or output response mode, and the Fast Path input or output response mode is reset by the /STOP USER and /START USER commands issued in sequence, IMS writes a X'67D0' Subtype 11 trace log record.

The /START USER command no longer removes the user from test mode, and preset mode. MFSTEST mode can now be associated with the node and with the user. MFSTEST mode (at the user level) and exclusive mode are still reset. The other statuses are no longer significant and therefore not carried across signon or restart.

If global resource information is not kept in Resource Manager, the /START USER command allows a user to signon to the local IMS. If global resource information is kept in Resource Manager, the /START USER command allows a user to signon to any IMS in the IMSplex.

If a user in conversational mode receives the message, DFS058I START COMMAND COMPLETED EXCEPT when a /START USER command is issued, it is possible that the conversation is INUSE by some other process. This is a temporary condition; you can reissue the /START USER command.

Examples

The following is an example of the /START USER command:

Entry ET:

```
/DISPLAY USER IMSUS01 IMSUS02
```

Response ET:
```
  USER    ENQCT DEQCT  QCT
  IMSUS01     0     0     0 STOPPED
  IMSUS02     0     0     0 STOPPED
  *91091/111727*
```

Entry ET:
```
R 38,/START USER IMSUS01
```

Response ET:
```
DFS058I 11:19:05 START COMMAND COMPLETED
```

Entry ET:
```
/DISPLAY USER IMSUS01 IMSUS02
```

Response ET:
```
USER    ENQCT DEQCT   QCT
IMSUS01     0     0     0
IMSUS02     0     0     0 STOPPED
*91091/113038*
```

Entry ET:
```
/START USER APPLE*
```

Response ET:
```
DFS3633 11:19:35 GENERIC PARAMETER RESOURCES NOT FOUND, NO ACTION TAKEN
```

/START VGR command

The /START VGR command causes the IMS subsystem to join a VTAM generic resource group. The command is rejected if the VTAM ACB is closed (usually the result of a /STOP DC command).

Subsections:
- "Environment"
- "Syntax" on page 878
- "Keywords" on page 878

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 303. Valid environments for the /START VGR command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
GRSNAME	X	X	X
VGR	X		X

Syntax

```
►►──┬─/START─┬──VGR──┬──────────────────────────────────────────┬──►◄
    └─/STA───┘       └─GRSNAME──generic_resource_name─┘
```

Keywords

The following keywords are valid for the /START VGR command:

The GRSNAME keyword allows you to specify the generic resource name if the IMS subsystem does not have one. The IMS subsystem already has a generic resource name if it has the GRSNAME= keyword specified on its EXEC statement.

The GRSNAME is ignored if GRSNAME= was specified on the EXEC statement, or if it was already specified on a previous /START VGRS command. If GRSNAME= was not specified on the EXEC statement or any previous /START VGRS command, then all VTAM sessions must be terminated prior to executing the /START command with GRSNAME specified.

/START WADS command

The /START WADS command indicates that either a previously stopped WADS is to be started or that IMS is to add a new WADS to the pool of available WADSs.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 879

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 304. Valid environments for the /START WADS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
WADS	X	X	X

Syntax

```
►►──┬─/START─┬──WADS──wads#──────────────────────────────────────────►◄
    └─/STA───┘
```

Keywords

The following keywords are valid for the /START WADS command:

wads#
> If a new WADS is being added, *wads#* is a WADS identifier that is defined by the DFSMDA macro specification. *wads#* must be 0-9.

Usage notes

When using the /START WADS command, a WADS must be defined in the
DFSMDA macro, even if it is allocated in JCL.

/START XRCTRACK command

The /START XRCTRACK command results in calls to the log router to initiate or
terminate XRC tracking. It is only valid on a tracking IMS system.

Subsections:
- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 305. Valid environments for the /START XRCTRACK command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/START	X	X	X
XRCTRACK	X	X	X

Syntax

```
►►──┬─/START─┬──XRCTRACK──────────────────────────────────────────►◄
    └─/STA───┘
```

Chapter 24. /STOP commands

The /STOP commands stop the sending, receiving, or queuing of output messages to a particular communication line, terminal, user, or logical path.

You can also use these commands to stop the scheduling or queuing of messages containing a specific transaction code, the execution of a specific program, or the use of a given database.

For VTAM nodes, the currently connected terminal is disconnected. All further logons are rejected until the node is the subject of a /START or /RSTART command.

The /STOP command validity checks all parameters entered by the terminal operator. If an error is detected on parameters that are independent of one another, only the invalid parameters are indicated as being in error and the /STOP command processes the rest of the parameters.

The /STOP command can be used to reset conditions previously established by the /START, /RSTART, /PSTOP, /PURGE, or /MONITOR commands.

These commands can be issued to an IMSplex using the Batch SPOC utility.

/STOP ADS command

The /STOP ADS command specifies the area data set to be closed and deallocated.

The AREA is not stopped as long as at least one data set in the AREA remains open. /STOP ADS is rejected if the specified ADS is the last data set available in the AREA.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 306. Valid environments for the /STOP ADS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
ADS	X	X	

Syntax

```
►►──┬─/STOP─┬──ADS─ddname───────────────────────────────────►◄
    └─/STO──┘
```

Usage notes

Although the /STOP ADS command has no option of LOCAL/GLOBAL, if the DEDB area is shared at the block level, the response is the same as if GLOBAL were specified.

The output of the /STOP ADS command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS140I, DFS0488I, DFS0666I, DFS1407I, DFS3720I, DFS3721I, DFS3771I

/STOP APPC command

The /STOP APPC command instructs IMS to stop scheduling transactions from LU 6.2 devices. The /STOP APPC command can be used in a transient stopped state. It causes remote LU 6.2 devices to receive a sense code of TP_Not_Available_No_Retry. This is likely to lead to further attempts to access IMS.

Subsections:
- "Environment" on page 883
- "Syntax" on page 883
- "Keywords" on page 883
- "Usage notes" on page 883

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 307. Valid environments for the /STOP APPC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
APPC	X		X
CANCEL	X	X	X

Syntax

```
►►──┬─/STOP─┬──APPC──────────────────────────────────────────────────────►◄
    └─/STO──┘    └─CANCEL─┘
```
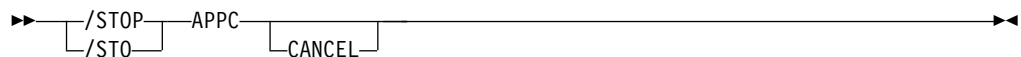
Keywords

The following keyword is valid for the /STOP APPC command:

CANCEL

Causes APPC/MVS to initiate a shutdown request when a long stopped period is anticipated, for example, at the end of the day's processing. All remote LU 6.2 devices receive a sense code of TP_Not_Available_No_Retry. The remote LU 6.2 devices stop trying to access this application.

The /STOP APPC CANCEL command sets the desired status to CANCEL according to responses from APPC/MVS. If the desired status is DISABLED, then IMS rejects /STOP APPC CANCEL when it is entered.

Note: The sense code returned to the LU 6.2 remote device for an incoming ATTACH to a stopped APPC/IMS system is determined by APPC/MVS, and it might differ from release to release. In general, the remote LU 6.2 application should wait for a period of time after rejection before any attempts to reestablish a session with IMS.

Usage notes

The /STOP APPC command sets the desired status to STOPPED. The current status is set to STOPPED or FAILED according to the response from APPC/MVS.

/STOP AREA command

The /STOP AREA command specifies that the data sets associated with this area are closed.

Subsections:

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 308. Valid environments for the /STOP AREA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
AREA	X	X	
GLOBAL	X	X	
LOCAL	X	X	
NOPFA	X	X	

Syntax
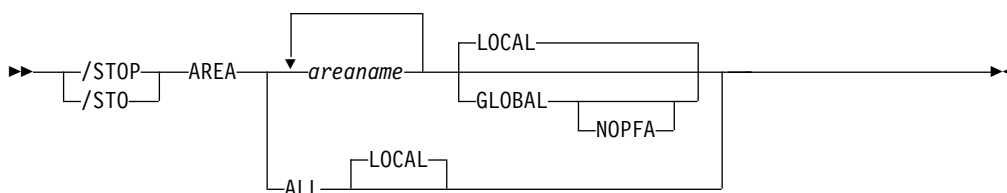
Keywords

The following keywords are valid for the /STOP AREA command:

GLOBAL

Specifies when an IRLM is active and that the command applies to all subsystems sharing the database or area.

The GLOBAL keyword and the ALL parameter are mutually exclusive. If both keywords are specified, the command is rejected. The GLOBAL keyword requires that IRLM be active. If IRLM is not active, the command is rejected. DBRC is informed that the database or area has been stopped and will update the RECON data set to indicate the stopped condition.

The GLOBAL command is processed by the IMS system where the command was initiated. This system will process the command locally and then request IRLM NOTIFY to route and process the command on sharing IMS systems.

If global DB or AREA status is maintained, the global status maintained in RM is also updated. The global status is set to STOPPED.

If the command is entered from OM API, the global status is updated by the command master IMS. If the command is not entered from OM API, the IMS that initiated the GLOBAL command updates the global status in RM

You must specify if the global area status must be maintained in RM. You can specify this during IMS initialization in either the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSAREA(Y)). You can also change it dynamically using the UPDATE IMS SET(PLEXPARM(GSTSAREA(Y))) command. If you do not specify that the global area status is to be maintained, then the GLOBAL keyword is processed as in prior releases, and the global status is not updated.

If global status in RM is successfully updated, message DFS0988I for RSRCTYPE=AREA is issued. If global status is not successfully updated, message DFS3308I is issued, indicating RM failure, and no command response lines are generated. Any RM error is traced to the OCMD trace table. Users can issue a QRY DB STATUS(GLOBAL) command to obtain the global status of the resources in RM.

The X'594C' log record for DEDB areas includes both global status and global command time stamp.

The GLOBAL keyword is not supported on an RSR tracking subsystem.

If the GLOBAL keyword on a command is entered from an OM API, the command should only be routed to one IMS system in the IMSplex. The IMS that receives the command from OM will make DBRC calls to update the RECON with GLOBAL status. It will also request IRLM NOTIFY to route and process the command on sharing IMS systems, and then process the command locally.

Messages produced on the NOTIFIED systems will appear only on the system console and will not be routed back to the OM API which originally entered the command.

If multiple IMS systems have been explicitly specified in the route list, the master IMS system will process the command as described previously. However, the non-master IMS systems, to which OM routes the command, will reject the command with the following return and reason code listed in the following table:

Table 309. Return and reason code for GLOBAL keyword issued from the OM API

Return code	Reason code	Meaning
X'00000004'	X'00001000'	The command contained the GLOBAL keyword and was routed to more than one IMS system in the IMSPLEX. The non-master IMS systems will reject this command when OM routes the command to them. The master IMS system will process this command and use IRLM NOTIFY to route and process the command on the non-master IMS systems. See the discussion under the GLOBAL keyword.

LOCAL
Specifies that the command only applies to the subsystem in which the command is entered. This command does not affect any other subsystem sharing the database or area. LOCAL is the default.

NOPFA
Specifies that DBRC is not notified that the database or area has changed status. You can use this keyword when you need to authorize the database for use after it is offline, for example, for offline utilities. By using this keyword, DBRC does not prevent further authorizations for the database or area. NOPFA can be specified only with the GLOBAL keyword.

Recommendation: Before restarting the database or area, issue this command without the NOPFA keyword to inform DBRC of the change in status for the database or area.

Usage notes

Use the /DISPLAY AREA command to determine if the area is stopped or closed. If the area is stopped, the area must be made available using the /START AREA command. In z/OS, all the data sets are deallocated. If the system processes a /STOP AREA command during HSSP processing, the area will be released after the current commit processing completes. Any image copy option in effect at /STOP time can affect the continued system operation. All virtual storage option (VSO) DEDB areas that are being stopped and that are in a z/OS data space are removed from the data space and updates are written out to DASD.

The output of the /STOP AREA command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. For commands that specify GLOBAL, only the command master returns the asynchronous messages to OM. When a command is processed with the LOCAL keyword, all IMS systems are able to return the asynchronous messages to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS140I, DFS170I, DFS0488I, DFS0666I, DFS1407I, DFS3062I, DFS3342I, DFS3720I, DFS3824I

/STOP AREA is not supported on an RSR tracking subsystem.

While the database is being quiesced, this command cannot be processed successfully.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP AREA command and the IMS type-2 commands that perform similar functions.

Table 310. Type-2 equivalents for the /STOP AREA command

Task	/STOP AREA command	Similar IMS type-2 command
Stops an area.	/STOP AREA *areaname*	UPDATE AREA NAME(*areaname*) STOP(SCHD)

Examples

The following is an example of the /STOP AREA command:

Entry ET:
```
/STOP AREA DB1AREA0 DB1AREA1
```

Response ET:
```
DFS058I  STOP COMMAND IN PROGRESS
DFS0488I  STOP COMMAND COMPLETED.  AREA=DB1AREA0
DFS0488I  STOP COMMAND COMPLETED.  AREA=DB1AREA1
```

Explanation: The DEDB areas DB1AREA0 and DB1AREA1 are stopped for processing.

Related concepts:

➡ Maintaining global information for databases, DEDB areas, and transactions (System Administration)

Related reference:

"UPDATE AREA command" on page 1007

/STOP AUTOARCH command

The /STOP AUTOARCH command specifies that automatic archiving is to be stopped.

Subsections:
- "Environment"
- "Syntax"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 311. Valid environments for the /STOP AUTOARCH command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
AUTOARCH	X	X	X

Syntax

```
►►──┬─/STOP─┬──AUTOARCH──────────────────────────────────────────────►◄
    └─/STO──┘
```
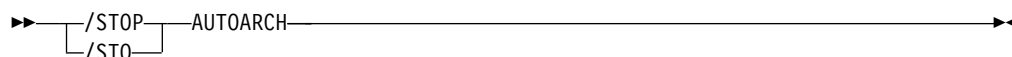
Examples

The following is an example of the /STOP AUTOARCH command:

Entry ET:
```
/STOP AUTOARCH
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: Automatic archiving is stopped.

/STOP BACKUP command

The /STOP BACKUP command terminates the alternate system in an XRF environment. This command must be entered on the alternate system. The ABDUMP keyword results in a dump of the alternate system.

Subsections:
- "Environment" on page 888
- "Syntax" on page 888

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 312. Valid environments for the /STOP BACKUP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
ABDUMP	X	X	X
BACKUP	X		X

Syntax

```
►►──┬─/STOP─┬──BACKUP──────────────────────────────────►◄
    └─/STO──┘        └─ABDUMP─┘
```
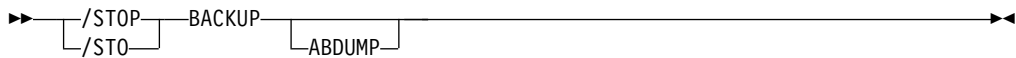
/STOP CLASS command

The /STOP CLASS command prevents further scheduling of application programs for the designated class.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 889

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 313. Valid environments for the /STOP CLASS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
CLASS	X		X

Syntax

```
                      ┌──────────┐
►►──┬─/STOP─┬──CLASS──┬─▼─cls#─┬──────────────────────────►◄
    └─/STO──┘         └─ALL────┘
```

Usage notes

All regions currently handling transactions assigned to the specific class are allowed to run until the limit count is reached (MPPs) or the input queue contains no more messages (BMPs and MPPs).

The region is not allowed to wait for the next message (wait-for-input mode).
Instead a QC status code (no more messages) is returned to the application (MPPs).

If the region is already scheduled and waiting for the next message (wait-for-input
mode) when the command is entered, the region is notified and a QC status code
is returned to the application. (MPPs).

A batch message processing region (BMP) scheduled against wait-for-input (WFI)
transactions returns a QC status code (no more messages) for /PSTOP REGION,
/DBD, /DBR, or /STA commands only.

Examples

The following is an example of the /STOP CLASS command:

Entry ET:
```
/STOP CLASS 3
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: No further scheduling of application programs for class 3 transactions
occurs. All message processing programs currently handling class 3 transactions
are allowed to run until the processing limit count is reached or the input queue
contains no more messages.

/STOP DATAGRP command

The /STOP DATAGRP command specifies groups of DL/I databases, Fast Path
DEDBs, and Fast Path areas to be stopped.

Data groups are logical groupings of databases and areas; they enable simplified
command processing for databases and areas. You define a data group in the
RECON data set by using the INIT.DBDSGRP command with parameters
GRPNAME and DBGRP.

DATAGRP is not valid on RSR tracking subsystems.

Subsections:
- "Environment"
- "Syntax" on page 890
- "Usage notes" on page 890
- "Equivalent IMS type-2 commands" on page 890

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 314. Valid environments for the /STOP DATAGRP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DATAGRP	X	X	

Command / Keywords	DB/DC	DBCTL	DCCTL
LOCAL	X	X	

Syntax

```
►►──┬─/STOP─┬──DATAGRP──datagroupname───────────────────────────────►◄
    └─/STO──┘                        └─LOCAL─┘
```

Usage notes

During /STOP DATAGRP processing, all virtual storage option (VSO) DEDBs that are in a z/OS data space are removed from the data space and updates are written out to DASD.

Recommendation: Although you can use DBDS groups as well as DB groups for this command, you should use DB groups whenever possible to eliminate the overhead of converting the DBDS group to a DB group.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP DATAGRP command and the IMS type-2 commands that perform similar functions.

Table 315. Type-2 equivalents for the /STOP DATAGRP command

Task	/STOP DATAGRP command	Similar IMS type-2 command
Stops a data group.	/STOP DATAGRP *datagrpname*	UPDATE DATAGRP NAME(*datagrpname*) STOP(SCHD)

Related reference:

/STOP DB command

The /STOP DB command prevents subsequently scheduled programs from accessing the database, without affecting currently scheduled programs or closing the database.

Subsections:

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 316. Valid environments for the /STOP DB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DB	X	X	
GLOBAL	X	X	
LOCAL	X	X	
NOPFA	X	X	

Syntax
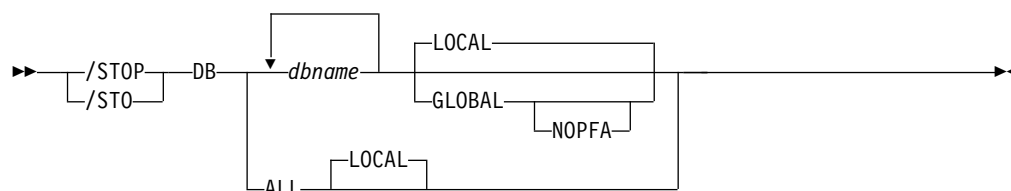
Usage notes

If the database is a DEDB or MSDB, programs using the database will not be scheduled. For other databases, the programs will still be scheduled but a call against the database will result in either a 3303 pseudoabend, or a BA status code, if the INIT call was issued.

The /STOP DB command deletes the randomizer routine from memory.

If the database is stopped after the region is scheduled, the region is not allowed to wait for the next message (wait-for-input mode). If there are no more messages available for the region to process, a QC status (no more messages) will be returned to the application (MPPs). If the region is already scheduled and waiting for the next message (wait-for-input mode) when the command is entered, the region is notified and a QC status code is returned to the application (MPPs).

A batch message processing region (BMP) scheduled against wait-for-input (WFI) transactions returns a QC status code (no more messages) for /PSTOP REGION, /DBD, /DBR, or /STA commands only.

In an IFP region, the /STOP command has no effect until the region is started again. Issuing a /STOP DB command with the ACCESS parameter is not valid for an MSBD.

The GLOBAL command is processed by the IMS system where the command was initiated. This system will process the command locally and then request IRLM NOTIFY to route and process the command on sharing IMS systems.

If global DB status is maintained, the global status maintained in RM is also updated. The global status is set to STOSCHD.

If the command is entered from OM API, the global status is updated by the command master IMS. If the command is not entered from OM API, the IMS that initiated the GLOBAL command updates the global status in RM.

If global status in RM is successfully updated, message DFS0988I for RSRCTYPE=DB is issued. If global status is not successfully updated, message DFS3308I is issued, indicating RM failure, and no command response lines are generated. Any RM error is traced to the OCMD trace table. Users can issue a QRY DB STATUS(GLOBAL) command to obtain the global status of the resources in RM.

The X'4C' log record for databases is updated to include both global status and global command time stamp.

For DBCTL, when CCTL schedules a PSB, the DBCTL thread SCHED request defines the thread as LONG or SHORT. If the database is currently scheduled to a LONG thread, the command is rejected; otherwise, the thread is allowed to complete before the database is acted upon. This results in either a commit point or transaction termination.

The output of the /STOP DB command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate to the database type and the command completion.
- Full-function database messages: DFS132, DFS160, DFS216, DFS0488I, DFS1407, DFS2026, DFS3318I, DFS3466I
- Fast Path database messages: No unique messages are returned.

See the AREA keyword for a description of the LOCAL and NOPFA keywords.

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB except for the /START DB command and the UPDATE DB START(ACCESS) command. A HALDB partition is not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

For HALDB databases, IMS tracks partition statuses and master database statuses separately. For example, a partition can be stopped, but its master database can be started. Alternatively, the partition can be started, but its master database can be stopped. Before opening, authorizing, or scheduling a partition, IMS always examines the status of the partition and the master database. If either the partition or the master database has a status that prevents the action, IMS does not perform the action.

Each partition has the access limitations of both itself and its master database. For example, if the master database has an access intent of read (READ) and one of its partitions has an access intent of update (UPD), the partition cannot be updated. Alternatively, if the master database has an access intent of update (UPD) and one of its partitions has an access intent of read (READ), the partition cannot be updated. Similar considerations apply to other statuses that affect access limitations, such as being stopped or locked.

Exception: If the HALDB master database has update access (UPD), the partitions can have an access intent of exclusive (EXCL), exceeding the access of the master.

Commands that are issued with a partition name affect only the status of the partition. Commands that are issued against the master database affect only the

status of the master database. Therefore, a start of a master database does not update the status of its partitions. If the partitions are stopped, they remain stopped. When a HALDB partition is explicitly stopped, it must be explicitly started again. The type-1 commands with the keyword ALL, type-2 commands with NAME(*), and commands against a HALDB master do not change the STOPPED (shown as STOACC, STOSCHD, or STOUPDS on QUERY DB) and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is UPDATE DB STOP(ACCESS) on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. However, the stopped status is only set in the master database. If a QUERY DB command is issued, only the HALDB master displays a status of STOACC (each HALDB partition does not display STOACC unless it was itself stopped). If a UPDATE DB STOP(ACCESS) command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

Restrictions:
- The /STOP DB command is not supported on an RSR tracking subsystem.
- The /STOP DB command cannot be processed against a HALDB partition on an IMS system while HALDB Online Reorganization (OLR) is running against that partition on the same IMS system.
- While the database is being quiesced, this command cannot be processed successfully.

The /STOP DB command is not allowed for a database that is marked bad with the `NOTINIT-48-REPOCHGLIST` reason code because the IMS change list processing is not complete for the database or the change list processing failed.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP DB command and the IMS type-2 commands that perform similar functions.

Table 317. Type-2 equivalents for the /STOP DB command

Task	/STOP DB command	Similar IMS type-2 command
Stops a database.	/STOP DB *dbname*	UPDATE DB NAME(*dbname*) STOP(SCHD)

Examples

The following are examples of the /STOP DB command:

Example 1 for /STOP DB command

Entry ET:
```
/STOP DATABASE TREEFARM
```

Response ET:
```
DFS058I  STOP COMMAND IN PROGRESS
DFS0488I  STOP COMMAND COMPLETED.  DBN=TREEFARM RC=0
```

Explanation: Database TREEFARM is stopped.

Example 2 for /STOP DB command

TSO SPOC input:

```
STO DB BANKATMS BANKTERM BANKLDGR BE3ORDER
```

TSO SPOC output:

```
SYS3      DFS0488I  STO COMMAND COMPLETED. DBN= BANKATMS RC= 0
SYS3      DFS0488I  STO COMMAND COMPLETED. DBN= BANKTERM RC= 0
SYS3      DFS0488I  STO COMMAND COMPLETED. DBN= BANKLDGR RC= 0
SYS3      DFS0488I  STO COMMAND COMPLETED. DBN= BE3ORDER RC= 0
IMS3      DFS0488I  STO COMMAND COMPLETED. DBN= BANKATMS RC= 0
IMS3      DFS0488I  STO COMMAND COMPLETED. DBN= BANKTERM RC= 0
IMS3      DFS0488I  STO COMMAND COMPLETED. DBN= BANKLDGR RC= 0
IMS3      DFS0488I  STO COMMAND COMPLETED. DBN= BE3ORDER RC= 0
```

OM API input:

```
CMD (  STO DB BANKATMS BANKTERM BANKLDGR BE3ORDER )
```

OM API output:

```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.197 22:05:21.270547</statime>
<stotime>2002.197 22:05:21.307712</stotime>
<staseq>B7EFC16AF6B13F26</staseq>
<stoseq>B7EFC16AFFC40D8C</stoseq>
<rqsttkn1>USRT005 10150521</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT005 </userid>
<verb>STO </verb>
<kwd>DB       </kwd>
<input>STO DB BANKATMS BANKTERM BANKLDGR BE3ORDER  </input>
</cmd>
<msgdata>
<mbr name="SYS3     ">
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKATMS RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKTERM RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKLDGR RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BE3ORDER RC= 0</msg>
</mbr>
<mbr name="IMS3     ">
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKATMS RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKTERM RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BANKLDGR RC= 0</msg>
<msg>DFS0488I  STO COMMAND COMPLETED. DBN= BE3ORDER RC= 0</msg>
</mbr>
</msgdata>
</imsout>
```

Explanation: The STOP command is routed from OM to the two active IMS systems - SYS3 and IMS3. The response from both IMS systems is returned to OM. The databases BANKATMS, BANKTERM, BANKLDGR, and BE3ORDER are stopped at both IMS systems.

Related concepts:

➡️ Maintaining global information for databases, DEDB areas, and transactions
(System Administration)

Related reference:

"UPDATE DB command" on page 1038

Related information:

➡️ DFS2838I (Messages and Codes)

/STOP DC command

The /STOP DC command prohibits you from logging on to VTAM and ensures
that all VTAM node sessions have terminated before IMS issues the DFS2111I
message, which says the ACB is closed.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 318. Valid environments for the /STOP DC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DC	X		X

Syntax

```
►►──┬─/STOP─┬──DC──────────────────────────────────────►◄
    └─/STO──┘
```

Usage notes

There are two ACBs if MNPS for XRF is used. If you use XRF with MNPS, both
the APPLID and MNPS ACB are closed. If you are not using XRF or using XRF
without MNPS, there is only one ACB, the VTAM ACB, which is closed. APPLID
ACB is the same as VTAM ACB.

The /STOP DC command can be used either before or after the /CLSDST NODE
or /STOP NODE command, the only difference being that logons can still occur if
the /STOP DC command is not entered. However, the command cannot start or
complete processing if the VTAM ACBs (APPLID and MNPS) are not open or the
VTAM nodes remain active. If the nodes are active, the /CLSDST NODE or /STOP
NODE command must be issued to close the nodes; in some cases, a /IDLE NODE
command can be issued to cause an OS VTAM VARY command to be issued
against any nodes that remain connected.

/STOP DESC command

The /STOP DESC command defines the LU62 descriptors from DFS62DTx
PROCLIB member to IMS.

Subsections:
- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 319. Valid environments for the /STOP DESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DESC	X		X

Syntax

```
►►──┬─/STOP─┬──DESC────────────────────────────────────────►◄
    └─/STO──┘
```
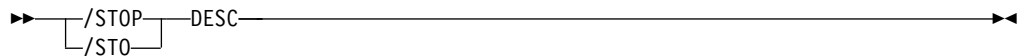
/STOP LINE command

The /STOP LINE command stops message queuing for lines and stops the sending
and receiving of messages over the lines. However, lines are not considered
stopped unless they are stopped and idle. Use /DISPLAY LINE to verify line
status.

Subsections:
- "Environment"
- "Syntax" on page 897
- "Usage notes" on page 897
- "Examples" on page 897

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 320. Valid environments for the /STOP LINE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
LINE	X		X
PTERM	X		X

Syntax

```
>>--+-/STOP-+--LINE--+---------<---------+--------------------------------------------------------><
    +-/STO--+        +--| line# |--------+
                     |
                     +--line#--PTERM--+----<----+--+
                     |                +-| pterm# |-+  |
                     |                +--ALL--------+  |
                     +--ALL-----------------------------+
```

Usage notes

The /STOP LINE PTERM command ensures that no input messages from any of the specified terminals assigned to the specified lines will be received by IMS after the command is issued.

The /STOP LINE command resets preset mode, test mode, response mode, lock pterm, lock lterm, pstop lterm, and purge lterm because these statuses are not significant and therefore are not kept after a /START LINE or restart.

Examples

The following are examples of the /STOP LINE command:

Example 1

Entry ET:
```
/STOP LINE 4,5,6,7,8,9,10,11
```

Response ET:
```
  DFS058I  STOP COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL STOPPED
```

Explanation: Lines 4, 5, 6, 7, 8, 9, 10, and 11 and their associated physical terminals are stopped.

Example 2

Entry ET:
```
/STOP LINE 4 8 900
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED EXCEPT LINE 900
```

Response RT:
```
DFS059I  TERMINAL STOPPED
```

Explanation: Lines 4 and 8 and their associated physical terminals are stopped. 900 is an invalid line number.

Example 3

Entry ET:
```
/STOP LINE 4 PTERM 1, 2
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL STOPPED
```

Explanation: Physical terminals 1 and 2 on line 4 are stopped.

/STOP LTERM command

The /STOP LTERM command specifies the LTERM that is to be stopped. The /STOP LTERM command with a logical terminal that is in a QLOCKED state does not reset the QLOCK state, but puts the LTERM in a STOPPED and QLOCKED state.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 899

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 321. Valid environments for the /STOP LTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
LTERM	X		X

Syntax
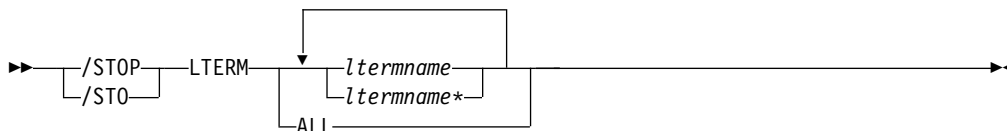
Usage notes

If IMS internally resets the QLOCK condition, the LTERM remains in a STOPPED state. (QLOCK indicates that the LTERM is locked from sending any further output or from receiving input that can create additional output for the same LTERM until the state is reset by a specific request received on the session.)

The /STOP LTERM command is rejected for remote logical terminals.

The LTERM parameter can be generic where the generic parameter specifies LTERMs that already exist.

When a specific lterm name is specified in the command, IMS creates the lterm if it does not exist and ETO is enabled.

If global resource information is kept in Resource Manager, the /STOP LTERM command stops messages from being queued to the lterm anywhere in the IMSplex and the change is reflected both in Resource Manager and in the local IMS system.

Examples

The following is an example of the /STOP LTERM command:

Entry ET:
```
/STOP LTERM APPLE, TREE, FRUIT
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED
```

Response RT:
```
DFS059I  TERMINAL STOPPED
```

Explanation: Logical terminals APPLE, TREE, and FRUIT are stopped.

/STOP LUNAME command

The /STOP LUNAME command specifies a particular LU name that is to be stopped.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes" on page 900

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 322. Valid environments for the /STOP LUNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
INPUT	X		X
LUNAME	X		X
OUTPUT	X		X

Syntax

```
 ►►──┬─/STOP─┬──LUNAME──┬──┬──▼── luname ───┬───┬──┬──INPUT──┬──────────────────────►◄
     └─/STO──┘          │  └──ALL───────────┘   │  └──OUTPUT─┘
                        │                       │
                        └─ luname ──TPNAME──┬──▼── tpname ──┬─┘
                                            └───────────────┘
```

Usage notes

Specifying the keyword INPUT with the LUNAME keyword stops an LU name for
any input and synchronous outbound activities. Specifying the parameter ALL
with INPUT causes all future LU 6.2 input and synchronous outbound activities to
be stopped as well.

Specifying the keyword OUTPUT with the LUNAME keyword stops an LU name
for any asynchronous outbound activities. Specifying the parameter ALL with
OUTPUT causes all future LU 6.2 asynchronous outbound activities to be stopped
as well.

Specifying neither INPUT nor OUTPUT is the same as specifying both INPUT and
OUTPUT. The LU name is stopped for any input, and both synchronous and
asynchronous outbound activities. Specifying the parameter ALL in this case stops
all future LU 6.2 inbound activities, synchronous and asynchronous outbound
activities.

A network-qualified LU name is optional for the LUNAME keyword. If the LU
name is not network-qualified and no TP name is specified, all network-qualified
LU names whose LU names match the LU name specified are also stopped.

The /STOP LUNAME TPNAME command stops a particular TP name of the LU
name specified. The keyword OUTPUT is the default for this command.

If the specified resource does not exist, a structure is created to retain the status.

Related reference:

➡ Command keywords and their synonyms (Commands)

/STOP MADSIOT command

The /STOP MADSIOT command allows users to disable the MADS I/O timing
function in a MADS I/O timing enabled environment.

Subsections:
- "Environment"
- "Syntax" on page 901
- "Usage notes" on page 901

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 323. Valid environments for the /STOP MADSIOT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
MADSIOT	X	X	

Syntax

```
►►──┬─/STOP─┬──MADSIOT───────────────────────────────────────────────►◄
    └─/STO──┘
```
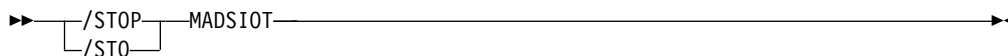
Usage notes

When /STOP MADSIOT completes normally, the following message is returned to the operator's console:

```
DFS1276I MADS I/O TIMING FUNCTION STOPPED SUCCESSFULLY
```

/START MADSIOT allows users to resume the MADS I/O timing function.

If MADS I/O Timing list structure is not defined in DFSVSMxx, the command will be rejected. If MADS I/O Timing function is already disabled, the command will be ignored. If MADS I/O Timing function is enabled and all sharing partners successfully disconnect from MADS I/O Timing list structure on the coupling facility, the command will complete successfully; if any sharing partners fails to disconnect to MASD I/O Timing list structure, the command will fail.

The output of the /STOP MADSIOT command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages as appropriate.

Fast Path messages: DFS0023I, DFS0008I, DFS1271I, DFS1276I, DFS1275E, DFS1219E

/STOP MSNAME command

The /STOP MSNAME command stops the sending of all messages (primary requests) from a terminal except those continuing a conversation. This includes all messages destined for remote transactions with the SYSID of the MSNAME and for remote logical terminals associated with this MSNAME.

Subsections:
- "Environment"
- "Syntax" on page 902
- "Examples" on page 902

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 324. Valid environments for the /STOP MSNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
MSNAME	X		X

Syntax

```
►►──┬─/STOP─┬──MSNAME──┬──┬─msname──┬──┐──────────────────────►◄
    └─/STO──┘          │  └─msname*─┘  │
                       └─ALL──────────┘
```
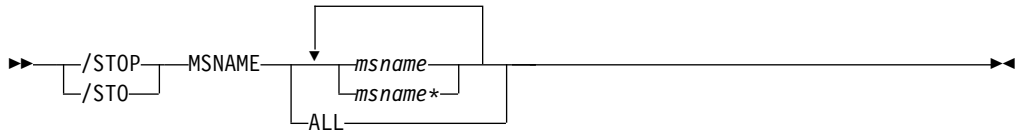
Examples

The following is an example of the /STOP MSNAME command:

Entry ET:
```
/STOP MSNAME BOSTON
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: The logical link path associated with the name BOSTON is stopped.

Related reference:

/STOP NODE command

The /STOP NODE command specifies the VTAM node to be stopped and logged off. The NODE parameter can be generic if the USER keyword is not specified and applies to nodes that already exist.

Subsections:
- "Environment"
-
-
-

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 325. Valid environments for the /STOP NODE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
NODE	X		X
USER	X		X

Syntax

```
>>--+--/STOP--+--NODE--+----------------+--------------------------><
    +--/STO---+        | <------------  |
                       +---nodename-----+
                       +---nodename*----+
                       |                                    |
                       +--nodename--USER--+-- <-------------+
                       |                  +---username------+
                       |                  +---ALL-----------+
                       +--ALL--------------------------------+
```

Usage notes

The /STOP NODE command prevents future logons until a /START NODE command is issued.

/STOP NODE without the USER keyword is supported for nodes that do not yet exist. It causes the node to be created and stopped which prevents the dynamic node from logging on. /STOP NODE without the USER keyword affects all half-sessions of the specified node.

/STOP NODE USER is valid for ISC and non-ISC nodes and users; however the user must still be allocated or signed on to the node. /STOP NODE USER for ISC nodes stops the named half-session defined in USER username for NODE nodename.

Restrictions for using NODE and USER parameters together:
- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- /STOP NODE USER commands are valid for ISC and non-ISC nodes and users.

/STOP NODE resets preset mode, test mode, lock node, lock lterm, pstop lterm, and purge lterm, because these statuses are not significant and therefore are not kept after a logon or restart. For Fast Path input response mode, you must also issue the /START NODE command before the mode is reset. The /STOP NODE command also takes other actions depending on the recovery settings for the node:

RCVYSTSN=NO

/STOP NODE acts like a /CHANGE NODE COLDSESS command for FINANCE and SLUP nodes by setting the session status to 'cold'. /STOP NODE acts like a /QUIESCE NODE command for ISC (LU6.1) nodes by initiating the shutdown and deallocating the user for the specified node. This action changes the session status to 'cold'. With these actions taken by the /STOP NODE command, the next session initiation request for this node is allowed to again attempt a session cold start (after a /START NODE command has been entered).

RCVYCONV=NO

/STOP NODE causes any IMS conversations (active and held) to be terminated. Any conversational message that is queued or being processed will have its output response message delivered asynchronously.

RCVYFP=NO

> /STOP NODE causes Fast Path status and messages to be discarded

RCVYRESP=NO

> /STOP NODE resets full-function response mode.

If global resource information is kept in Resource Manager, the /STOP NODE command sets a global stop status for the node and prevents the node from logging on anywhere in the IMSplex. If global resource information is not kept in Resource Manager, /STOP NODE creates the node, if it does not exist in an ETO environment, and sets stop status for the local node. If the node does not exist in a non-ETO environment, the /STOP NODE command is rejected.

Examples

The following is an example of the /STOP NODE command:

Entry ET:
```
/STOP NODE HARRY
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: The physical terminal associated with node HARRY is disconnected (/CLSDST) and further logons are prevented.

/STOP OLDS command

The /STOP OLDS command indicates that IMS is to stop using an OLDS log data set.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 905
- "Usage notes" on page 905
- "Examples" on page 905

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 326. Valid environments for the /STOP OLDS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
OLDS	X	X	X

Syntax

```
►►──┬─/STOP─┬──OLDS──olds#──────────────────────────────────►◄
    └─/STO──┘
```

Keywords

The following keyword is valid for the /STOP OLDS command:

***olds*#**
>   Identifies an OLDS that is defined by JCL or a DFSMDA macro and is
>   currently started. *olds*# must be 00 through 99.

Usage notes

The stopped OLDS will be dynamically deallocated when it is no longer possible
for it to be accessed for dynamic backout.

If in dual mode, both primary and secondary OLDSs are stopped. If there are only
two OLDS data sets available, or if the specified OLDS is the one currently being
used for output, the /STOP OLDS command will be rejected.

Examples

The following is an example of the /STOP OLDS command:

Entry ET:
```
/STOP OLDS 09
```

Response ET:
```
DFS058I  STOP COMMAND IN PROGRESS
DFS2500I DATASET DFSOLP09 SUCCESSFULLY DEALLOCATED
DFS3257I OLDS DEALLOCATED ON DFSOLP09
```

Explanation: The subject OLDS, DFSOLP09 (DFSOLS09), will be stopped.

/STOP OTMA command

The /STOP OTMA command causes IMS to leave the z/OS cross-system coupling
facility (XCF) group for IMS Open Transaction Manager Access (OTMA).

Subsections:
- "Environment"
- "Syntax" on page 906
- "Usage notes" on page 906
- "Examples" on page 906

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 327. Valid environments for the /STOP OTMA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
OTMA	X		X

Syntax

```
►►──┬─/STOP─┬──OTMA───────────────────────────────────────────────────────►◄
    └─/STO──┘
```

Usage notes

/STOP OTMA command processing is as follows:

1. IMS leaves the XCF group.
2. For any IMS OTMA output awaiting an ACK message, IMS aborts the message. For Commit-then-Send transactions, the output remains enqueued to the transaction pipe. For Send-then-Commit transactions, IMS aborts the transaction.

When the /STOP OTMA command is issued, it will clear or reject all the ICAL messages for all the transaction pipes. However, the /STOP OTMA command does not halt processing for the DFSYICAL tmember for synchronous program switching. To clear a wait state for a synchronous program switch request, use the /PSTOP or /STOP REGIN ABDUMP command.

Examples

The following is an example of the /STOP OTMA command:

Entry ET:
```
/STO OTMA
```

Response ET:
```
DFS2361I 14:02:05 XCF GROUP CLOSED SUCCESSFULLY. SYS3
DFS058I 14:02:06 STOP COMMAND COMPLETED   SYS3
DFS996I *IMS READY*  SYS3
```

/STOP PGM command

The /STOP PGM command specifies the application program that is to be stopped.

Subsections:
- "Environment"
- "Syntax" on page 907
- "Usage notes" on page 907
- "Equivalent IMS type-2 commands" on page 907
- "Examples" on page 907

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 328. Valid environments for the /STOP PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
PGM	X	X	X

Syntax

```
►►──┬─/STOP─┬──PGM──┬──┬──┬─pgmname─┬──────────────►◄
    └─/STO──┘       │  └───────◄────┘
                    └─ALL─────┘
```

Usage notes

The /STOP PGM command does not stop CPI Communications driven transaction programs.

The /STOP PGM command is not allowed for a program that is marked bad with the `NOTINIT-57-REPOCHGLIST` reason code because the IMS change list processing is not complete for the program or the change list processing failed.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP PGM command and the IMS type-2 commands that perform similar functions.

Table 329. Type-2 equivalents for the /STOP PGM command

Task	/STOP PGM command	Similar IMS type-2 command
Stops program scheduling.	/STOP PGM *pgmname*	UPDATE PGM NAME(*pgmname*) STOP(SCHD)

Examples

The following is an example of the /STOP PGM command:

Entry ET:
```
/STOP PROGRAM APPLETRE
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: Application program APPLETRE is stopped.
Related reference:
"UPDATE PGM command" on page 1259

/STOP REGION command

Use the /STOP REGION command to stop IMS regions, application programs, or both. /STOP REGION is not mirrored on the XRF alternate system. You must enter this command on the alternate system if you want it to affect the alternate system.

Subsections:
- "Environment" on page 908
- "Syntax" on page 908
- "Keywords" on page 908
- "Examples" on page 910

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 330. Valid environments for the /STOP REGION command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
ABDUMP	X	X	X
CANCEL	X	X	X
JOBNAME	X	X	X
REGION	X	X	X
TRANSACTION	X		X

Syntax
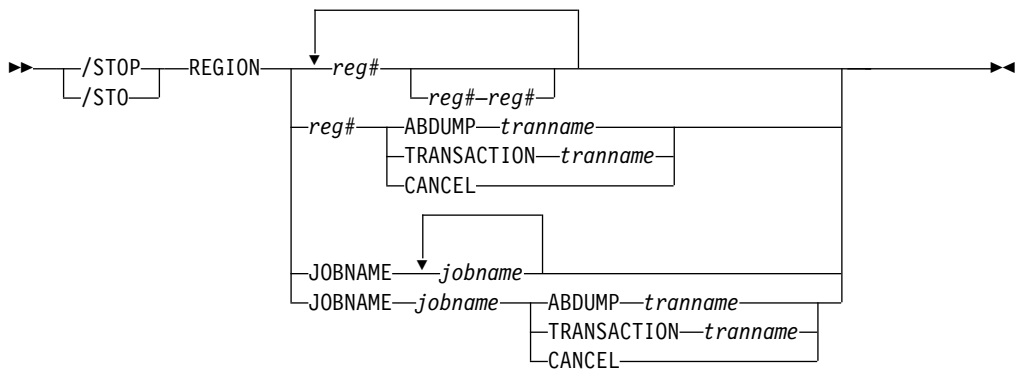
Keywords

The following keywords are valid for the /STOP REGION command:

REGION *reg#*

Is used to terminate one or more message processing regions at the conclusion of processing the current transaction. The region identifier is *reg#*.

/STOP REGION *reg#* can also be used to terminate Fast Path regions. /STOP REGION *reg#* cannot be used to terminate batch regions.

A Fast Path utility region is terminated at the next system checkpoint of the utility.

REGION *reg#–reg#*

Is used to terminate a range of message processing regions at the conclusion of processing the current transaction.

REGION *reg#* **ABDUMP** *tranname*

Causes abnormal termination of an application program.

The region identifier is *reg#* and the transaction code is *tranname*.

If the transaction indicated by *tranname* is currently running in REGION *reg#*, an error message is received at the master terminal, indicating an application

program abend. The region will remain active, but the transaction will be stopped. The command is ignored if the transaction is not currently scheduled in region *reg#*.

/STOP REGION *reg#* ABDUMP should be used only for a region that appears to be looping or in a wait state. If this command does not abnormally terminate the application running in a region, the /STOP REGION *reg#* CANCEL command can be used. This might cause the control region to terminate with abend 113 if parallel DL/I is being used. See /STOP REGION *reg#* CANCEL for further warnings.

If the /CHECKPOINT command cannot shut down IMS because a message processing region appears to be active, but the region is no longer active in the system (a condition commonly referred to as a phantom region), the /STOP REGION *reg#* ABDUMP command can be used to correct the situation. In this case, the /STOP REGION *reg#* ABDUMP command detects that the region is no longer active and cleans the internal IMS entries for the nonexistent region, which allows the shut down process to proceed normally.

If a message processing region experiences a catastrophic failure and abnormally terminates and a /DISPLAY ACTIVE REGION shows the region is still defined to IMS, the /STOP REGION *reg#* ABDUMP command can be used to correct the situation. In this case, the /STOP REGION *reg#* ABDUMP command detects the region is no longer active and cleans the internal IMS entries for the nonexistent region.

The *tranname* variable is not valid for batch, IFP, or Fast Path utility regions.

REGION *reg#* TRANSACTION *tranname*
Stops a message processing program in wait-for-input (WFI) mode from processing within the specified region.

The region identifier is *reg#* and the transaction code is *tranname*.

If the transaction indicated by *tranname* is currently running in region *reg#*, the IMS message DFS0569I is received at the master terminal, indicating that a QC status code (no more messages) was returned to the application program (MPPs). The region that contained the application is still active and the transaction is not stopped. A batch message processing program in WFI mode must be stopped using the /PSTOP command.

A batch message processing region (BMP) scheduled against wait-for-input (WFI) transactions returns a QC status code (no more messages) for, /PSTOP REGION, /DBD, /DBR, or /STA commands only.

REGION *reg#* CANCEL
Is used if the region cannot be stopped with a /STOP REGION ABDUMP command and must be preceded by a /STOP REGION ABDUMP command.

The region identifier is *reg#*.

Using the /STOP REGION CANCEL command can cause the IMS control region to terminate with user abend 113 if parallel DL/I is being used. A z/OS CANCEL command will be rejected.

REGION JOBNAME
Identifies regions to be stopped by their job names. The job name must be 1-8 alphanumeric or national ($,#,@) characters. The first character of the job name must be either alphabetic or national.

Examples

The following are examples of the /STOP REGION command:

Example 1 for /STOP REGION command

Entry ET:
```
/DISPLAY A
```

Response ET:
```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24                    1,  2
    1 BMP       BMP   BMP       BMP255
    3 IFPN      FPM   NO MSG.   DDLTJN26
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/142004*
```

Explanation: Fast Path message-driven region 3 currently has no messages to process.

Entry ET:
```
/STOP REG 3
```

Response ET:
```
DFS058I STOP COMMAND IN PROGRESS
```

Entry ET:
```
/DISPLAY A
```

Response ET:
```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24                    1,  2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBRC1CT13 DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-In -   0 ACTIV-OUT -   0
*89041/142102*
```

Explanation: Fast Path region 3 has been stopped. If region 3 had been processing a message, IMS would have terminated the region on completion of the transaction.

Example 2 for /STOP REGION command

Entry ET:
```
/DISPLAY A
```

Response ET:
```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS           CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24                    1,  2
    1 BMP       BMP   BMP       BMP255
```

```
      FPRGN    FP    NONE
      DBR1CT13 DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/142102*
```

Entry ET:

```
/STOP REG 1
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS0557I STOP REGION ID NOT VALID- REGION 0001 IS BMP.
```

Explanation: /STOP REGION (with no keywords) is not valid for batch regions.

Example 3 for /STOP REGION command

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS        CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24               1,   2
    1 BMP       BMP   BMP       BMP255
    3 FPU       FPU   IFP       DBF#FPU0
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/142453*
```

Entry ET:

```
/STOP REG 3
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
```

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS        CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24               1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-In -   0 ACTIV-OUT -   0
*89041/142758*
```

Explanation: Fast Path utility region 3 has been stopped. The Fast Path utility
DBF#FPU0 was terminated at the next system checkpoint.

Example 4 for /STOP REGION command

Entry ET:

/DISPLAY A

Response ET:
```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24                  1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE   SYS3
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/142758*
```

Explanation: Transaction TXCDRN24 in region 2 is looping or in a wait state.

Entry ET:

/STOP REG 2 ABDUMP TXCDRN24

Response ET:
```
DFS058I STOP COMMAND IN PROGRESS
DFS555I TRAN TXCDRN24 ABEND S000,U0474 SYS ID 220  MSG IN PROGRESS
```

Explanation: The application program has been terminated with a U0474 ABEND. This abend indicates termination in response to a user request (/STOP REGION ABDUMP).

Entry ET:

/DISPLAY A

Response ET:
```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
    2 MPP       TP    WAITING                            1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/143420*
```

Explanation: The application has been terminated but the region remains active.

Entry ET:

/DISPLAY PROG DDLTRN24

Response ET:
```
PROGRAM   TRAN       TYPE
DDLTRN24  TXCDRN24   TP
*90340/143749*
```

Explanation: The program has not been stopped.

Entry ET:

/DISPLAY TRANSACTION TXCDRN24

Response ET:

```
   TRAN   CLS ENQCT   QCT   LCT  PLCT CP NP LP SEGSZ SEGNO PARLM RC
TXCDRN24  2    1     0 65535 65535  1  1  1     0     0     0 0
PSBNAME: DDLTRN24
STATUS: STOP
*90340/143802*
```

Explanation: The transaction has been stopped.

Example 5 for /STOP REGION command

Entry ET:
```
/DISPLAY A
```

Response ET:
```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
    2 MPP       TP    WAITING                             1,   2
    1 BMP       BMP   BMP       BMP255
    3 FPU       FPU   IFP       DBP#FPU0
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/144248*
```

Entry ET:
```
/STOP REG 3 ABDUMP
```

Response ET:
```
DFS058I STOP COMMAND IN PROGRESS
```

Explanation: A transaction code is not entered when terminating a Fast Path utility with a /STOP REGION ABDUMP command.

Example 6 for /STOP REGION command

Entry ET:
```
/DISPLAY A
```

Response ET:
```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24 WAIT-INPUT       1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINE ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/150141*
```

Explanation: Message processing program DDLTRN24 is waiting for an input message.

Entry ET:
```
/STOP REGION 2 TRANSACTION TXCDRN24
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS0569I PSTOP OR STOP COMPLETE FOR REGION0002 TRAN TXCDRN24.
```

Explanation: A QC status code was returned to the WFI application program DDLTRN24.

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
    2 MPP       TP    WAITING                             1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-In -   0 ACTIV-OUT -   0
*89041/150206*
```

Explanation: The WFI application has been terminated but the region is still active.

Entry ET:

```
/DISPLAY TRANSACTION TXCDRN24
```

Response ET:

```
  TRAN   CLS ENQCT   QCT   LCT  PLCT CP NP LP SEGSZ SEGNO PARLM RC
TXCDRN24  2    4      0 65535 65535  1  1  1     0     0     0  0
PSBNAME: DDLTRN24
*90340/150219*
```

Explanation: The transaction is not stopped.

Example 7 for /STOP REGION command

Entry ET:

```
/DISPLAY A
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS          CLASS
    2 MPP       TP    TXCDRN24  DDLTRN24 WAIT-INPUT       1,   2
    1 BMP       BMP   BMP       BMP255
      FPRGN     FP    NONE
      DBR1CT13  DBRC
VTAM ACB CLOSED
LINE ACTIVE-IN -   1 ACTIV-OUT -   0
NODE ACTIVE-IN -   0 ACTIV-OUT -   0
LINK ACTIVE-IN -   0 ACTIV-OUT -   0
*89041/150813*
```

Entry ET:

```
/STOP REGION 2 TRANSACTION TRAN255
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS0558I TRAN TRAN255 NOT SCHEDULED
```

Explanation: TRAN255 is a valid transaction for the IMS system but it is not currently scheduled in region 2. If TRAN255 had not been a valid transaction for the IMS system, only message DFS230I (TRAN SPECIFIED WITH ABDUMP OR TRAN KEYWORD IS NOT VALID) would have been issued.

Example 8 for /STOP REGION command

Entry ET:

```
D A,L
```

Response ET:

```
JOBS     M/S     TS USERS    SYSAS     INITS    ACTIVE/MAX VTAM     OAS
00001    00010    00001      00019    00020    00001/00020        00000
LLA      LLA      LLA      NSW S VLF      VLF       VLF      NSW  S
JES2     JES2     IEFPROC  NSW S RMF      RMF       IEFPROC  NSW  S
IMSVTAM  IMSVTAM  IEFPROC  NSW S TSO      TSO       STEP1    NSW  S
CQS      CQS      IEFPROC  NSW S IMSECTA9 IMSECTA9 IEFPROC  NSW  S
DLIECTA9 DLIECTA9 DLISAS   NSW S DBRECTA9 DBRECTA9 DBRC     NSW  S
MPP610C  MPP      MPP      NSW J
USRT001  OWT
```

Explanation: MPP610C is an IMS message processing region.

Entry ET:

```
/STOP REGION JOBNAME MPP610C
```

Response ET:

```
DFS058I STOP COMMAND IN PROGRESS
DFS552I MESSAGE REGION MPP610C  STOPPED ID=00001 TIME=1616  SYSX

SMF000I  MPP610C    MPP          DFSRRC00    0000
$HASP395 MPP610C  ENDED
```

Related reference:

/STOP RTC command

Use the /STOP RTC command to specify that transactions associated with this routing code are not processed.

Subsections:

- "Environment"
- "Syntax" on page 916
- "Usage notes" on page 916
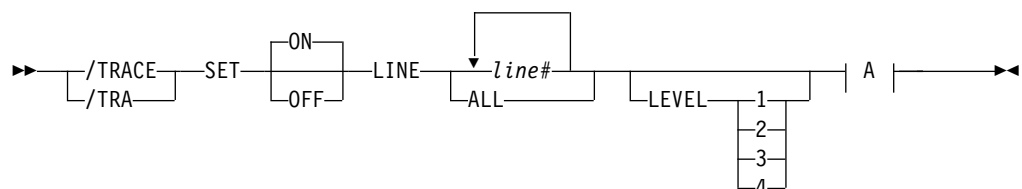- "Equivalent IMS type-2 commands" on page 916

Environment

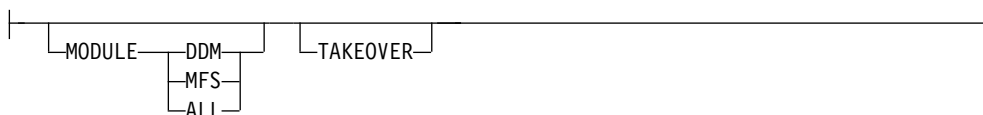The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 331. Valid environments for the /STOP RTC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
RTC	X		X

Syntax

```
>>--+--/STOP--+--RTC--+-<-------+--------------------------------><
    |         |        |   +<-+ |
    +--/STO---+        +---rtcname---+
                       |             |
                       +----ALL------+
```

Usage notes

The /STOP RTC command is not allowed for a routing code that is marked bad with the `NOTINIT-07-REPOCHGLIST` reason code because the IMS change list processing is not complete for the routing code or the change list processing failed.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP RTC command and the IMS type-2 commands that perform similar functions.

Table 332. Type-2 equivalents for the /STOP RTC command

Task	/STOP RTC command	Similar IMS type-2 command
Stops queuing to a Fast Path routing code.	/STOP RTC *rtcname*	UPDATE RTC NAME(*rtcname*) STOP(Q)

Related reference:

/STOP SB command

Use the /STOP SB command to disallow further use of sequential buffering. /STOP SB does not affect sequential buffering applications scheduled before this command was issued.

Subsections:

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 333. Valid environments for the /STOP SB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SB	X	X	

Syntax

```
►►──┬─/STOP─┬──SB──────────────────────────────────────────────────────►◄
    └─/STO──┘
```

Examples

The following is an example of the /STOP SB command:

Entry ET:
```
/STOP SB
```

Response ET:
```
DFS058  STOP COMMAND COMPLETED
```

Entry ET:
```
/DISPLAY POOL DBAS
```

Response ET:
```
SEQUENTIAL BUFFERING:  STATUS = STOPPED
  MAX     N.A. FREE   N.A.  CURR     0K  HIGH  320K
DATABASE BUFFER POOL: SIZE   67584
  REQ1      0 REQ2     0 READ     0 BISAM     0 WRITES     0
  KEYC      0 LCYL     0 PURG     0 OWNRR     0 ERRORS 00/00
  DATABASE BUFFER POOL: BSIZE 12288
    RRBA     0 RKEY     0  BFALT    0 NREC 0 SYN PTS     0
    NMBUFS  29 VRDS     0   FOUND   0 VWTS 0  ERRORS 00/00
  DATABASE BUFFER POOL: BSIZE 356352
    RRBA     0 RKEY     0  BFALT    0 NREC 0 SYN PTS     0
    NMBUFS  29 VRDS     0  FOUND    0 VWTS 0  ERRORS 00/00
  *90253/104547*
```

/STOP SERVGRP command

Use the /STOP SERVGRP command to stop communications between the service
group in an RSR complex at which the command was entered and the service
group at the other site. /STOP SERVGRP also severs the relationship between the
IMS subsystem and the TMS subsystem.

Subsections:
- "Environment"
- "Syntax" on page 918
- "Usage notes" on page 918

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 334. Valid environments for the /STOP SERVGRP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SERVGRP	X	X	X

Syntax

```
►►──┬─/STOP─┬──SERVGRP──────────────────────────────────────────────►◄
    └─/STO──┘
```

Usage notes

Once communications are stopped, the logger stops sending log data to the RSR tracking subsystem. No more attempts to re-establish failed conversations are made at OLDS switch. The /STOP SERVGRP command is normally not needed. The /STOP SERVGRP command is valid from an active subsystem and a tracking subsystem.

Successful completion of the syntax checking of the /STOP SERVGRP command results in the DFS058 STOP COMMAND COMPLETED message, although processing of the command continues asynchronously.

/STOP SLDSREAD command

The /STOP SLDSREAD command indicates whether IMS is enabled to retrieve records from both a system log data set (SLDS) and OLDS or OLDS only. The default is that SLDSREAD is enabled.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

*Table 335. Valid environments for the /STOP SLDSREAD command and keywords*

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SLDSREAD	X	X	X

Syntax

```
►►──┬─/STOP─┬──SLDSREAD──────────────────────────────────────────────►◄
    └─/STO──┘
```

Usage notes

If the SLDSREAD process is active, issuing the /STOP SLDSREAD command causes a U4095 symptom dump to be taken, and the SLDSREAD processing is stopped. This abend is not fatal, and IMS continues to function normally. Any backout processes that were active have to be backed out manually by using the batch backout utilities.

If the SLDSREAD process is not active, issuing the /STOP SLDSREAD command prevents any SLDSREAD processes from being started if an SLDS should be needed for backout processing. Use the batch backout utilities to back out any application programs that failed.

/STOP SUBSYS command

Use the /STOP SUBSYS command to specify the name of the external subsystem whose connection is to be terminated.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 920

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 336. Valid environments for the /STOP SUBSYS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SUBSYS	X	X	X

Syntax
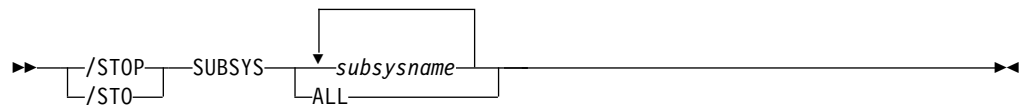
Usage notes

The /STOP SUBSYS command does enable application programs currently accessing external resources to complete normally. When those applications have terminated, the connection to the subsystem will also terminate. The application must complete all message processing before actual connection termination. The next occurrence of an external subsystem call will receive a nonzero return code, indicating the connection is not available. A /START command is then necessary to reestablish the connection.

The /STOP SUBSYS command can also be used to dynamically reconfigure existing subsystem definitions. The operator can issue the /STOP SUBSYS command, change or add to the PROCLIB member, and then issue the /START SUBSYS command. IMS attempts to connect those subsystems defined in the PROCLIB member.

If system failure occurs after the /STOP SUBSYS command is processed, the stopped status is still set.

Examples

The following are examples of the /STOP SUBSYS command:

Example 1

Entry ET:
```
/STOP SUBSYS ALL
```

Response ET:
```
DFS058I  STOP COMMAND IN PROGRESS
```

Explanation: IMS has initiated the termination of the connection. When all dependent regions have terminated their connections, IMS will complete the termination. It is likely that an external subsystem message indicating connection termination will be received at this time.

Example 2

Entry ET:
```
/STOP SUBSYS XXX1 XXX3
```

Response ET:
```
DFS058I  STOP COMMAND IN PROGRESS
```

Explanation: IMS has initiated the termination of the connection. When all dependent regions have terminated their connections, IMS will complete the termination. It is likely that an external subsystem message indicating connection termination will be received at this time

/STOP SURV command

Use the /STOP SURV command in an XRF environment to stop the operation of the IMS surveillance function.

Subsections:
- "Environment"
- "Syntax" on page 921
- "Keywords" on page 921

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 337. Valid environments for the /STOP SURV command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
SURV	X		X

Syntax

```
►►──┬─/STOP─┬──SURV──┬─ALL─┬──────────────────────────────────────────►◄
    └─/STO──┘        ├─LNK─┤
                     ├─LOG─┤
                     └─RDS─┘
```

Keywords

The following keywords are valid for the /STOP SURV command:

ALL
> The same as specifying LNK, LOG, and RDS. This is the default.

LNK
> ISC link.

LOG
> System log.

RDS
> Restart data set.

/STOP THREAD command

Use the /STOP THREAD command to stop an inactive CCTL thread. The DEDB utility region is terminated at the next system checkpoint.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 922
- "Usage notes" on page 922
- "Examples" on page 922

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 338. Valid environments for the /STOP THREAD command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
ABDUMP	X	X	X
CANCEL	X	X	X
THREAD	X		X
TRANSACTION	X		X

Syntax

```
                              ┌──────────────┐
                              ▼              │
►►──┬─/STOP─┬──THREAD──┬───reg#─────┴───────────────────────────────────────►◄
    └─/STO──┘          └─reg#─┬─ABDUMP──tranname────────┬─
                             ├─TRANSACTION──tranname───┤
                             └─CANCEL──────────────────┘
```

Keywords

For information about the keywords that are valid for the /STOP THREAD command, see the description about the same keywords under the /STOP REGION command.

Usage notes

The /STOP THREAD command is not valid for:
- Active CCTL threads
- BMPs

The THREAD ABDUMP command abnormally ends BMPs and DEDB utilities. If this command is used with CCTL threads, a U0474 abend results.

Examples

The following are examples of the /STOP THREAD command:

Example 1

Entry ET:
```
/DISPLAY A REGION
```

Response ET:
```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS     CLASS
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
   2  CICS1A    DBT   IEFPROC   BMP255   ACTIVE
   3  CICS1A    DBT   IEFPROC   PLVAPZ12 ACTIVE
   1  CICS1A    DBT   IEPROC             AVAILABLE
      DBRCHTA1  DBRC
      DLICHTA1  DLS
*00082/142907*
```

Entry ET:
```
/STOP THREAD 2
```

Response ET:
```
DFS058I STOP COMMAND IN PROGRESS
DFS0556I COMMAND REJECTED; DBCTL THREAD IS ACTIVE
```

Example 2

Entry ET:
```
/DISPLAY A REGION
```

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS     CLASS
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
    2 CICS1A    DBT   IEFPROC   BMP255   ACTIVE
    3 CICS1A    DBT   IEFPROC   PLVAPZ12 ACTIVE
    1 CICS1A    DBT   IEFPROC            AVAILABLE
      DBRCHTA1  DBRC
      DLICHTA1  DLS
*00082/143027*
```

Entry ET:

`/STOP THREAD 1`

Response ET:

`DFS058I STOP COMMAND IN PROGRESS`

Entry ET:

`/DISPLAY A REGION`

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS     CLASS
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
    2 CICS1A    DBT   IEFPROC   BMP255   ACTIVE
    3 CICS1A    DBT   IEFPROC   PLVAPZ12 ACTIVE
      DBRCHTA1  DBRC
      DLICHTA1  DLS
*00082/143055*
```

Example 3

Entry ET:

`/DISPLAY A REGION`

Response ET:

```
REGID JOBNAME   TYPE  TRAN/STEP PROGRAM  STATUS     CLASS
      BATCHREG  BMP   NONE
      FPRGN     FP    NONE
    2 CICS1A    DBT   IEFPROC   BMP255   ACTIVE
    3 CICS1A    DBT   IEFPROC   PLVAPZ12 ACTIVE
      DBRCHTA1  DBRC
      DLICHTA1  DLS
*00082/144731*
```

Entry ET:

`/STOP THREAD 2 ABDUMP`

Response ET:

`/DFS058I STOP COMMAND IN PROGRESS`

Response ET:

```
DFS554A CICS1A     00002 IEFPROC  BMP255   (3)        000,0474   20
/082 14:49:11    RTKN= CICS1    B3C81CB789F4BE83
```

Entry ET:

`/DISPLAY A REGION`

Response ET:

```
REGID JOBNAME    TYPE  TRAN/STEP PROGRAM  STATUS      CLASS
      BATCHREG   BMP   NONE
      FPRGN      FP    NONE
    3 CICS1A     DBT   IEFPROC   PLVAPZ12 ACTIVE
      DBRCHTA1   DBRC
      DLICHTA1   DLS
*00082/145038*
```

Related reference:

/STOP TMEM command

Use the /STOP TMEM command to cause IMS to send an Open Transaction
Manager Access (OTMA) command to OTMA clients to request that input be
suspended for the specified transaction pipe name.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 925
- "Examples" on page 925

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 339. Valid environments for the /STOP TMEM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
TMEM	X		X
TPIPE	X		X

Syntax
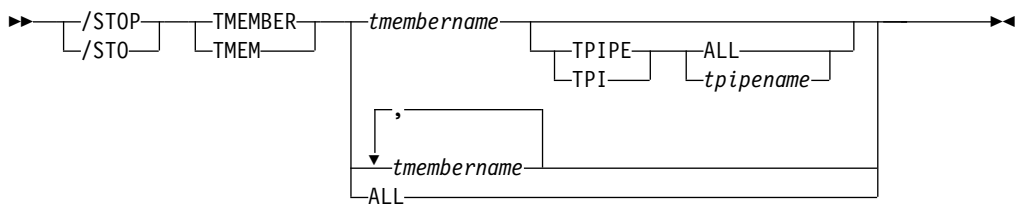
Keywords

The following keyword is valid for the /STOP TMEM command:

TPIPE
> When used with the /STOP command, TPIPE causes IMS to send an OTMA
> command to its OTMA clients to request that the input be suspended for the
> specified transaction pipe. IMS then stops sending output to the OTMA client.
> If the member specified is a super member, output is suspended for the super

member's transaction pipe, but no OTMA command is sent. If the member specified is a regular member whose hold queue output is managed by a super member, IMS suspends output from the specified member's transaction pipe and it also suspends output from the super member's transaction pipe. An OTMA command is sent to the regular member's OTMA client. Output is only suspended on the IMS that processes the command. If output cannot be suspended for both the regular member's transaction pipe and the super member's transaction pipe, it is not suspended for either transaction pipe. The DFS058I COMMAND COMPLETED EXCEPT message is issued with the name of the regular member for which output could not be suspended

While processing the /STOP TMEMBER TPIPE command, IMS creates a temporary transaction pipe (if one does not already exist) with the stopped status. IMS sets the synchronization status for this transaction pipe when it sends or receives the first message for the transaction pipe.

While processing the /STOP TMEMBER TPIPE command, OTMA checks for wait status (WAIT_A, WAIT_H, WAIT_R, and WAIT-SYNCPOINT) for the messages using the tpipe. If a wait status is found, OTMA clears the wait status by generating an internal NAK message. This NAK message for a send-then-commit (CM1) response will cause a U0119 pseudoabend for the transaction. However, the NAK message for a commit-then-send (CM0) response will wash back the response to the tpipe queue. The message in the tpipe queue can be retrieved again later.

After a /STOP TMEMBER *xxx* TPIPE ALL command is issued, newly created tpipes will not be stopped for either input or output.

The /STOP TMEMBER *xxx* TPIPE *xxx* command will not create a temporary tpipe if the tmember does not exist (DFS058I STOP COMMAND COMPLETED EXCEPT TPIPE *xxx* will be issued).

Usage notes

You can stop any number of individual tmembers or all tmembers. IMS then stops sending output to the OTMA client and prevents any further output from being sent to the client.

When the /STOP TMEM TPIPE command is issued, it will clear the wait states of all messages for the transaction pipe.

You can issue the /STOP TMEM command to the DFSYICAL OTMA member to halt synchronous program switch processing. After the DFSYICAL tmember is stopped, all subsequent DL/I ICAL calls made to initiate a synchronous program switch are rejected. The requesting application will receive the AIB return code X'00000100' with the reason code X'00000110' and the extended reason code X'00000005'.

Examples

The following is an example of the /STOP TMEM command:

Entry ET:
```
/STO TMEMBER CLIENT1 TPIPE TPIPESY
```

Response ET:
```
DFS058I 15:38:03 STOP COMMAND COMPLETED   SYS3
DFS996I *IMS READY*  SYS3
```

/STOP TRAN command

Use the /STOP TRAN command to stop the queuing and scheduling of messages destined for a transaction or class of transactions, or to stop transaction scheduling by class. However, output can still be queued if it originates from the application program.

The /STOP TRAN command stops the scheduling of transactions; however, the transactions will continue to be processed until the limit count is reached. If the limit count is large, the processing interval will be long.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands" on page 927
- "Examples" on page 927

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 340. Valid environments for the /STOP TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
CLASS	X		X
TRAN	X		X

Syntax
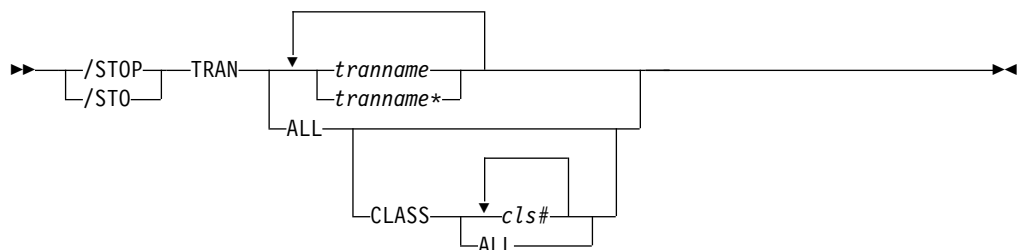
Usage notes

If the region is already scheduled and waiting for the next message (wait-for-input mode) when the command is entered, a QC status (no more messages) is returned to the application (MPPs). If there are no more messages available for the region to process, the region is not allowed to wait for the next message. Instead, a QC status is returned to the application (MPPs).

A batch message processing region (BMP) scheduled against wait-for-input (WFI) transactions returns a QC status code (no more messages) for /PSTOP REGION, /DBD, /DBR, or /STA commands only.

In a shared-queues environment, if you issue a /STOP TRAN command for a transaction that is not defined on that IMS subsystem, IMS creates an SMB if the Output Creation user exit routine indicates the destination is a valid transaction. The SMB is marked as "dynamic".

A dynamic SMB created by a /STOP TRAN command can only be used to queue messages for the transaction and place the messages on the shared queues. The transaction cannot be scheduled or assigned. IMS does process checkpoints for the transaction, but does not save them across an IMS restart if they do not have a valid status.

In a shared-queues environment, the /STOP TRAN command will result in IMS deregistering interest for the transaction, which indicates that the transaction cannot be scheduled at that IMS.

The /STOP TRAN command is not allowed for a transaction that is marked bad with the NOTINIT-15-REPOCHGLIST reason code because the IMS change list processing is not complete for the transaction or the change list processing failed.

Equivalent IMS type-2 commands

The following table shows variations of the /STOP TRAN command and the IMS type-2 commands that perform similar functions.

Table 341. Type-2 equivalents for the /STOP TRAN command

Task	/STOP TRAN command	Similar IMS type-2 command
Stops the queuing and scheduling of messages destined for a transaction.	/STOP TRAN *tranname*	UPDATE TRAN NAME(*tranname*) STOP(Q,SCHD)

Examples

The following are examples of the /STOP TRAN command:

Example 1 for /STOP TRAN command

Entry ET:
```
/STOP TRANSACTION ALL CLASS 6
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: All transactions associated with class 6 will be marked as stopped and all class 6 transactions are no longer available for scheduling. All message processing regions currently processing class 6 transactions are allowed to run until the processing limit count is reached or the input queue contains no more messages.

Example 2 for /STOP TRAN command

Entry ET:
```
/STOP TRANSACTION PIT, SEED
```

Response ET:
```
DFS058I  STOP COMMAND COMPLETED
```

Explanation: Transaction codes PIT and SEED are stopped.

Related reference:

"UPDATE TRAN command" on page 1326

/STOP USER command

The /STOP USER command requires the ISC user to stop or the signed on user to stop and sign off. The USER parameter can be generic and applies only to users that already exist.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Examples" on page 929

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 342. Valid environments for the /STOP USER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
USER	X		X

Syntax
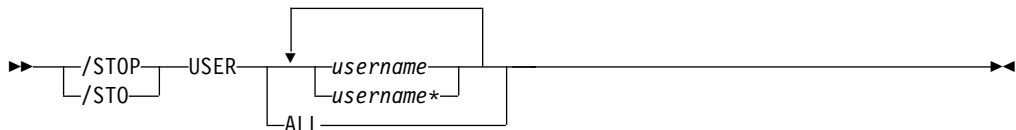
Usage notes

For ISC users, the /STOP USER command specifies the ISC user that is to be made unavailable for allocation until a /START USER command is issued.

For signed on users, the /STOP USER command should specify the user structure name to prevent future signons until a /START USER command is issued.

The /STOP USER command is supported for users that do not yet exist. It causes the user to be created and stopped, which prevents the dynamic user from signing on.

The /STOP USER command will not cause the user to be signed off if the associated node is not active or the associated node is not in session.

The /STOP USER command for an ETO user session resets status that is not significant such as preset mode, test mode, lock lterm, pstop lterm, and purge lterm. For Fast Path input response mode, you must also issue the /START USER command before the mode is reset.

The /STOP USER command for ETO users also takes other actions depending on the recovery settings for the user:

RCVYCONV=NO
> /STOP USER causes any IMS conversations (active and held) for an ETO user to be terminated. Any conversational message that is queued or being processed will have its output response message delivered asynchronously.

RCVYFP=NO
> /STOP USER causes Fast Path status and messages for an ETO user to be discarded.

RCVYRESP=NO
> /STOP USER resets full-function response mode.

If global resource information is kept in Resource Manager, the /STOP USER command sets a global stop signon status for the user and prevents the user from signing on anywhere in the IMSplex. If global resource information is not kept in Resource Manager, /STOP USER creates the user, if it does not exist in an ETO environment, and sets stop status for the local user. If the user does not exist in a non-ETO environment, the /STOP USER command is rejected.

Examples

The following is an example of the /STOP USER command:

Entry ET:
```
/DISPLAY USER IMS*
```

Response ET:
```
USER      ENQCT DEQCT   QCT   SYS3
IMSUS06      0     0      0 ALLOC(DTSLU602)
IMSUS04      0     0      0 ALLOC(DTSLU603)
IMSUS03      0     0      0 ALLOC(DTSLU601)
IMSUS02      0     0      0 ALLOC(DTSLU202)
IMSUS01      0     0      0 ALLOC(DTSLU201)
IMSUS09    N/A   N/A    N/A ALLOC(ENDS02  ) STATIC
IMSUS08    N/A   N/A    N/A ALLOC(ENDS01  ) STATIC
IMSUS11    N/A   N/A    N/A ALLOC(ENDS03  ) STATIC
IMSUS10    N/A   N/A    N/A ALLOC(OMSSLU2A) STATIC
*91091/111454*
```

Entry ET:
```
/STOP USER IMSUS01 IMSUS02
```

Response ET:
```
DFS058I 11:16:24 STOP COMMAND COMPLETED
```

Entry ET:
```
/STOP USER HELLO%
```

Response ET:
```
DFS3633 11:18:25 GENERIC PARAMETER RESOURCES NOT FOUND, NO ACTION TAKEN
```

Entry ET:
```
/DISPLAY USER IMSUS01 IMSUS02
```

Response ET:
```
USER     ENQCT DEQCT  QCT
IMSUS01      0     0     0 STOPPED
IMSUS02      0     0     0 STOPPED
*91091/111727*
```

/STOP VGR command

Use the /STOP VGR command to cause the IMS subsystem to drop out of a generic resources group. This command is rejected if the VTAM ACB is closed (usually the result of a /STOP DC command).

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 343. Valid environments for the /STOP VGR command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
VGR	X		X

Syntax

```
►►──┬─/STOP─┬──VGR──────────────────────────────────────────►◄
    └─/STO──┘
```

Usage notes

While this command prevents VTAM from routing new sessions using a generic resource name to the IMS subsystem, it does not affect existing sessions, and affinities remain (until terminated through normal processing).

/STOP WADS command

Use the /STOP WADS command to indicate that a WADS is to be removed from the pool of available WADS. IMS does not enable the active WADS (if WADS mode is single), or the active WADS pair (if WADS mode is dual), to be stopped. wads# must be 0 through 9.

Subsections:
- "Environment" on page 931
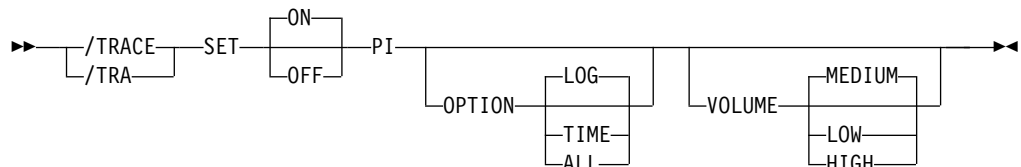- "Syntax" on page 931

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 344. Valid environments for the /STOP WADS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
WADS	X	X	X

Syntax

```
►►──┬─/STOP─┬──WADS─wads#──────────────────────────────────────────────►◄
    └─/STO──┘
```

/STOP XRCTRACK command

The /STOP XRCTRACK command results in calls to the log router to initiate or terminate XRC tracking. It is only valid on a tracking IMS system.

Subsections:
- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 345. Valid environments for the /STOP XRCTRACK command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/STOP	X	X	X
DC	X	X	X

Syntax

```
►►──┬─/STOP─┬──XRCTRACK────────────────────────────────────────────────►◄
    └─/STO──┘
```

Chapter 25. /SWITCH command

Use the /SWITCH command to switch active data sets or to change between the active and alternate systems. Certain combinations of keywords are valid only in the active or alternate systems, as shown in the format in the syntax diagram.

This command can be issued to an IMSplex using the Batch SPOC utility.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 934
- "Examples" on page 935

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 346. Valid environments for the /SWITCH command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/SWITCH	X	X	X
ABDUMP	X		X
ACTIVE	X		X
BACKUP	X		X
CHECKPOINT	X	X	X
FORCE	X		X
OLDS	X	X	X
SYSTEM	X		X
WADS	X	X	X

Syntax

/SWITCH for an active XRF subsystem
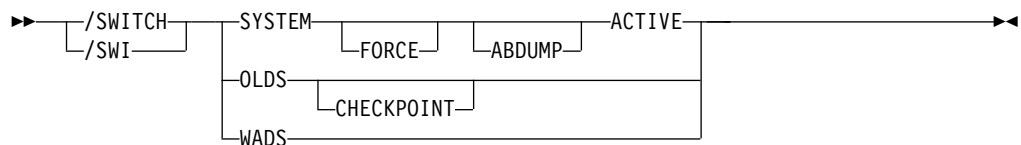
/SWITCH for an alternate XRF subsystem

```
►►──┬──/SWITCH──┬──┬──SYSTEM──FORCE──────────────────────────────────────────►◄
    └──/SWI─────┘  │              └──ABDUMP──┘  └──BACKUP──┘
                   ├──OLDS────────────────────────────────────────┐
                   │         └──CHECKPOINT──┘                      │
                   └──WADS──────────────────────────────────────────┘
```
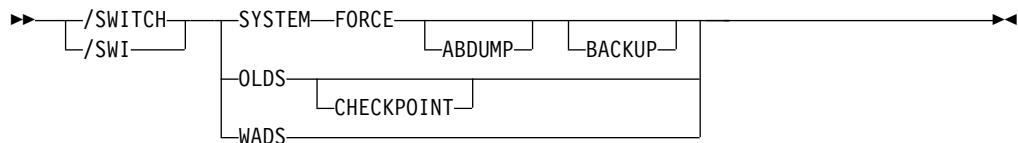
Keywords

The following keywords are valid for the /SWITCH command:

SYSTEM

Requests a takeover by the alternate system from the current active system in an XRF environment.

/SWITCH SYSTEM without the FORCE keyword causes the active system to attempt to gracefully quiesce before the alternate system becomes active. System activity currently in progress is allowed to complete. New activity is queued. This disables surveillance on the active system, but not the alternate. Therefore, surveillance (if enabled) will eventually trigger a takeover if it does not eventually quiesce.

Unless the optional FORCE keyword is specified, the command is only operable when entered on the active system.

FORCE

Causes an immediate termination of the active system, forcing the alternate system to become the active.

ABDUMP

Results in a diagnostic dump of the active system when entered from either the active system or the alternate system (if it is on the same processor as the active system).

ACTIVE, BACKUP

Indicates the system on which the command is being entered. The keyword ACTIVE is required when the command is entered on an active system. The keyword BACKUP is optional when the command is entered on an alternate system. /SWITCH SYSTEM FORCE, without the ACTIVE keyword, can only be entered on an alternate system. This prevents the inadvertent abend of a newly created active system that is mistakenly assumed to still be the alternate system.

OLDS, CHECKPOINT

Causes switching of the active log data set. This log switch capability is identical to that provided with /DBDUMP and /DBRECOVERY commands. You can specify the CHECKPOINT keyword to take a simple checkpoint after the active log data set has been switched to the next OLDS. The /SWITCH OLDS CHECKPOINT command operates in all IMS environments.

WADS

Causes switching of the active write-ahead log data set. If you are using dual logging for the WADS, this command causes IMS to use the next available WADS pair.

This command is rejected if no unused WADS is available, or for dual logging, if no unused pair of WADS is available.

Examples

The following are examples of the /SWITCH command:

Example 1 for /SWITCH command

Entry ET:
```
/SWITCH OLDS
```

Response ET:
```
DFS3257I ONLINE LOG NOW SWITCHED
DFS058I 17:10:51 SWITCH COMMAND COMPLETED
```

Entry ET:
```
/SWITCH OLDS CHECKPOINT
```

Response ET:
```
DFS3257I ONLINE LOG NOW SWITCHED - FROM DFSOLP01 TO DFSOLP02
DFS058I 17:12:53 SWITCH COMMAND COMPLETED

DFS2719I MSDB CHECKPOINT WRITTEN TO MSDBCP2
DFS994I *CHKPT 91057/171254**SIMPLE*
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:    1
DFS3804I LATEST RESTART CHKPT: 91057/132000, LATEST BUILDQ CHKPT:
91057/132414
```

Example 2 for /SWITCH command

Entry ET (Master Terminal for active system IMSA):
```
/SWITCH SYSTEM FORCE
```

Response ET:

A response message is not returned for the /SWITCH SYSTEM FORCE command. Any further input to the master terminal of the active system is inhibited.

Response RT (z/OS console for active system IMSA):

The z/OS console for the active system will show a user 0604 abend in progress for IMSA. If the ABDUMP keyword had been included on the /SWITCH command, the 0604 abend would be accompanied by a diagnostic dump of the active system.

Response RT (master terminal for alternate system IMSB):

The following figure is a screen that shows some of the messages associated with the beginning of takeover on the alternate system.

```
02/05/15 15:28:27  RSENAME: DFSRSENM  BACKUP   TAKEOVER IN PROGRESS  IMSB
DFS3890I 15:27:18 TAKEOVER REQUESTED
DFS970I 15:28:05 UNEXPECTED  STATUS   ,NODE APPLA  ,USER  N/A  ,SEND  ,RC
=14,FDB2=13,NSECIT =29,SENSE=00000000,REASON=00
DFS3257I ONLINE LOG CLOSED ON DFSOLP00
DFS3891I 15:28:18 TAKEOVER IN PROGRESS

 DFS2591I NO MSDB HEADERS FOUND, IMAGE COPY LOAD IGNORED
DFS3839I 14:26:46 XRF INITIAL DB PRE-OPEN COMPLETE.
DFS3838I 14:28:41 XRF INITIAL DC PRE-OPEN COMPLETE.
-------------------------------------------------------------------------




                                               PASSWORD:
```

Figure 1. Alternate system at start of takeover

Intermediate screens are not shown. They would indicate such takeover functions as:

- Enabling of dependent region processing
- IRLM takeover
- Backout processing
- Draining of suspend queue
- Session switching

The following figure is a screen that show takeover is complete.

```
02/05/15 15:30:59  RSENAME: DFSRSENM  ACTIVE  AWAITING I/O PREVENTION  IMSB
 DFS994I *CHKPT 85135/152931**SIMPLE**
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:   11
DFS3804I LAST CHKPT ID VALID FOR RESTART: 85135/152931-BUILDQ: 85135/142629


DFS994I TAKEOVER COMPLETED.
DFS3859I 15:29:19 PRIORITY 4 SESSIONS SWITCHED.
DFS3860I 15:29:19 ALL TERMINAL SESSIONS SWITCHED.
-------------------------------------------------------------------------




                                               PASSWORD:
```

Figure 2. Newly created active system after takeover

Takeover is complete and the alternate system is now an active system. The XRF environment status line indicates that the newly created active system is running in I/O toleration mode (awaiting I/O prevention).

Related concepts:

Takeover phase of the XRF process (System Administration)

Chapter 26. TERMINATE commands

Use the TERMINATE commands to terminate a global online change or to stop one or more HALDB OLRs that are in progress.

- "TERMINATE OLC command"
- "TERMINATE OLREORG command" on page 951

TERMINATE OLC command

When the TERMINATE OLC (stop online change) command is issued by an IMS command master that is running with RM services (RMENV=Y), the command terminates a global online change and coordinates with all of the IMS systems in the IMSplex.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "TERMINATE OLC error handling" on page 940
- "Output fields" on page 942
- "Return, reason, and completion codes" on page 943
- "Examples" on page 949

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the TERMINATE OLC command can be issued.

Table 347. Valid environments for the TERMINATE OLC command

Command / Keyword	DB/DC	DBCTL	DCCTL
TERMINATE OLC	x	x	x

Syntax

```
►►──┬─TERMINATE─┬──OLC───────────────────────────────────────────────►◄
    └─TERM──────┘
```

Usage notes

If an IMS is not running with RM services (RMENV=N), the TERMINATE OLC command terminates an online change for that IMS only. If an IMS system is not running with RM, each IMS must have a unique OLCSTAT data set which cannot be shared. If the OLCSTAT data set contains the name of an IMS other than the one that is processing the online change, TERMINATE OLC is rejected because the OLCSTAT data set is invalid for the environment. To determine which IMS member names are invalid, issue the QUERY OLC command to display the contents of the OLCSTAT data set. You can use the OLC utility, DFSUOLC0, to correct the data set.

Each IMS system that does not have RM services and participates in global online change, must separately issue the TERMINATE OLC command. In a non-RM environment, if more than one IMS is specified in the route list for the TERMINATE OLC command or the default of route all is specified, online change is only performed for the IMS command master. To determine which IMS systems are defined with RMENVNO, issue a QUERY MEMBER SHOW(ATTRIB) command.

This command can be specified only through the OM API. OM sends the TERMINATE OLC command to an IMS in the IMSplex.

This command can be issued to an IMSplex using the Batch SPOC utility.

The TERMINATE OLC command can be used to abort an IMSplex-wide global online change initiated by a INITIATE OLC PHASE(PREPARE) command, before the online change is successfully committed with a INITIATE OLC PHASE(COMMIT) command.

The TERMINATE OLC command can be used to abort an online change after an INITIATE OLC PHASE(COMMIT) failure that occurs before the OLCSTAT data set is updated. Once the commit process has updated the OLCSTAT data set, the online change is considered to be successful and cannot be aborted.

A TERMINATE OLC command that aborts a global online change is similar to the /MODIFY ABORT command, except that it applies to all of the IMS systems in an IMSplex that are participating in the global online change. The TERMINATE OLC command can be used to terminate an INIT OLC TYPE(ACBMBR) command in progress. If the updated members have not been committed, the TERM OLC command will delete the new versions of the updated members in the active ACBLIB. The TERM OLC command for a TYPE(ACBLIB) will log a X'7010' log record indicating that the member OLC has been terminated.

The TERMINATE OLC command is not supported if local online change is enabled. The TERMINATE OLC command is rejected if the IMS to which the command is routed does not support global online change. If this occurs and there is an IMS that supports global online change, the user must route the command to a specific IMS that supports global online change.

The TERMINATE OLC command is invalid on the XRF alternate, RSR tracker, and FDBR system.

The OM command timeout default of 300 seconds (5 minutes) might not be enough time for the online change phase to complete. You might need to specify a timeout value on the command based on the needs of the installation. To change the timeout value, use the TIMEOUT parameter of the CSLOMCMD command request or specify a Wait (timeout) value when you issue the TERMINATE OLC command from TSO SPOC.

The command syntax for this command is defined in XML and is available to automation programs which communicate with OM.

TERMINATE OLC error handling

Errors unique to the processing of this command are returned as a completion code. A completion code is returned for each action against an individual resource.

The TERMINATE OLC command might result in an error that leaves IMS systems in various online change states. Correct the error. Issue the QUERY MEMBER TYPE(IMS) SHOW(STATUS) command to display the online change state of all the IMS systems in the IMSplex. Evaluate the QUERY MEMBER TYPE(IMS) output to help you determine what to do:

- None of the IMS systems in an online change state

  The TERMINATE OLC command succeeded or was not applicable. No further action needs to be taken.

- Some of the IMS systems in a prepare complete state

  The online change is not committed. Correct the problem that caused TERMINATE OLC to fail, then try the TERMINATE OLC command again.

  The IMS systems that are in an online change state remain in an online change state until you abort the online change.

- All IMS systems in a prepare complete state

  The online change is not committed. Correct the problem that caused TERMINATE OLC to fail, then try the TERMINATE OLC command again.

  The IMS systems that are in an online change state remain in an online change state until you abort the online change.

- Some IMS systems in prepare complete and commit phase 1 complete state

  The commit phase failed before the master updated the OLCSTAT data set, so the online change is not committed. Correct the problem that caused TERMINATE OLC to fail and try the TERMINATE OLC command again.

  The IMS systems that are in an online change state remain in an online change state until you abort the online change.

- All IMS systems in commit phase 1 complete state

  If the commit phase failed before the master updated the OLCSTAT data set, the online change is not committed. Correct the problem that caused TERMINATE OLC to fail and try the TERMINATE OLC command again.

  If the commit phase failed after the master updated the OLCSTAT data set, the online change is committed. The TERMINATE OLC command is not permitted. You must correct the problem that caused the commit command to fail and try the INITIATE OLC PHASE(COMMIT) command again.

  The IMS systems that are in an online change state remain in an online change state until you abort the online change or commit the online change.

  You can determine whether the OLCSTAT data set has been updated by the modify ID. Issue the QUERY OLC LIBRARY(OLCSTAT) SHOW(MODID) command. Check if the modify ID returned is different from the modify ID returned by the INITIATE OLC PHASE(PREPARE) command, or the modify ID returned by a QUERY OLC LIBRARY(OLCSTAT) SHOW(MODID) command issued before the INITIATE OLC PHASE(COMMIT) command.

- Some IMS systems in commit phase 1 complete state and some in commit phase 2 complete state

  The online change is committed. The TERMINATE OLC command is not permitted. You must correct the problem that caused the commit command to fail and try INITIATE OLC PHASE(COMMIT) again.

  The IMS systems that are in an online change state remain in an online change state until you finish the online change with an INITIATE OLC PHASE(COMMIT) command.

- All IMS systems in commit phase 2 complete state

The online change is committed. The TERMINATE OLC command is not permitted. You must correct the problem that caused the commit command to fail and try INITIATE OLC PHASE(COMMIT) again.

The IMS systems that are in an online change state remain in an online change state until you finish the online change with an INITIATE OLC PHASE(COMMIT) command.

- Some IMS systems in commit phase 2 complete state and some not in online change state

  The online change is committed. The TERMINATE OLC command is not permitted. You must correct the problem that caused the commit command to fail and try INITIATE OLC PHASE(COMMIT) again.

  The IMS systems that are in an online change state remain in an online change state until you finish the online change with an INITIATE OLC PHASE(COMMIT) command.

Errors unique to the processing of this command are returned as a completion code. A completion code is returned for an IMS participating in the online change phase.

Output fields

The following table shows the output fields for a TERMINATE OLC command. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Keyword
> Identifies keywords on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 348. Output fields for the TERMINATE command

Short label	Keyword	Meaning
MBR	N/A	IMSplex member that built the line of output. IMS identifier of the IMS that was master of the abort phase. IMS identifier is always returned.
IMSMBR	N/A	IMS member that performed the global online change phase. The IMS member name is always returned.
CC	N/A	Completion code from the IMS member that performed the online change phase. Completion code is always returned.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
ERRT	N/A	Error text associated with a nonzero completion code returned by the IMS member that performed the online change phase. Error text might be returned if the completion code is nonzero.

Return, reason, and completion codes

The OM return and reason codes that might be returned as a result of this command are standard for all commands entered through the OM API. Refer to the OM CSLOMCMD Return and Reason code section for the list of codes and their meanings.

An IMS return and reason code is returned to OM by the TERMINATE OLC command.

Some reason codes are accompanied by a complete list of IMS systems and completion codes. The reason code meaning indicates whether a list is returned. A partial list of IMS systems and completion codes might be returned with any TERMINATE OLC error reason code, if any output was built before the error was detected.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 349. Return and reason codes for the TERMINATE OLC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The TERMINATE OLC command completed successfully. The TERMINATE OLC command is applied to all of the IMS systems listed in the OLCSTAT data set. All of the IMS systems in the IMSplex are no longer in an online change state.
		An output line is built for each IMS listed in the OLCSTAT data set. Each output line contains the IMS member name and a completion code of zero.
X'00000004'	X'0000100C'	The TERMINATE OLC command completed successfully, but was not applicable to one or more IMS systems for acceptable reasons. The TERMINATE OLC command applies to all of the IMS systems listed in the OLCSTAT data set.
		An output line is built for each IMS listed in the OLCSTAT data set. Each output line contains the IMS member name and a completion code. A nonzero completion code may be accompanied by error text. One or more of the IMS systems contain a completion code that indicates that the terminate online change did not apply to this IMS, such as the IMS state is abended, the IMS state is shutdown, or this IMS is already in the correct online change state. The TERMINATE OLC completion code table contains the list of completion codes and error text that can be returned by the TERMINATE OLC command.

Table 349. Return and reason codes for the TERMINATE OLC command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003000'	The TERMINATE OLC command is successful for at least one IMS but not all IMS systems. The TERMINATE OLC command applies to all IMS systems listed in the OLCSTAT data set. An output line is built for each IMS listed in the OLCSTAT data set. Each output line contains the IMS member name and a completion code. A nonzero completion code might be accompanied by error text. One or more of the IMS systems returned an error completion code. The TERMINATE OLC completion code table contains the list of completion codes and error text that can be returned by the TERMINATE OLC command. If the TERMINATE OLC command fails for one or more IMS systems, correct the problem and issue the TERMINATE OLC command again. For more details, see "TERMINATE OLC error handling" on page 940.
X'0000000C'	X'00003004'	The TERMINATE OLC command failed for all of the IMS systems. The TERMINATE OLC command applies to all of the IMS systems listed in the OLCSTAT data set. An output line is built for each IMS listed in the OLCSTAT data set. Each output line contains the IMS member name and a completion code. A nonzero completion code might be accompanied by error text. All of the IMS systems returned an error completion code. The TERMINATE OLC completion code table contains the list of completion codes and error text that can be returned by the TERMINATE OLC command. If the TERMINATE OLC command fails for one or more IMS systems, correct the problem and issue the TERMINATE OLC command again. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000010'	X'00004004'	The TERMINATE OLC command failed because there is no CQS. RM attempted to access the process resource on the resource structure, but it failed because CQS is not available. The online change phase might have succeeded on one or more IMS systems. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000010'	X'0000400C'	The TERMINATE OLC command failed because it is invalid for an XRF alternate.
X'00000010'	X'00004014'	The TERMINATE OLC command failed because it is invalid for an RSR tracker.
X'00000010'	X'00004018'	The TERMINATE OLC command failed because the RM resource structure is not available. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000010'	X'0000401C'	The TERMINATE OLC command failed because it is invalid for an FDBR region.
X'00000010'	X'00004100'	The TERMINATE OLC command is rejected because the resource structure is full. RM failed trying to create the process resource on the resource structure. One or more IMS systems might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 940.

Return code	Reason code	Meaning
X'00000010'	X'00004104'	The TERMINATE OLC command failed because RM is not available. The online change phase might have succeeded on one or more IMS systems. Either there is no RM address space, or RM is active but not registered to SCI because CQS or the resource structure is not available. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000010'	X'00004108'	The TERMINATE OLC command failed because SCI is not available. One or more IMS systems might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000010'	X'0000410C'	The TERMINATE OLC command is rejected, because global online change is not enabled. Local online change is enabled. Use the /MODIFY command for local online change. If your IMSplex is made up of some IMS systems that support global online change and some that support local online change, route the TERMINATE OLC command to an IMS that is enabled for global online change.
X'00000010'	X'00004110'	The TERMINATE OLC command is rejected, because the command does not apply to the online change state of the command master. TERMINATE OLC is rejected if the command master is not in an online change state. TERMINATE OLC is rejected if the command master has already committed the online change. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000010'	X'0000412C'	The OLCSTAT data set contains the name of an IMS other than the IMS processing the online change. Use DFSUOLC0 to correct the data set.
X'00000010'	X'00004114'	The TERMINATE OLC command failed because of an error accessing the OLCSTAT data set. One or more IMS systems in the IMSplex might still be in an online change state. A DFS2843 message is sent to the OM output exit as unsolicited output. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000010'	X'00004118'	The TERMINATE OLC command failed because of an error allocating the OLCSTAT data set. One or more IMS systems in the IMSplex might still be in an online change state. A DFS2848 message is sent to the OM output exit as unsolicited output. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000010'	X'0000411C'	The TERMINATE OLC command failed because of an error in the OLCSTAT data set contents. One or more of the values is invalid. A DFS2844 message is sent to the OM output exit as unsolicited output.
X'00000010'	X'00004120'	The TERMINATE OLC command is rejected because an online change phase is already in progress on this IMS, which might be INITIATE OLC, TERMINATE OLC, or /DISPLAY MODIFY.
X'00000014'	X'00005000'	The TERMINATE OLC command is rejected because an IMODULE GETSTOR storage request failed.

Table 349. Return and reason codes for the TERMINATE OLC command (continued)

Return code	Reason code	Meaning
X'00000014'	X'00005004'	The TERMINATE OLC command failed because a DFSOCMD response buffer could not be obtained. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000014'	X'00005100'	The TERMINATE OLC command failed because of an RM error. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000014'	X'00005104'	The TERMINATE OLC command failed because of a CQS error. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000014'	X'00005108'	The TERMINATE OLC command failed because of an SCI error. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 940.
X'00000014'	X'00005FFF'	The TERMINATE OLC command failed because of an internal IMS error. One or more IMS systems in the IMSplex might still be in an online change state. For more details, see "TERMINATE OLC error handling" on page 940.

The following table contains the completion codes that can be returned on a TERMINATE OLC command, the meaning of the completion code, and any error text associated with the code.

Table 350. Completion codes for the TERMINATE OLC command

Completion code	Meaning	ERROR TEXT (uppercase)
0	The online change commit or abort phase completed successfully.	
1	The online change type does not apply to this IMS. For example, an ACBLIB online change does not apply to a DCCTL IMS. This IMS does nothing.	
2	The online change phase was not attempted by this IMS for one of the following reasons: • The online change phase master encountered an error and did not direct this IMS to perform the online change phase.	

Table 350. Completion codes for the TERMINATE OLC command (continued)

Completion code	Meaning	ERROR TEXT (uppercase)
3	The online change for this IMS is already completed or terminated. This IMS coordinates the termination of the global online change, but this IMS does not have to do anything locally. The TERMINATE OLC command completed successfully and cleaned up information about the global online change, if there was any. An example of this is the information that RM keeps to manage the global online change.	
58	An IMS is not registered to RM. An OLCSTAT data set contains an IMS that is not registered to RM. Terminate fails for that IMS.	
60	IMODULE GETMAIN storage error.	
61	BCB storage error.	
62	HIOP storage error.	
63	WKAP storage error.	

80	Data set error.	Function (8 char), ddname (8 char), return code (8 bytes), and error detail (8 char).
		
		Function can be one of the following:
		• OPEN
		  Data set open error.
		• READ
		  Data set read error.
		
		DDname can be OLCSTAT.
		
		Return code is the data set service return code.
		
		Reason code is the data set service reason code.

90	Internal error.	Module name that detected internal error (8 char), unused (8 char), return code or function code (8 bytes), and error detail (8 char).

Table 350. Completion codes for the TERMINATE OLC command (continued)

Completion code	Meaning	ERROR TEXT (uppercase)
91	The online change commit phase 2 or abort phase timed out before this IMS responded to the online change commit phase 2 or abort phase. The commit phase 2 or abort might have succeeded on this IMS. Issue QUERY MEMBER TYPE(IMS) to determine the online change state of this IMS.	

Table 350. Completion codes for the TERMINATE OLC command (continued)

Completion code	Meaning	ERROR TEXT (uppercase)
B2	IMS state error.	IMS state error (32 char). The IMS state can be one of the following: • ABENDED This IMS ended abnormally since the last successful online change. Online change is terminated on this IMS. • NOT-REACHABLE The online change phase is rejected because this IMS is NOT-REACHABLE. The SCI on the OS image where this IMS is active is down. Restart the SCI and issue the INITIATE OLC or TERMINATE OLC command again. • OLC ALREADY COMMITTED The online change terminate is rejected because online change is already committed. All IMS systems have completed commit phase 1 and the OLCSTAT data set was updated. • OLC NOT IN PROGRESS The IMS is not in an online change state. The request to terminate the online change does not apply to this IMS. • OLC PHASE IN PROGRESS The online change phase is rejected because this IMS has an online change phase already in progress. • RESTART NOT COMPLETE This IMS initialized before the online change was initiated, but has not completed restart. The online prepare or abort phase is rejected as long as this IMS is in this state. Cancel this IMS, then abort the online change before attempting the online change prepare phase again. • SHUTDOWN This IMS shut down normally since the last successful online change. Online change is terminated on this IMS.

Examples

TSO SPOC input:

```
TERMINATE OLC
```

TSO SPOC output:

```
MbrName  Member    CC
IMS3     IMS2      0
IMS3     IMS3      0
IMS3     SYS3      0
```

OM API input:

```
CMD (TERMINATE OLC)
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 15:49:27.197919</statime>
<stotime>2002.16315:49:27.712209</stotime>
<staseq>B7C4ADFC0D4DF841</staseq>
<stoseq>B7C4ADFC8ADD1F45</stoseq>
<rqsttkn1>USRT011 10084927</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS2    ">
<typ>IMS    </typ>
<styp>DBDC   </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
</mbr>
<mbr name="SYS3    ">
<typ>IMS    </typ>
<styp>DBDC   </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
</mbr>
</cmderr>
<cmd>
<master>IMS3   </master>
<userid>USRT011 </userid>
<verb>TERM</verb>
<kwd>OLC  </kwd>
<input>TERMINATE OLC</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR"    llbl="MbrName" scope="LCL" sort="n" key="0" scroll="yes" len="8"
dtype="CHAR" align="left" />
<hdr slbl="IMSMBR" llbl="Member"  scope="LCL" sort="a" key="1" scroll="no"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"     llbl="CC"      scope="LCL" sort="n" key="0" scroll="yes" len="4"
dtype="INT"  align="right" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(IMS3   ) IMSMBR(SYS3   ) CC(   0) </rsp>
<rsp>MBR(IMS3   ) IMSMBR(IMS2   ) CC(   0) </rsp>
<rsp>MBR(IMS3   ) IMSMBR(IMS3   ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Global online change was aborted for the IMSplex after a successful
INITIATE OLC PHASE(PREPARE) command. Global online change was
successfully terminated.

Related concepts:

Related reference:

TERMINATE OLREORG command

The TERMINATE OLREORG command is used to stop one or more HALDB online reorganizations (OLRs) that are in progress.

The command stops OLR processing at a unit of reorganization boundary, unless the OPTION keyword is specified.

This command supports the type-1 command format and the type-2 command format. The type-1 command format is /TERMINATE OLREORG. The type-1 command response is returned as a DFS0725I pre-edit message. The type-2 command format is TERMINATE OLREORG. The type-2 command response is returned as XML and is available to automation programs.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 952
- "Usage notes" on page 952
- "Output fields" on page 953
- "Return, reason, and completion codes" on page 954
- "Examples" on page 954

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the TERMINATE OLREORG command can be issued.

Table 351. Valid environments for the TERMINATE OLREORG command, keywords, and parameters

Command / Keywords	DB/DC	DBCTL	DCCTL
TERMINATE OLREORG	X	X	
NAME	X	X	
OPTION	X	X	

Syntax

Type-1 command syntax

```
►►─┬─/TERMINATE─┬─OLREORG─NAME─(─┬─partname─┬─)─┬──────────────────────┬─►◄
   └─/TERM──────┘                └─*────────┘   └─OPTION(─┬─ABORT─┬─)──┘
                                                          └─FORCE─┘
```

Type-2 command syntax

```
>>--+--TERMINATE--+--OLREORG--NAME--(--+--partname--+--)------------------->
    +--TERM------+                     |    ,       |
                                       +----*-------+

>--+---------------------------+-------------------------------------------><
   +--OPTION(--+--ABORT--+--)--+
               +--FORCE--+
```
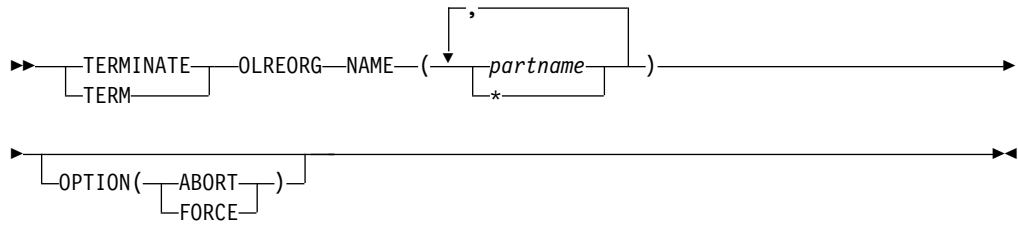
Keywords

The following keywords are valid for the TERMINATE OLREORG command:

NAME()
>   Specifies the names or name of a HALDB partition for which the OLR is to be
>   stopped. You can specify only PHDAM or PHIDAM HALDB partition names.
>   A parameter with the wildcard character (*) is not allowed, except as NAME(*)
>   for all defined HALDB partitions.
>
>   For the type-2 version of this command, you can specify one or more HALDB
>   partition names.
>
>   For the type-1 version of the command, you can specify only one partition
>   name.

OPTION()
>   Allows you to specify the FORCE or ABORT options.
>
>   **ABORT**
>   >   Causes the HALDB OLR to be stopped immediately, possibly with a
>   >   completion code of abend U0474, without waiting for the current
>   >   unit-of-reorganization to complete. Backout may be required depending on
>   >   the state of the online reorganization at termination.
>
>   **FORCE**
>   >   Specifies that the HALDB OLR for the named *part name* is to be stopped
>   >   when the next record boundary is encountered. All of the moved data up
>   >   to that point is committed to DASD and no backout is required.

Attention: If the HALDB OLR is stopped prior to completion, the
OPTION(NODEL) is not retained and must be specified on the INITIATE
OLREORG command that is issued to resume the stopped online reorganization or
on the UPDATE OLREORG command.

Usage notes

If /TERMINATE OLREORG or TERMINATE OLREORG is issued from OM API, it
is treated as a type-2 command. Therefore, if you issue TERM OLREORG as a
type-2 command from an OM API, the only valid command verb form is TERM or
TERMINATE. Similarly, if /TERMINATE OLREORG is issued from a terminal, it is
treated as a type-1 command. Therefore, you can issue /TERMINATE OLREORG
using the first three command characters, such as /TER OLREORG.

If an online reorganization is started with the ALTER option to modify the
structure of the database, IMS reorganizes all of the partitions in the database. In

this case, if you need to stop the reorganization of all of the partitions in the database, specify an asterisk in the NAME keyword, for example TERMINATE OLREORG NAME(*). You cannot specify the name of the HALDB master database. You can use either the type-1 or type-2 format of the TERMINATE OLREORG command to stop alter processing; however, if the altered database is shared, only the type-2 command can be routed to all of the IMS systems that share the database.

This command can be issued to an IMSplex using the Batch SPOC utility.

The TERMINATE OLREORG command causes HALDB OLR to be stopped for the specified HALDB partitions. After a HALDB OLR is terminated, it is no longer active and it does not have an owning IMS. The partitions remain in cursor-active status until the online reorganization is resumed with an INITIATE OLREORG command and completes, or until you run an offline reorganization.

Command responses for /TERMINATE OLREORG

When you issue the /TERMINATE OLREORG command as a type-1 command, the command response is returned in a message format.

When the command completes successfully, the message, DFS0725I, is returned to the system console and to the master terminal with a completion code of 0. If the command results in an error, a non-zero completion code or an error message is returned to the master terminal and system console.

```
DFS0725I INITIATE|UPDATE|TERMINATE OLREORG COMMAND FOR DB dbnamexx COMPLETE.
CC= nn
  where: dbnamexx is the HALDB partition name entered on the command
         nn       is the completion code
```

Output fields

This section describes the responses from the OM API for the TERMINATE OLREORG command. The following table shows the TERMINATE OLREORG output fields. The columns in the table are as follows:

Short label
>   Contains the short label that is generated in the XML output. This field does not apply to the /TERMINATE command.

Show Keyword
>   Identifies the command keyword that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
>   Provides a brief description of the output field.

Table 352. Output fields for TERMINATE OLREORG command

Short label	Show Keyword	Meaning
PART	N/A	Partition name.
MBR	N/A	The IMS that built the command response line.
CC	N/A	Completion code.

Return, reason, and completion codes

The OM return and reason codes that might be returned as a result of the
TERMINATE OLREORG command are standard for all commands that are entered
through the OM API.

The following table includes the return and reason codes and a brief explanation of
the codes. The return or reason code returned for the command might also indicate
an error from a CSL request.

Table 353. Return and reason codes for the TERMINATE OLREORG command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The TERMINATE OLREORG command completed successfully.
X'00000004'	X'00001010'	No matches found for filter.
X'0000000C'	X'00003000'	At least one request was successful.
X'0000000C'	X'00003004'	None of the requests was successful.
X'00000010'	X'00004014'	Command issued on an RSR tracker.
X'00000010'	X'0000400C'	Command issued on an XRF alternate.
X'00000014'	X'00005000'	A GETMAIN error occurred.

The following table includes an explanation of the completion codes. Errors unique
to the processing of TERMINATE OLREORG command are returned as completion
codes. A completion code is returned for each action against a HALDB partition.

Table 354. Completion codes for the TERMINATE OLREORG command

Completion code	Meaning
0	The TERMINATE OLREORG command completed successfully for the partition.
10	Resource name is invalid.
14	Resource is not a partition name.
1C	Resource is a partitioned secondary index.
24	No HALDB OLR is in progress.

Examples

The following are examples of the TERMINATE OLREORG command:

Example 1 for /TERMINATE OLREORG command

Entry ET:
```
/TERM OLREORG NAME(PDHDOKA)
```

Response ET:
```
DFS0725I TERMINATE OLREORG COMMAND FOR DB PDHDOKA  COMPLETE. CC=  24
```

Explanation: The TERM OLREORG command is issued for partition PDHDOKA to
stop the OLR that is in progress. The command is not successful because OLR is
not in progress for the partition.

Example 2 for TERMINATE OLREORG command

TSO SPOC input:

```
TERM OLREORG NAME(PDHDOKA,PDHDOKC)
```

TSO SPOC output:

```
Partition MbrName    CC
    PDHDOKA   IMSA        0
    PDHDOKA   IMS1       24
    PDHDOKC   IMSA       24
    PDHDOKC   IMS1       24
```

OM API input:

```
CMD (   TERM OLREORG NAME(PDHDOKA,PDHDOKC))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.168 21:31:13.035976</statime>
<stotime>2003.168 21:31:13.038227</stotime>
<staseq>B9962C747D6C8868</staseq>
<stoseq>B9962C747DF93586</stoseq>
<rqsttkn1>USRT005 10143113</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
</ctl>
<cmderr>
<mbr name="IMSA    ">
<typ>IMS     </typ>
<styp>DBCTL   </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntext>At least one request successful</rsntext>
</mbr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003004</rsn>
<rsntext>At least one request successful</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMSA    </master>
<userid>USRT005 </userid>
<verb>TERM</verb>
<kwd>OLREORG         </kwd>
<input>TERM OLREORG NAME(PDHDOKA,PDHDOKC) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PART" llbl="Partition" scope="LCL" sort="A" key="1"
 scroll="NO" len="9" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="N" key="0" scroll="NO"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="N" key="0" scroll="YES"
 len="4" dtype="INT" align="right" />
</cmdrsphdr>
<cmdrspdata>
<rsp> PART(PDHDOKA ) MBR(IMSA    ) CC(    0) </rsp>
<rsp> PART(PDHDOKC ) MBR(IMSA    ) CC(   24) </rsp>
```

```
<rsp> PART(PDHDOKA ) MBR(IMS1    ) CC(  24) </rsp>
<rsp> PART(PDHDOKC ) MBR(IMS1    ) CC(  24) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The TERM OLREORG command is issued to stop the OLR for partitions PDHDOKA and PDHDOKC. The command is routed to IMSA and IMS1. The command is successful for partition PDHDOKA at IMSA, where OLR is in progress. The command is not successful for PDHDOKC at IMSA because OLR is not in progress for PDHDOKC on IMSA. The command is not successful for either of the two partitions at IMS1, because OLR is not in progress for PDHDOKA and PDHDOKC on IMS1.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

Chapter 27. /TEST command

Use the /TEST command to place a terminal or a user into either test mode or MFSTEST mode.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 958
- "Examples" on page 959

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 355. Valid environments for the /TEST command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TEST	X		X
LINE	X		X
NODE	X		X
PTERM	X		X
USER	X		X

Syntax
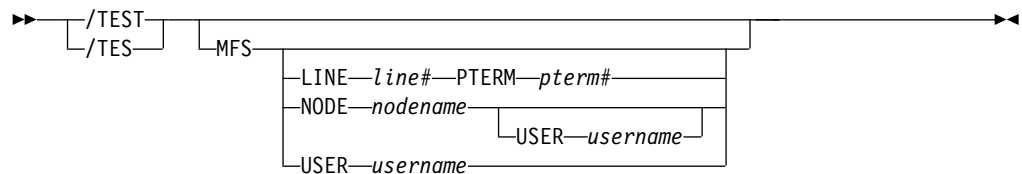
Keywords

The following keywords are valid for the /TEST command:

MFS
> Specifies MFS test mode for the terminal or user. The MFS parameter is valid only for terminals supported by Message Format Service (MFS). When the /TEST MFS USER command is issued for a dynamic user, it is not possible to determine if the MFSTEST mode is valid until the user signs on to a terminal. Once the user signs on to a terminal, a check is made to determine whether that terminal supports MFSTEST mode. If the terminal does not support MFSTEST mode, the mode is not propagated to the terminal and is removed from the USER unless another /TEST MFS USER command is issued.

LINE, PTERM
Specifies the non-VTAM line and pterm to place into MFSTEST mode.

NODE
Specifies the VTAM node to place into MFSTEST mode.

The /TEST MFS NODE form of the command is valid only for statically defined nodes.

Restrictions for using NODE and USER parameters together:
- Commands with the NODE USER keyword pair are valid only if:
  - The USER is signed on to the NODE
  - In an ISC environment, the USER is allocated to the NODE
  - The nodes and users already exist
- /TEST MFS NODE USER commands are valid for ISC, LUP, and 3600 nodes only. For ISC, the /TEST MFS NODE *nodename* USER *username* form of the command is supported for ISC nodes and applies to the half-session allocated to the USER username.

USER
When specified without the NODE keyword, USER specifies the dynamic user to place into MFSTEST mode. MFSTEST mode status is kept from one signon to another. For example, if a user issues a /TEST MFS command, signs off, and then signs on again at another terminal, the MFSTEST mode is still in effect. If the user does not exist, it is created and MFSTEST mode is set.

Usage notes

This command can be issued to an IMSplex using the Batch SPOC utility.

In test or echo mode, any input from the terminal is sent back. Input and output errors are not checked, and IMS error notification procedures are bypassed. Echo mode continues until reset with an /END, /STOP LINE, /STOP LINE PTERM, or a /STOP NODE command. The /DISPLAY command identifies a terminal or user in test mode or MFSTEST mode. If no keywords are supplied, the terminal entering the command is placed into test mode or MFSTEST mode.

Test mode is not a command significant status, so the commands to set test mode are not recoverable nor are they kept after signons and can only be set by the end user or terminal, not remotely by an operator. The /TEST LINE, /TEST NODE, and /TEST USER commands, which set test mode remotely, are no longer supported.

In MFSTEST mode, terminals supported by the Message Format Service use format blocks from a special test library if the requested format block is in the test library; otherwise, the blocks are obtained from the production library. MFSTEST mode continues until reset with an /END command. Certain error conditions can occur that cause MFSTEST mode to terminate. If an error condition occurs, the terminal operator receives an error message.

MFSTEST mode is a command significant status, is recoverable and is remembered across logons and signons. For example, if a /TEST MFS NODE command is entered at a node, the node logs off and logs back on at another terminal, MFSTEST mode is still in effect. If a dynamic user issues a /TEST MFS USER command, signs off, and then signs on again at another terminal, MFSTEST mode is still in effect.

/TEST MFS NODE applies to dynamic nodes in addition to static nodes because MFSTEST mode is associated with dynamic nodes as well as dynamic users. /TEST MFS NODE and /TEST MFS NODE USER set MFSTEST mode at the node level. /TEST MFS USER sets MFSTEST at the user level. /TEST MFS with no keywords sets MFSTEST at the node level for static terminals (they have no user level) and at the user level for dynamic terminals.

The /TEST NODE USER command is supported for static and dynamic ISC sessions. For ISC, /TEST MFS NODE USER is required. You cannot use /TEST MFS NODE (without USER) for ISC and have it apply to all of the half-sessions.

If global resource information is kept in Resource Manager, MFSTEST mode is set globally. If global resource information is not kept in Resource Manager, the resource does not exist, and ETO is enabled, the resource (node or user) is created and MFSTEST mode is set. If a temporary node is dynamically created to hold command status, and the temporary node has MFSTEST status, then, when a logon occurs for the node, the MFSTEST status is set for the logged-on node. If the node logging on is an ISC parallel session, MFSTEST is set only for the first half-session that is logged on. Subsequent ISC half-sessions will not be put into MFSTEST mode.

Examples

The following are examples of the /TEST command:

Example 1 for /TEST command

Entry ET:
```
/TEST
```

Response ET:
```
DFS058I  TEST COMMAND COMPLETED
```

Entry ET:
```
NOW IS THE TIME TO COME TO THE AID
```

Response ET:
```
NOW IS THE TIME TO COME TO THE AID
```

Explanation: The entering terminal is placed in echo mode and continues to receive message input as output until test mode is terminated.

Example 2 for /TEST command

Entry ET:
```
/TEST MFS
```

Response ET:
```
DFS058I  TEST COMMAND COMPLETED
```

Explanation: The entering terminal is placed into MFSTEST mode.

Chapter 28. /TRACE commands

Use the /TRACE commands to direct and to control the IMS capabilities for tracing internal IMS events. They also start, stop, and define the activity to be monitored by the IMS Monitor.

The information developed by the LINE, LINK, NODE, UNITYPE, TRANSACTION, PROGRAM, PSB, and TCO keywords is written on the IMS system log (type X'5F' for PSB, and type X'67' records for the other keywords mentioned.) PI (program isolation) and TABLE trace information is kept in storage or logged (type X'67' records), depending on specification of additional keywords. The MONITOR keyword provides no such output on the system log; it only controls the IMS Monitor. The monitor develops its own output data and writes it on a separate data set.

This command can be issued to an IMSplex using the Batch SPOC utility.

The status and options of the current IMS traces can be displayed with the /DISPLAY TRACE command.

Subsections:
Related concepts:

➦ Using IMS reports (System Administration)

Related reference:

/TRACE EXIT command

Use the /TRACE EXIT command to identify user exit tracing.

Subsections:
* "Environment"
* "Syntax"
* "Keywords"
* "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 356. Valid environments for the /TRACE EXIT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
EXIT	X		X
SET	X	X	X

Syntax

```
>>--+--/TRACE--+--SET--+--ON--+--EXIT--+--ALL----+--DFSMSCE0--------------------><
    +--/TRA----+       +--OFF-+        +--LRDI---+
                                       +--LRIN---+
                                       +--LRLT---+
                                       +--LRTR---+
                                       +--PRCH---+
                                       +--PRGU---+
                                       +--PRIS---+
                                       +--TR62---+
                                       +--TRBT---+
                                       +--TROT---+
                                       +--TRVT---+
```
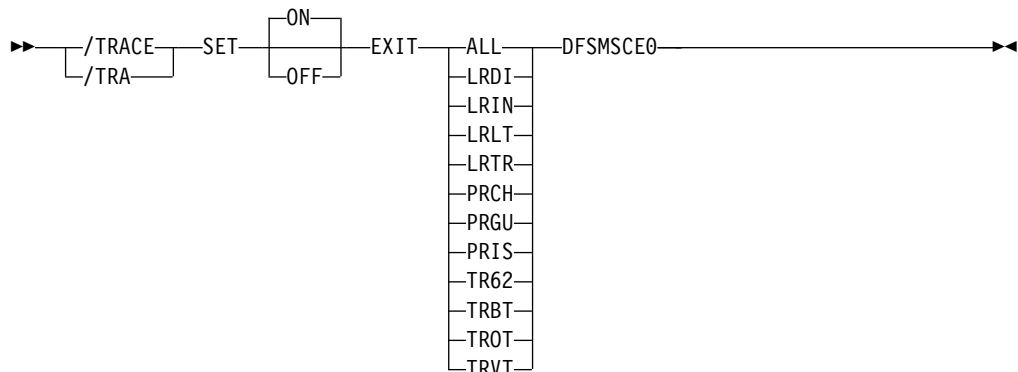
Keywords

The following keyword is valid for the /TRACE EXIT command:

DFSMSCE0
> The TM and MSC Message Routing and Control user exit. When this keyword is used on the /TRACE EXIT command, the traces causes IMS to write type 6701-MSEA and 6701-MSEB log records to the log data set when the exit routine is called. A 6701-MSEA record is logged when the exit is called if the trace is active for the entry point. A 6701-MSEB record is logged when the exit returns to IMS.

Usage notes

When using the EXIT keyword, you must specify one of the following parameters to turn on or off:

ALL The trace is turned on or off for all entry points.

LRDI Calls the Link Receive Direct Routing exit entry point.

LRIN Calls the Link Receive Intermediate exit entry point.

LRLT Calls the Link Receive LTERM exit entry point.

LRTR Calls the Link Receive Transaction exit entry point.

PRCH Calls the Program Routing CHNG Call exit entry point.

PRGU Calls the Program Routing GU Call exit entry point.

PRIS Calls the Program Routing ISRT Call exit entry point.

TR62 Calls the Terminal Routing LU62 exit entry point.

TRBT Calls the Terminal Routing non-VTAM exit entry point.

TROT Calls the Terminal Routing OTMA exit entry point.

TRVT Calls the Terminal Routing VTAM exit entry point.

/TRACE LINE command

Use the /TRACE LINE command to cause events related to the lines to be traced.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 964
- "Examples" on page 965

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 357. Valid environments for the /TRACE LINE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
LEVEL	X		X
LINE	X		X
MODULE	X		X
SET	X	X	X
TAKEOVER	X		X

Syntax

A:

```
├──────┬─────────────────────┬──┬──────────────┬──────────────────────────┤
       └─MODULE──┬─DDM─┬─────┘  └─TAKEOVER─┘
                 ├─MFS─┤
                 └─ALL─┘
```

Keywords

The following keywords are valid for the /TRACE LINE command:

LEVEL

Expands the LINE, LINK, NODE, or UNITYPE trace functions. The LEVEL specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

LEVEL indicates the extent of the control block trace information desired. The indicated control blocks are only traced at relevant times. All levels are inclusive of numerically lower levels. The following list displays the levels and their associated blocks.

Level	Blocks
1	CLB (DECB) or LLB(MSC)
	CTB or LTB(MSC)
	IOB (for non-VTAM lines) or IOSB (MSC for channel-to-channel links)
2	CNT or LNB(MSC)
	CXB
	CRB
	CIB
	CCB
	PD stack
3	queue manager buffers
	Input/output line buffers
	LXB (for channel-to-channel links and processor storage to processor storage)
4	save area sets (IMS dispatching)

If the first /TRACE SET ON command does not specify LEVEL, a default of 4 will be used. Specifying LEVEL on subsequent commands will change the defaults.

MODULE

Is used to expand the LINE, LINK, NODE, or UNITYPE trace functions. The MODULE specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

MODULE indicates which modules are to have their control blocks traced.

ALL   Both device-dependent module (DDM) and MFS

DDM   Communication analyzer and device-dependent module interfaces

MFS Communication analyzer and Message Format Service (MFS) module
interfaces

If the first /TRACE SET ON command does not specify MODULE, a default of
ALL will be used. Specifying MODULE on subsequent commands will change
the defaults.

TAKEOVER

Controls tracing during takeover only, and is separate from regular tracing.
TAKEOVER tracing can be set for LINE, LINK, NODE, and UNITYPE
keywords.

TAKEOVER only applies in an XRF environment. When TAKEOVER is used
with SET OFF, the trace is turned off before takeover. When an output message
is dequeued for a terminal, takeover tracing will stop for that terminal.

If both regular and takeover tracing are entered, the most recent setting will
override any previous settings; for example, takeover tracing will override
regular tracing if regular tracing was entered first. This means that either
regular or TAKEOVER tracing can be in effect, but not both.

/TRACE TAKEOVER can be issued only from an XRF active system. It is
rejected if entered from an alternate or non-XRF system. It is recovered across
restart and takeover, and only needs to be entered once until cold start. Tracing
occurs only if the session was active at the time of the takeover.

Examples

The following is an example of the /TRACE LINE command:

To turn on the Message Format Service module and communication analyzer level
4 control block tracing for all physical terminals on line 4.

Entry ET:
```
/TRACE SET ON LINE 4 LEVEL 4 MODULE MFS
```

Response ET:
```
DFS058I  TRACE COMMAND COMPLETED
```

/TRACE LINK command

Use the /TRACE LINK command to cause events related to the logical links to be
traced.

Subsections:
- "Environment"
- "Syntax" on page 966
- "Keywords" on page 966
- "Examples" on page 967

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 358. Valid environments for the /TRACE LINK command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
LEVEL	X		X
LINK	X		X
MODULE	X		X
SET	X	X	X
TAKEOVER	X		X

Syntax

A:
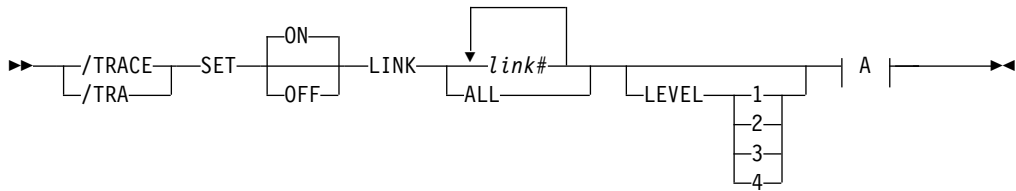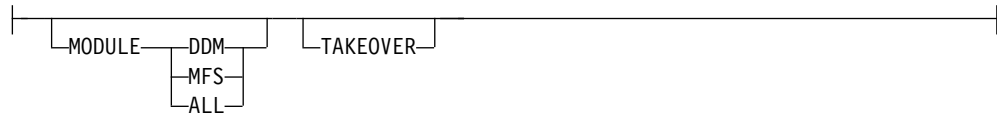
Keywords

The following keywords are valid for the /TRACE LINK command:

LEVEL

Expands the LINE, LINK, NODE, or UNITYPE trace functions. The LEVEL specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

LEVEL indicates the extent of the control block trace information desired. The indicated control blocks are only traced at relevant times. All levels are inclusive of numerically lower levels. The following list displays the levels and their associated blocks.

Level   Blocks

1   CLB (DECB) or LLB(MSC)

          CTB or LTB(MSC)

          IOB (for non-VTAM lines) or IOSB (MSC for channel-to-channel links)

2   CNT or LNB(MSC)

          CXB

          CRB

          CIB

          CCB

PD stack

3   queue manager buffers

Input/output line buffers

LXB (for channel-to-channel links and processor storage to processor storage)

4   save area sets (IMS dispatching)

If the first /TRACE SET ON command does not specify LEVEL, a default of 4 will be used. Specifying LEVEL on subsequent commands will change the defaults.

MODULE

Is used to expand the LINE, LINK, NODE, or UNITYPE trace functions. The MODULE specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

MODULE indicates which modules are to have their control blocks traced.

ALL   Both device-dependent module (DDM) and MFS

DDM   Communication analyzer and device-dependent module interfaces

MFS   Communication analyzer and Message Format Service module interfaces

If the first /TRACE SET ON command does not specify MODULE, a default of ALL will be used. Specifying MODULE on subsequent commands will change the defaults.

TAKEOVER

Controls tracing during takeover only, and is separate from regular tracing. TAKEOVER tracing can be set for LINE, LINK, NODE, and UNITYPE keywords.

TAKEOVER only applies in an XRF environment. When TAKEOVER is used with SET OFF, the trace is turned off before takeover. When an output message is dequeued for a terminal, takeover tracing will stop for that terminal.

If both regular and takeover tracing are entered, the most recent setting will override any previous settings; for example, takeover tracing will override regular tracing if regular tracing was entered first. This means that either regular or TAKEOVER tracing can be in effect, but not both.

/TRACE TAKEOVER can be issued only from an XRF active system. It is rejected if entered from an alternate or non-XRF system. It is recovered across restart and takeover, and only needs to be entered once until cold start. Tracing occurs only if the session was active at the time of the takeover.

Examples

The following is an example of the /TRACE LINK command:

To turn on tracing for a logical link:

Entry ET:
```
/TRACE SET ON LINK 2 LEVEL 4 MODULE ALL
```

Response ET:

```
DFS058I  TRACE COMMAND COMPLETED
```

/TRACE LUNAME command

Use the /TRACE LUNAME command to activate and deactivate tracing for a particular LU name or TP name of the LU name.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 359. Valid environments for the /TRACE LUNAME command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
INPUT	X		X
LUNAME	X		X
OUTPUT	X		X
SET	X	X	X

Syntax
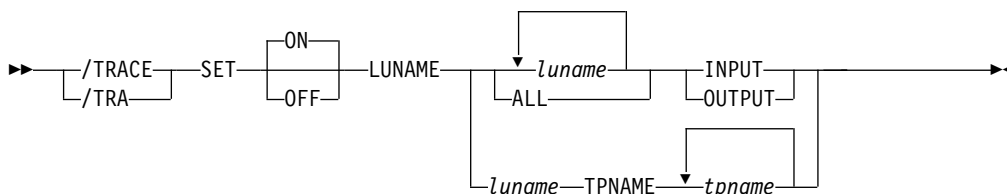
Usage notes

Specifying this command causes trace entries to be written to the LUMI trace table. For this reason, the /TRACE SET ON TABLE LUMI command must be entered first in order to create the table for trace entries that will be created by subsequent /TRACE SET ON LUNAME commands. A trace entry is written:
- On LU 6.2 module entries/exits
- When APPC calls are made
- When errors are encountered

The INPUT and OUTPUT keywords provide the operator with the flexibility to control the volume of trace data for LU 6.2 devices.

Specifying the keyword INPUT with the LUNAME keyword indicates tracing is activated or deactivated for input and synchronous outbound activities. Specifying

the parameter ALL with the INPUT keyword causes all future LU 6.2 input and synchronous outbound activities to be traced as well.

Specifying the keyword OUTPUT with the LUNAME keyword indicates tracing is activated or deactivated for asynchronous outbound activities. Specifying the parameter ALL with OUTPUT causes all future LU 6.2 asynchronous outbound activities to be traced as well.

Specifying neither INPUT or OUTPUT is the same as both INPUT and Tracing is activated or deactivated for input and both synchronous and asynchronous outbound activities. Specifying the parameter ALL in this case causes all future LU 6.2 inbound activities, synchronous and asynchronous outbound activities to be traced as well.

The network-qualified LU name is optional for the LUNAME keyword. If the LU name is not a network-qualified LU name and no TP name is specified, tracing is activated or deactivated for all the network-qualified LU names in the system whose LU name matches the LU name specified.

If the specified resource does not exist and tracing is activated, a structure is created to retain the status.

/TRACE MONITOR command

Use the /TRACE MONITOR command to activate or deactivate the IMS Monitor.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 970
- "Usage notes" on page 971
- "Examples" on page 972

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 360. Valid environments for the /TRACE MONITOR command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
MONITOR	X	X	X
SET	X	X	X

Syntax

```
►►──┬─/TRACE─┬──SET──┬──ON──┬──MONITOR──┤ B ├────────────────────►◄
    └─/TRA───┘       └─OFF─┘
```

B:

```
                      ┌──ALL──────────────┐
        ┌─◄──────────┐│                   │
├───────┼──APDB───────┴───────────────────┴──────────────────────►
        │              ┌─◄────────────┐
        │              │ ┌─dbname──────┤
        │              └─┼─areaname────┤
        │                └─partitionname┘
        │
        ├──APMQ────────────────────────────┐
        ├──LA──────────────────────────────┤
        ├──SCHD────────────────────────────┤
        └──ALL──────────────────────────────┘

►──┬────────────────────────────────────┬──┬──────────────────────┬──►
   │            ┌─◄──────────┐           │  └─INTERVAL──#seconds──┘
   └──REGION──┬─┴─reg#────────┴──────────┤
              ├───reg#─reg#──────────────┤
              ├───regionname─────────────┤
              └──ALL─────────────────────┘
```
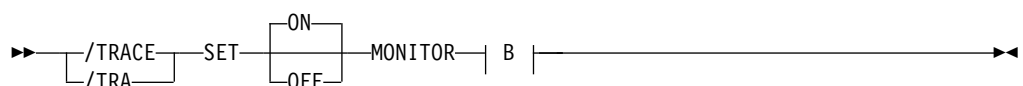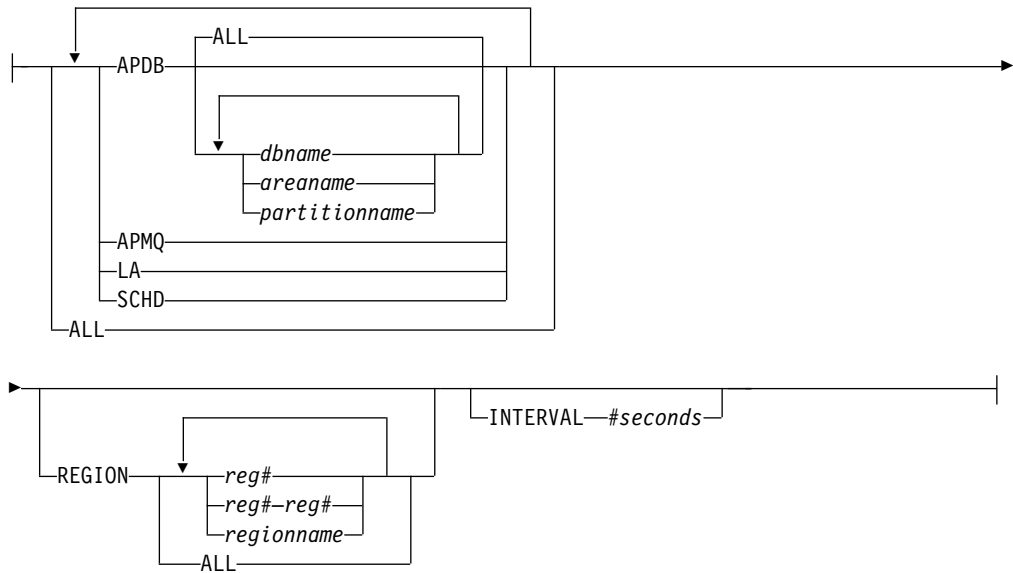
Keywords

When activating the monitor, you must specify one or more of the following MONITOR parameters to indicate the events to be monitored:

ALL
Monitors all of the activity in this list.

APDB
Monitors activity between application programs and databases, including Fast Path activity. Monitoring includes all application program requests to external subsystem databases. Monitoring can optionally be limited to a subset of full-function databases or partitions, Fast Path DEDBs and MSDBs, and the areas comprising those DEDBs by specifying database names or area names.

You can specify the ALL parameter to indicate all databases, areas, and partitions, or you can explicitly enter database names, area names, and partition names.

APMQ
Monitors activity between application programs and message queues, including Fast Path activity.

INTERVAL
Monitors events for a fixed interval of time, entered in seconds. INTERVAL defines the period of time after which no monitor log records will be written.

The duration of the monitoring must be less than twenty-four hours (86,400 seconds). When INTERVAL is not specified, monitoring will continue until the /TRACE SET OFF MONITOR command is issued, or until IMS shuts down.

INTERVAL does not define when the Monitor will be turned off, because the IMS Monitor will not be turned off until the first attempt is made to write a monitor log record after the defined interval has expired.

LA Monitors line and logical link events.

REGION

Monitors events related to specific dependent regions. The regions might or might not currently be active. Each region can be specified as:

- A region number from 1 to 999 (*reg#*). The number cannot exceed the MAXPST with which IMS was brought up.
- A range of region numbers from 1 to 999 (*reg#–reg#*). The number cannot exceed the MAXPST with which IMS was brought up.
- A region name (*regionname*).

If REGION is not specified, or when REGION ALL is specified, the activities of all dependent regions are monitored.

SCHD

Monitors scheduling and termination events, including Fast Path activities.

Usage notes

You can specify any combination of ALL, APDB, APMQ, INTERVAL, LA, REGION, and SCHD parameters on the MONITOR keyword, as shown in the following table.

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the MONITOR keyword parameters can be issued.

Table 361. MONITOR keyword parameter environments

Keyword parameter	DB/DC	DBCTL	DCCTL
ALL	X	X	X
APDB	X	X	
APMQ	X		X
INTERVAL	X	X	X
LA	X		X
REGION	X	X	X
SCHD	X	X	X

The IMS Monitor report output varies depending upon which keywords or parameters you specify. Sections of the report can be misleading if required records are excluded. For example, if you specify the APDB parameter without the SCHD parameter, PSB/PCB relationships will not be correctly represented. To get the total DL/I call reports without the IMS line activity, the correct parameters to specify are APDB, APMQ, and SCHD.

The monitor writes log records until one of the following occurs:
- /TRACE SET OFF MONITOR is entered.
- The time interval specified by the INTERVAL parameter is reached.
- IMS is shut down.

When deactivating the monitor, no parameters are required. Any parameters that are entered, other than ALL, are ignored. ALL is the default. If the monitor is to be reactivated, a new set of MONITOR keyword parameters must be selected.

The APDB *dbname* option cannot be used to filter the TYPE62 monitor log records by database names.

Examples

The following are examples of the /TRACE MONITOR command:

Entry ET:
```
/TRACE SET ON MONITOR ALL
```

Response ET:
```
DFS058I TRACE COMMAND COMPLETED
DFS2500I DATASET DFSDCMON SUCCESSFULLY ALLOCATED
DFS2212I DC MONITOR STARTED
```

Explanation: The monitor is activated and all events will be monitored.

Entry ET:
```
/TRACE SET OFF MONITOR
```

Response ET:
```
DFS058I TRACE COMMAND COMPLETED
DFS2500I DATASET DFSDCMON SUCCESSFULY DEALLOCATED
DFS2212I DC MONITOR STOPPED
```

Explanation: The monitor is deactivated.

Entry ET:
```
/TRACE SET ON MONITOR LA
```

Response ET:
```
DFS058I TRACE COMMAND COMPLETED
```

Explanation: The monitor is activated. Line and logical link activities will be monitored. The monitoring of events from the previous activation of the monitor no longer apply.

/TRACE NODE command

Use the /TRACE NODE command to cause events related to the node or nodes to be traced.

Subsections:
- "Environment"
- "Syntax" on page 973
- "Keywords" on page 973
- "Usage notes" on page 974
- "Examples" on page 975

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 362. Valid environments for the /TRACE NODE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X

Table 362. Valid environments for the /TRACE NODE command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
LEVEL	X		X
MODULE	X		X
NODE	X		X
SET	X	X	X
TAKEOVER	X		X
USER	X		X

Syntax

A:
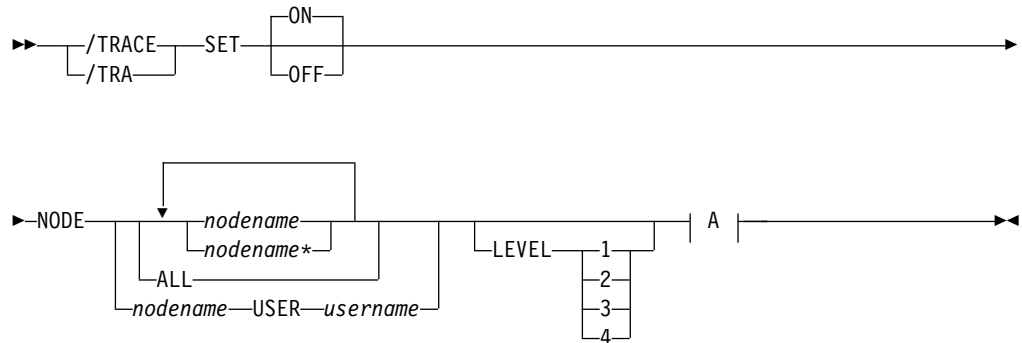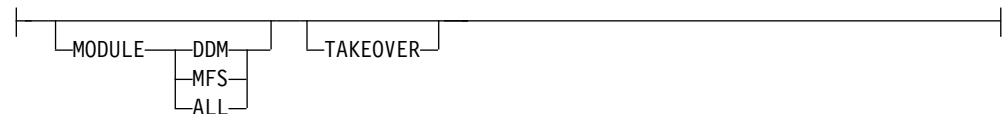
Keywords

The following keywords are valid for the /TRACE NODE command:

LEVEL

> Expands the LINE, LINK, NODE, or UNITYPE trace functions. The LEVEL specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.
>
> LEVEL indicates the extent of the control block trace information desired. The indicated control blocks are only traced at relevant times. All levels are inclusive of numerically lower levels. The following list displays the levels and their associated blocks.

> **Level** **Blocks**

> **1** CLB (DECB) or LLB(MSC)

>   CTB or LTB(MSC)

>   IOB (for non-VTAM lines) or IOSB (MSC for channel-to-channel links)

> **2** CNT or LNB(MSC)

>   CXB

CRB

CIB

CCB

PD stack

3   queue manager buffers

Input/output line buffers

LXB (for channel-to-channel links and processor storage to processor storage)

4   save area sets (IMS dispatching)

If the first /TRACE SET ON command does not specify LEVEL, a default of 4 will be used. Specifying LEVEL on subsequent commands will change the defaults.

MODULE

Is used to expand the LINE, LINK, NODE, or UNITYPE trace functions. The MODULE specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

MODULE indicates which modules are to have their control blocks traced.

ALL   Both device-dependent module (DDM) and MFS

DDM Communication analyzer and device-dependent module interfaces

MFS   Communication analyzer and Message Format Service module interfaces

If the first /TRACE SET ON command does not specify MODULE, a default of ALL will be used. Specifying MODULE on subsequent commands will change the defaults.

TAKEOVER

Controls tracing during takeover only, and is separate from regular tracing. TAKEOVER tracing can be set for LINE, LINK, NODE, and UNITYPE keywords.

TAKEOVER only applies in an XRF environment. When TAKEOVER is used with SET OFF, the trace is turned off before takeover. When an output message is dequeued for a terminal, takeover tracing will stop for that terminal.

If both regular and takeover tracing are entered, the most recent setting will override any previous settings; for example, takeover tracing will override regular tracing if regular tracing was entered first. This means that either regular or TAKEOVER tracing can be in effect, but not both.

/TRACE TAKEOVER can be issued only from an XRF active system. It is rejected if entered from an alternate or non-XRF system. It is recovered across restart and takeover, and only needs to be entered once until cold start. Tracing occurs only if the session was active at the time of the takeover.

Usage notes

The NODE parameter can be generic if the USER keyword is not specified and applies to nodes that already exist. Generic NODE parameters do not cause any dynamic nodes to be created.

For ISC nodes, the /TRACE NODE nodename without the USER applies to all half-sessions for NODE nodename, including dynamic ISC sessions that are dynamically allocated later.

For nodes that do not exist, /TRACE SET ON NODE *nodename* without the USER keyword causes the dynamic NODE nodename to be created to maintain knowledge of the trace request when the node becomes active. Until the node becomes active, /DISPLAY NODE shows a type of UNK (unknown). If the trace is subsequently turned off and the temporary node still exists, it is deleted at the next checkpoint.

Restrictions for using NODE and USER parameters together:
- Commands with the NODE USER keyword pair are valid only if:
    - The USER is signed on to the NODE
    - In an ISC environment, the USER is allocated to the NODE
    - The nodes and users already exist
- /TRACE NODE USER commands are valid for ISC and non-ISC nodes and users.

If global resource information is kept in Resource Manager, /TRACE NODE sets a global trace status for the node and sets the trace status locally. If global resource information is not kept in Resource Manager, /TRACE NODE sets the trace status locally. If the node does not exist in an ETO environment, IMS creates the node and sets trace status for the local node.

Examples

The following are examples of the /TRACE NODE command:

The following example shows how IMS creates temporary nodes to retain trace status data.

The /TRACE command is issued for a dynamic ISC NODE that does not yet exist, DTSLU607, causing a temporary node to be created to retain the trace status. Once the ISC parallel sessions IMSUS01 and IMSUS02 are allocated, the trace status is applied to them both.

Entry ET:
```
  /TRACE SET ON NODE DTSLU607
```

Response ET:
```
  DFS058 TRACE COMMAND COMPLETED
```

Entry ET:
```
  /DISPLAY NODE DTSLU607
```

Response ET:
```
NODE-USR TYPE   CID        RECD ENQCT DEQCT   QCT  SENT
DTSLU607 UNK    00000000      0     0     0     0     0 TRA
*90127/091634*
```

Entry ET:
```
  /DISPLAY NODE DTSLU607
```

Response ET:
```
NODE-USR TYPE   CID      RECD ENQCT DEQCT  QCT  SENT
DTSLU607 LUT6
-N/A     UNK    00000000   0    0     0    0     0 TRA
-IMSUS01        01000002   0    0     0    0     0 SIGN(IMSUS01 )
                                                  IDLE CON TRA PRI
-IMSUS02        01000004   0    0     0    0     0 SIGN(IMSUS02 )
                                                  IDLE CON TRA PRI

*90127/091432*
```

In the following example, the /TRACE command is issued for ISC node
LU6NDPA. In the response ET, the value TCP in the TYPE column indicates that
the ISC node uses TCP/IP, not VTAM.

Entry ET:
```
 /TRACE SET ON NODE LU6NDPA
```

Response ET:
```
DFS058I 12:28:44 TRACE COMMAND COMPLETED   IMS1
```

Entry ET:
```
/DISPLAY NODE LU6NDPA
```

Response ET:
```
DFS000I      NODE-USR TYPE   CID      RECD ENQCT DEQCT  QCT  SENT
DFS000I      LU6NDPA  TCP
DFS000I      -N/A            ********   0    0     0    0     0 IDLE
TRA STATIC   IMS1
DFS000I      -N/A            ********   0    0     0    0     0 IDLE
TRA STATIC   IMS1
DFS000I      -N/A            ********   0    0     0    0     0 IDLE
TRA STATIC   IMS1
DFS000I      *12065/122852*   IMS1
```

/TRACE OSAMGTF command

Use the /TRACE SET OSAMGTF command to activate or deactivate the OSAM
Buffer Handle GTF (Generalized Trace Facility) subroutine. When activated, the
OSAMGTF trace generates the z/OS Generalized Trace Facility trace entries for all
the IMS requests to the OSAM Buffer Handler.

Subsections:
- "Environment"
- "Syntax" on page 977

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 363. Valid environments for the /TRACE OSAMGTF command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
OSAMGTF	X	X	
SET	X	X	X

Syntax

```
►►──┬─/TRACE─┬──SET──┬──ON──┬──OSAMGTF──────────────────────────────────────────◄◄
    └─/TRA───┘       └─OFF──┘
```

/TRACE PI command

Use the /TRACE PI command to cause program isolation trace entries to be written to a trace table.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 978
- "Examples" on page 979

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 364. Valid environments for the /TRACE PI command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
OPTION	X	X	X
PI	X	X	
SET	X	X	X
VOLUME	X	X	X

Syntax

```
►►──┬─/TRACE─┬──SET──┬──ON──┬──PI─┬─────────────────────┬─┬──────────────────────────┬──◄◄
    └─/TRA───┘       └─OFF──┘     │        ┌─LOG─┐       │ │        ┌─MEDIUM─┐        │
                                 └─OPTION──┼─TIME─┼──────┘ └─VOLUME──┼─LOW────┼───────┘
                                          └─ALL──┘                  └─HIGH───┘
```

Keywords

The following keywords are valid for the /TRACE PI command:

OPTION
Indicates a request for one of the following program isolation trace options:

ALL   Both LOG and TIME.

LOG   Requests that traced data be written to the system log. If SET OFF, tracing continues but no buffers are transferred to the system log. LOG is the default.

The log option includes the possibility of externally tracing to a data set other than the IMS OLDS. If specified, DASD external tracing has first priority, TAPE external tracing has second priority, and IMS OLDS has third priority. External tracing to the OLDS is not done without operator approval. EXTERNAL trace is available to the alternate system only for DASD/TAPE type, but not for OLDS.

TIME Requests that an additional time field be included in each ENQ/DEQ request trace record if a WAIT was needed. This field will contain elapsed wait time. If set OFF, tracing continues but only the time of day is recorded.

Entries for Fast Path have no elapsed wait time.

VOLUME
Specifies the volume of entries to be written to the PI trace table: LOW volume, MEDIUM volume (default), or HIGH volume.

Usage notes

PI trace entries are written in the same trace table as DL/I and lock activity trace entries. A PI trace entry contains information about program isolation ENQ/DEQ calls and DL/I calls. The trace entry created by /TRACE TABLE DLI contains different information about DL/I calls and is written as a separate entry in the same trace table. Starting the LOCK trace also causes PI tracing to occur.

If PI is entered without the OPTION keyword, the program isolation trace is kept in storage without being logged. If you are using the program isolation trace to provide statistics and performance data, you should enter OPTION(ALL).

The following table lists various /TRACE command formats and shows whether the command influences tracing, logging, and the additional time field.

Table 365. /TRACE command formats

Command	Tracing	Logging	Additional time field
/TRACE SET ON PI	Yes	No	No
/TRACE SET OFF PI	No	No	No
/TRACE SET ON PI OPTION[1]	Yes	Yes	No
/TRACE SET OFF PI OPTION[1]	Yes	No	No
/TRACE SET ON PI OPTION TIME	Yes	No	Yes
/TRACE SET OFF PI OPTION TIME	Yes	No	No
/TRACE SET ON PI OPTION ALL	Yes	Yes	Yes
/TRACE SET OFF PI OPTION ALL	Yes	No	No

Note:

1. This is the same command as /TRACE SET ON | OFF PI OPTION LOG.

Examples

The following are examples of the /TRACE PI command:

Example 1

To turn on program isolation tracing, include the additional time field in the trace record and have the trace information logged:

Entry ET:
```
/TRACE SET PI OPTION ALL
```

Response ET:
```
DFS058I  TRACE COMMAND COMPLETED
```

Example 2

To turn off logging of program isolation trace data but continue the trace in storage:

Entry ET:
```
/TRACE SET OFF PI OPTION LOG
```

Response ET:
```
DFS058I  TRACE COMMAND COMPLETED
```

Example 3

To stop program isolation tracing.

Entry ET:
```
/TRACE SET OFF PI
```

Response ET:
```
DFS058I  TRACE COMMAND COMPLETED
```

/TRACE PGM command

Use the /TRACE PGM command to trace the DL/I portion of Data Communications (DC) for a specific program.

Subsections:
- "Environment"
- "Syntax" on page 980
- "Usage notes" on page 980
- "Equivalent IMS type-2 commands" on page 981

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 366. Valid environments for the /TRACE PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
PGM	X	X	X
SET	X	X	X

Syntax
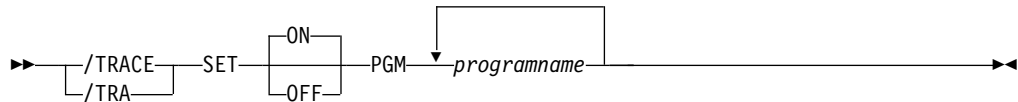
```
>>--+--/TRACE--+--SET--+--ON--+--PGM--+<--------------+--programname--------------------------><
    +--/TRA----+       +--OFF-+       +--------------+
```

Usage notes

Each DL/I call to a TPPCB, issued by the user application program, is traced on entry to and exit from the DC call handler DFSDLA30. On entry to DFSDLA30 a type 6701-LA3A record is written, on exit from DFSDLA30 a type 6701-LA3B record is written.

Each record will contain the following items if applicable:
- TPPCB
- Up to 64 bytes of the I/O area
- SMB
- PST

If the batch message program (BMP) being traced is IBM IMS Queue Control Facility for z/OS, a 6701-MRQB record is logged by the IMS Queue Control Facility module DFSQMRQ0. The default program name for the IMS Queue Control Facility BMP is MRQPSB, and can be overridden on the MSGQUEUE system definition macro.

Items logged in the 6701-MRQB record, if applicable, are:
- TPPCB
- AIB
- I/O AREA
- PST
- QTPDST
- QSAPWKAD
- QMBA
- PSTDCA
- REG14-12

When CPI Communications driven transaction programs issue the DL/I APSB call specifying a PSB that contains alternate PCBs, only the PGM keyword is applicable.

Equivalent IMS type-2 commands

The following table shows variations of the /TRACE PGM command and the IMS type-2 commands that perform similar functions.

Table 367. Type-2 equivalents for the /TRACE PGM command

Task	/TRACE PGM command	Similar IMS type-2 command
Starts the tracing of a program.	/TRACE SET ON PGM *pgmname*	UPDATE PGM NAME(*pgmname*) START(TRACE)
Stops the tracing of a program.	/TRACE SET OFF PGM *pgmname*	UPDATE PGM NAME(*pgmname*) STOP(TRACE)

Related concepts:

Diagnosing problems in the Queue Control Facility Message Requeuer (Diagnosis)

Related reference:

IMS Queue Control Facility overview

/TRACE PSB command

Use the /TRACE PSB command to record all full function IMS DL/I database calls issued for the named PSB. FP/DC/SAA calls are not captured when /TRACE SET ON PSB initiated tracing of PSBs.

Subsections:
- "Environment"
- "Syntax" on page 982
- "Keywords" on page 982
- "Usage notes" on page 982
- "Examples" on page 982

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 368. Valid environments for the /TRACE PSB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
COMP	X	X	
NOCOMP	X	X	
PSB	x	X	
SET	X	X	X

Syntax

```
►►──┬─/TRACE─┬──SET──┬──ON──┬──PSB──psbname──┬──NOCOMP──┬──────────────────────►◄
    └─/TRA───┘        └─OFF──┘                └──COMP────┘
```
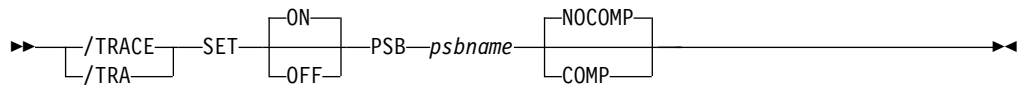
Keywords

The following keywords are valid for the /TRACE PSB command:

COMP

Used with the /TRACE SET PSB command to generate PCB and data-compare statement images.

The /TRACE SET PSB *psbname* COMP command only applies to BMPs in a DBCTL environment.

NOCOMP

Prevents PCB and data-compare statement images from being generated. NOCOMP is the default.

Usage notes

For LU 6.2, the PSB keyword is applicable only if the CPI Communications driven transaction program has issued a DL/I APSB call to allocate a PSB.

The information resulting from the use of this keyword is written on the X'5F' log record.

Examples

The following is an example of the /TRACE PSB command:

To trace all DL/I calls issued for PSB AALST:

Entry ET:
```
/TRACE SET ON PSB AALST COMP
```

Response ET:
```
DFS058I   TRACE COMMAND COMPLETED
```

/TRACE TABLE command

Use the /TRACE TABLE command with the SET keyword to start or stop online tracing into the specified trace tables.

Subsections:
- "Environment" on page 983
- "Syntax" on page 983
- "Keywords" on page 983
- "Usage notes" on page 986
- "Examples" on page 987

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 369. Valid environments for the /TRACE TABLE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
OPTION	X		X
SET	X	X	X
TABLE	X	X	X
VOLUME	X	X	X

Syntax
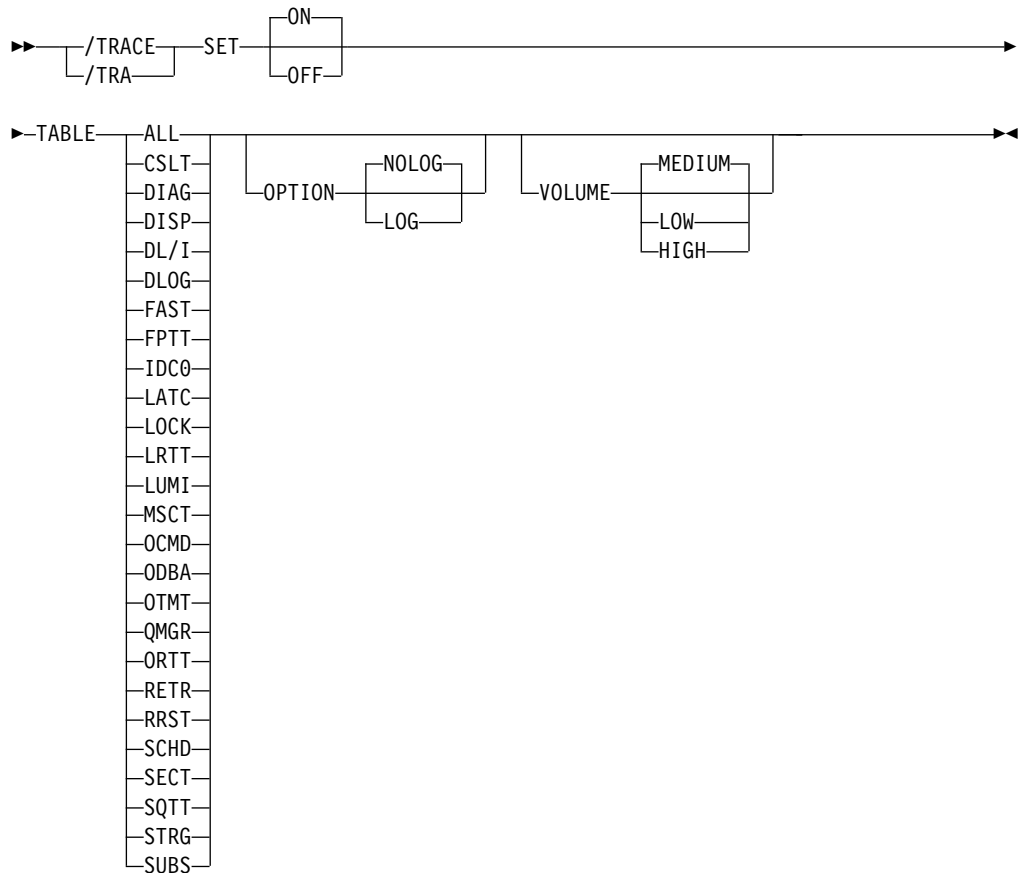
Keywords

The following keywords are valid for the /TRACE TABLE command:

OPTION
Indicates whether the trace tables are to be logged.

NOLOG
Indicates that the trace tables are not logged. This means that there will be no record of trace tables that have been reused. NOLOG is the default.

The same trace table is used for DL/I, LOCK, and PI trace information. Whether this table is logged depends on the logging option of the last /TRACE command that was entered for DL/I, PI, or LOCK.

LOG
Requests that the trace tables be written to the system log before they are reused (wrapped).

The log option includes the possibility of externally tracing to a data set other than the IMS OLDS. If specified, DASD external tracing has first priority, TAPE external tracing has second priority, and IMS OLDS has third priority. External tracing to the OLDS is not done without operator approval. This option is ignored for the retrieve trace and the Fast Path trace (default).

FPTRACE has its own SYSOUT. External trace is available to the alternate system only for DASD/TAPE type, but not for OLDS.

VOLUME
Specifies the volume of entries to be written to the specified trace table: LOW volume, MEDIUM volume (default), or HIGH volume.

The VOLUME keyword is valid on active and RSR tracking subsystems.

You can specify the trace that you want to activate or deactivate by using one of the following TABLE keyword parameters:

ALL
Indicates that traces into all trace tables are to be enabled or disabled. This is the default.

CSLT
Indicates that the CSL trace is to be activated or deactivated.

DIAG
Indicates that the /DIAGNOSE command trace tables are to be activated or deactivated.

DISP
Indicates that the dispatcher trace is to be activated or deactivated.

DL/I
Indicates that DL/I tracing is to be activated or deactivated.

DLOG
Indicates that the logging trace is to be activated or deactivated.

FAST
Indicates that the Fast Path trace is to be activated or deactivated.

Fast Path trace is activated by including the FPTRACE DD statement in the dependent region JCL to define the destination of the trace output and by issuing the operator command /TRACE SET ON TABLE FAST.

Recommendation: Run this trace only in a test environment because the FPTRACE output is large.

FPTT
Indicates that the Fast Path table trace is to be activated or deactivated.

IDC0

Indicates that tracing of errors in modules DFSCNXA0 and DFSIDC00 is to be activated or deactivated.

LATC

Indicates that the latch trace is to be activated or deactivated.

LOCK

Indicates that LOCK and PI tracing is to be activated or deactivated.

LRTT

Indicates that the log router trace table is to be activated or deactivated. LRTT is only valid for an RSR tracking subsystem.

LUMI

Indicates that the LU 6.2 manager trace is to be activated or deactivated. /TRACE SET ON TABLE LUMI must be entered first before entering any /TRACE SET ON LUNAME command in order to create the LUMI trace table for trace entries.

LUMI is not valid for an RSR tracking subsystem.

MSCT

Indicates that the MSC trace is to be activated or deactivated.

OCMD

Indicates that the OM command trace is to be activated or deactivated.

ODBA

Indicates that the ODBA trace table is to be activated or deactivated.

ODBA is not valid in an RSR tracker, batch, or DCCTL environment.

ORTT

Indicates that the Online Recovery System trace is to be activated or deactivated.

OTMT

Indicates that the IMS Open Transaction Manager Access (OTMA) trace is to be activated or deactivated.

QMGR

Indicates that the queue manager trace is to be activated or deactivated.

RETR

Indicates that the DL/I retrieve trace is to be activated or deactivated.

RETR is not valid for an RSR tracking subsystem.

RRST

Indicates that the Resource Recovery trace is activated or deactivated.

SCHD

Indicates that the scheduler trace is to be activated or deactivated.

SCHD is not valid for an RSR tracking subsystem.

SECT

Indicates that the security trace table is to be activated or deactivated.

SQTT

Indicates that the shared queues trace is to be activated or deactivated. SQTT is only valid in a shared-queues environment.

STRG

Indicates that the storage manager trace is to be activated or deactivated.

SUBS

Indicates that the external subsystem trace is to be activated or deactivated.

SUBS is not valid for an RSR tracking subsystem.

Usage notes

The TABLE keyword parameter indicates the specific trace that is to be activated or deactivated. The DL/I, LOCK, and PI traces share the same trace tables. However, turning on the DL/I trace does not turn on the LOCK trace, and vice versa.

The following trace into trace tables can be turned on or off with the online /TRACE command.

The following table shows the environments in which the trace tables are valid.

Table 370. Trace tables and environments in which they are valid

Trace table	DB/DC	DBCTL	DCCTL
ALL	X	X	X
CSLT	X	X	X
DIAG	X	X	X
DISP	X	X	X
DL/I	X	X	X
DLOG	X	X	X
FAST	X	X	X
FPTT	X	X	X
IDC0	X		X
LATC	X	X	X
LOCK	X	X	
LRTT	X	X	X
LUMI	X		X
MSCT	X		X
OCMD	X	X	X
ODBA	X	X	
ORTT	X		X
OTMT	X	X	
QMGR	X		X
RETR	X	X	
RRST	X		X
SCHD	X	X	X
SECT	X	X	X
SQTT	X		X
STRG	X	X	X
SUBS	X	X	X

Examples

The following are examples of the /TRACE TABLE command:

Example 1 for /TRACE TABLE command

To turn on online tracing into the DL/I trace table:

Entry ET:
```
/TRACE SET ON TABLE DL/I
```

Response ET:
```
DFS058I TRACE COMMAND COMPLETED
```

Example 2 for /TRACE TABLE command

To turn on the dispatcher's trace tables and have them written to the system log:

Entry ET:
```
/TRACE SET ON TABLE DISP OPTION LOG
```

When the dispatcher's trace tables are no longer required:

Entry ET:
```
/TRACE SET OFF TABLE DISP
```

Response ET:
```
DFS058I  TRACE COMMAND COMPLETED
```

Example 3 for /TRACE TABLE command

To turn on storage manager trace tables:

Entry ET:
```
/TRACE SET ON TABLE STRG
```

Response ET:
```
DFS058I TRACE COMMAND COMPLETED
```

When the storage manager trace tables are no longer needed:

Entry ET:
```
/TRACE SET OFF TABLE STRG
```

Response ET:
```
DFS058I TRACE COMMAND COMPLETED
```

Example 4 for /TRACE TABLE command

To turn on online tracing into the security trace table:

Entry ET:
```
/TRACE SET ON TABLE SECT
```

Response ET:

```
DFS058I TRACE COMMAND COMPLETED
```

/TRACE TCO command

Use the /TRACE TCO command to trace TCO (Time Controlled Operation) activity. For the first /TRACE command with the TCO keyword, the default module and level information is used.

TCO trace is basically a DC LINE or NODE trace, and the information developed is also written on the type X'67' log record.

Subsections:
- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 371. Valid environments for the /TRACE TCO command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
SET	X	X	X
TCO	X	X	X

Syntax

```
>>──┬─/TRACE─┬──SET──┬─ON──┬──TCO───────────────────────────><
    └─/TRA───┘       └─OFF─┘
```

/TRACE TIMEOUT command

Use the /TRACE TIMEOUT command to start or stop the I/O Timeout Detection facility.

Subsections:
- "Environment"
- "Syntax" on page 989
- "Keywords" on page 989
- "Usage notes" on page 989

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 372. Valid environments for the /TRACE TIMEOUT command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
AUTO	X		X
MSG	X		X
SET	X	X	X
TIMEOUT	X		X

Syntax

```
>>──┬─/TRACE─┬──SET──┬──ON──┬──TIMEOUT──┬──00────────────────┬──────><
    └─/TRA───┘       └─OFF──┘           └─time#──┬─────────┬─┘
                                                 ├──MSG────┤
                                                 └──AUTO───┘
```
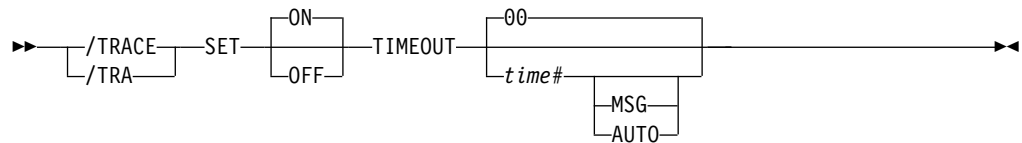
Keywords

The following keywords are valid for the /TRACE TIMEOUT command:

time#
> is the number of minutes used to determine if the I/O response is overdue. After this number of minutes, time has run out for the response. The range is from 1 through 60; the default value is 0.
>
> If *time#* is 0, or not specified, then when I/O is initiated for a node, the node will be placed on a queue, so that its status can be displayed with the /DISPLAY TIMEOVER command. You will not be notified if the node does not receive a response and the time elapses, and the node will not be reactivated.
>
> If *time#* is not 0, then the following keywords can be used.
>
> **MSG**
> > Indicates that a message is issued to the master terminal when I/O takes longer than *time#* minutes. The message indicates that the time has elapsed.
>
> **AUTO**
> > IMS issues a message to the master terminal, then perform a VTAM VARY NET,INACT and a VARY NET,ACT, if I/O takes longer than *time#* minutes. An /OPNDST is performed for operable devices that are not shared. For ISC nodes, a message is issued, but there is no automatic (AUTO) restart of any sessions and no VTAM VARY commands issued.

Usage notes

If you want to change the time period or the action to be taken if timeout occurs, you can enter the /TRACE SET ... TIMEOUT command while the Timeout Detection facility is already active. However, if nodes are receiving or sending input or output, they will function according to the previous settings of the /TRACE ... TIMEOUT command. If this is undesirable, then you should enter the /TRACE SET OFF TIMEOUT command before reentering /TRACE SET ... TIMEOUT.

If the timeout trace facility failed during IMS initialization, the /TRACE SET ... TIMEOUT command is rejected with an error message.

The VTAM TIMEOUT I/O facility is automatically started during IMS shutdown. It is set for 1 minute and AUTO.

/TRACE TMEMBER command

Use the /TRACE TMEMBER command to trace IMS Open Transaction Manager Access (OTMA) client activity for OTMA clients.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 991
- "Examples" on page 991

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 373. Valid environments for the /TRACE TMEMBER command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
SET	X	X	X
TMEMBER	X		X
TPIPE	x		X

Syntax
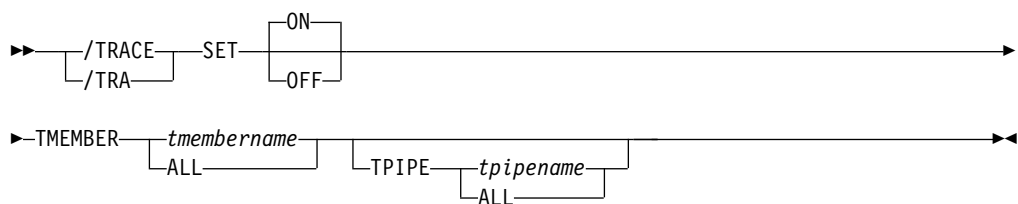
Keywords

The following keyword is valid for the /TRACE TMEMBER command:

TPIPE

Is used to trace transaction pipe activity for OTMA clients.

While processing the /TRACE TMEMBER TPIPE command, IMS creates a temporary transaction pipe (if one does not already exist) with the trace status. IMS sets the synchronization status for this transaction pipe when it sends or receives the first message for the transaction pipe.

If the member specified is a super member, trace status is updated for the super member's transaction pipe. If the member specified is a regular member whose hold queue output is managed by a super member, trace status is updated for both the regular member's transaction pipe and the super member's transaction pipe.

Usage notes

If the member specified is a super member, trace status is updated for the super member. If the member specified is a regular member whose hold queue output is managed by a super member, trace status is updated for both the regular member and the super member. Trace status is only updated on the IMS that processes the command. If the member specified is a regular member whose hold queue output is managed by a super member, and the trace status cannot be updated for both the regular member and the super member, the status is not updated for either member. The DFS058I COMMAND COMPLETED EXCEPT message is issued with the name of the regular member for which trace status could not be updated.

The TRACE SET ON/OFF TMEMBER *xxxx* command is issued to trace all the member activities, including transaction pipe activities. This command is the superset of the TRACE SET ON/OFF TMEMBER *xxxx* TPIPE ALL command.

Examples

The following is an example of the /TRACE TMEMBER command:

Entry ET:

```
/TRACE SET ON TMEMBER CLIENT1 TPIPE TPIPESY
```

Response ET:

```
DFS058I 15:45:05 TRACE COMMAND COMPLETED   SYS3
DFS996I *IMS READY*  SYS3
```

/TRACE TRAN command

Use the /TRACE TRAN command to trace the DL/I portion of Data Communications (DC) for a specific transaction.

Subsections:
- "Environment"
- "Syntax" on page 992
- "Usage notes" on page 992
- "Equivalent IMS type-2 commands" on page 992
- "Examples" on page 993

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 374. Valid environments for the /TRACE TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X

Table 374. Valid environments for the /TRACE TRAN command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
SET	X	X	X
TRAN	X		X

Syntax
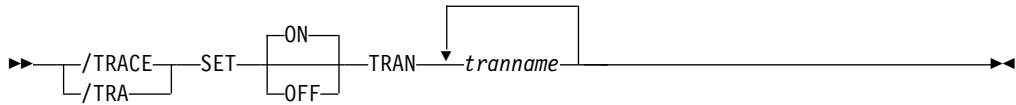
Usage notes

Each DL/I call to a TPPCB, issued by the user application program, is traced on entry to and exit from the DC call handler DFSDLA30. Upon entry to DFSDLA30, a type 6701-LA3A record is written; upon exit from DFSDLA30, a type 6701-LA3B record is written.

Additionally, in a shared-queues environment, a type 6701-ITQA record is written by module DFSITQS0 whenever a notification is received from CQS that there are messages available for the transaction.

Each of type 6701-LA3A and 6701-LA3B records will contain the following items, if applicable:
- TPPCB
- Up to 64 bytes of the I/O area
- Scheduler message block (SMB)
- Partition specification table (PST)

Each type 6701-ITQA record will contain the following items:
- IMSID of the IMS that wrote the record
- The AWE that was queued to DFSITQS0 to notify it that the transaction had messages to process
- SMB

Equivalent IMS type-2 commands

The following table shows variations of the /TRACE TRAN command and the IMS type-2 commands that perform similar functions.

Table 375. Type-2 equivalents for the /TRACE TRAN command

Task	/TRACE TRAN command	Similar IMS type-2 command
Starts the tracing of a transaction.	/TRACE SET ON TRAN *tranname*	UPDATE TRAN NAME(*tranname*) START(TRACE)
Stops the tracing of a transaction.	/TRACE SET OFF TRAN *tranname*	UPDATE TRAN NAME(*tranname*) STOP(TRACE)

Examples

The following is an example of the /TRACE TRAN command:

To log the PCB, I/O area, and PST whenever module DFSDLA30 is invoked to process transaction APPLE:

Entry ET:

```
/TRACE SET ON TRAN APPLE
```

Response ET:

```
DFS058I TRACE COMMAND COMPLETED
```

Related reference:

"UPDATE TRAN command" on page 1326

/TRACE TRAP command

Use the /TRACE TRAP command to detect overwrites of MFS blocks. When /TRACE is used, IMS attempts to detect overwrites in the MFS blocks.

If an overwrite occurs, IMS sends a warning message.

Subsections:
- "Environment"
- "Syntax"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 376. Valid environments for the /TRACE TRAP command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
SET	X	X	X
TRAP	X		X

Syntax

```
>>──┬─/TRACE─┬──SET──┬──ON──┬──TRAP──┬──1──┬──────────────────><
    └─/TRA───┘       └─OFF──┘        └──2──┘
```
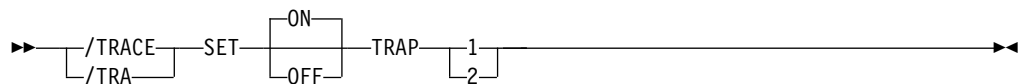
Related reference:

Trace records (Diagnosis)

/TRACE UNITYPE command

Use the /TRACE UNITYPE command to trace events that are related to the physical terminals of the specified type.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 995

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 377. Valid environments for the /TRACE UNITYPE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/TRACE	X	X	X
LEVEL	X		X
SET	X	X	X
UNITYPE	X		X

Syntax
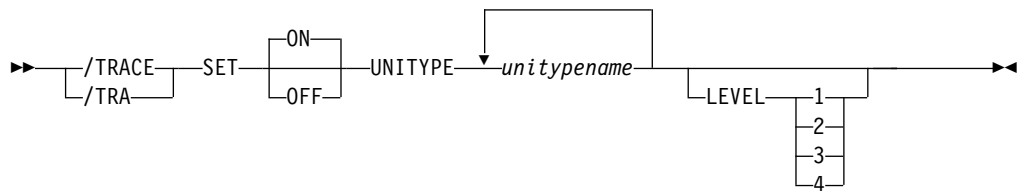
Keywords

The following keywords are valid for the /TRACE UNITYPE command:

LEVEL
> Expands the LINE, LINK, NODE, or UNITYPE trace functions. The LEVEL specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.
>
> LEVEL indicates the extent of the control block trace information desired. The indicated control blocks are only traced at relevant times. All levels are inclusive of numerically lower levels. The following list displays the levels and their associated blocks.

> **Level**　**Blocks**
>
> **1**　　　CLB (DECB) or LLB(MSC)
>
> 　　　　　CTB or LTB(MSC)
>
> 　　　　　IOB (for non-VTAM lines) or IOSB (MSC for channel-to-channel links)
>
> **2**　　　CNT or LNB(MSC)

CXB

CRB

CIB

CCB

PD stack

3   queue manager buffers

Input/output line buffers

LXB (for channel-to-channel links and processor storage to processor storage)

4   save area sets (IMS dispatching)

If the first /TRACE SET ON command does not specify LEVEL, a default of 4 is used. Specifying LEVEL on subsequent commands will change the defaults.

MODULE

Is used to expand the LINE, LINK, NODE, or UNITYPE trace functions. The MODULE specification is for the entire IMS system and is changed only by reissuing /TRACE with different values or by restarting the IMS control region.

MODULE indicates which modules are to have their control blocks traced.

ALL   Both device-dependent module (DDM) and MFS

DDM   Communication analyzer and device-dependent module interfaces

MFS   Communication analyzer and Message Format Service module interfaces

If the first /TRACE SET ON command does not specify MODULE, a default of ALL will be used. Specifying MODULE on subsequent commands will change the defaults.

Usage notes

The UNITYPE keyword is used to trace all terminals of a specific type. Parameters (*unitypename*) are similar to the identifiers displayed in the TYPE column by the /DISPLAY NODE and /DISPLAY LINE/PTERM commands. The following table shows the terminal types for UNITYPE parameters.

Table 378. UNITYPE parameters and terminal types

UNITYPE parameter	Terminal type
2260R	2260/2265 REMOTE
3286	3284/3286
2980	2980
3270R	3270 REMOTE
3270L	3270 LOCAL
RDR/PTR	LOCAL SYSIN/SYSOUT
FIN	3600
3277	3270 VTAM
SLU1	SLU TYPE 1
SLU2	SLU TYPE 2

Table 378. UNITYPE parameters and terminal types (continued)

UNITYPE parameter	Terminal type
SLUP	SLU TYPE P
LUT6	LU TYPE 6
NTO	NTO
CONSOLE	z/OS SYSTEM CONSOLE
TWX	TWX SWITCHED
3275SW	3270 SWITCHED
MSCMTM	MSC Memory to Memory communications
MSCCTC	MSC Channel to Channel communications
MSCTCPIP	MSC TCP/IP communications
MSCVTAM	MSC VTAM communications

If global resource information is kept in Resource Manager, /TRACE UNITYPE
sets a global trace status for all of the nodes of a specific type. This requires that
the inactive static nodes be processed on every IMS system. The UNITYPE
keyword is similar to specifying a generic parameter. If global resource information
is not kept in Resource Manager, /TRACE UNITYPE sets the trace status locally.

Chapter 29. /UNLOCK commands

/UNLOCK commands release resources that, in most cases, have been previously locked by the /LOCK command.

If the terminals are on a switched communication network and a physical or logical terminal disconnection occurs, an implied /UNLOCK is processed against the physical terminal and inquiry logical terminal.

When using ISC, the /UNLOCK command can only be used with logical terminals assigned to allocated users.

This command can be issued to an IMSplex using the Batch SPOC utility.

/UNLOCK SYSTEM is only valid if it is entered from the master terminal or from the system console on an XRF system.

A resource name can be defined with password protection in SAF for the DATABASE, LTERM, PROGRAM, and TRANSACTION keywords. If the parameter, LOCKSEC=Y (N is the default) is specified on the DFSPBxxx IMS.PROCLIB member, IMS calls the SAF and user exit. If the resource is not defined to SAF, or is defined and is authorized to the user, the command is processed. If the resource is defined to SAF but not authorized for use, the command is rejected with a DFS3689W message.

A period that is specified in the password brackets is not treated as a delimiter.

Subsections:
- "/UNLOCK DB command"
- "/UNLOCK LTERM command" on page 999
- "/UNLOCK NODE command" on page 1000
- "/UNLOCK PGM command" on page 1001
- "/UNLOCK PTERM command" on page 1002
- "/UNLOCK SYSTEM command" on page 1003
- "/UNLOCK TRAN command" on page 1005

/UNLOCK DB command

The /UNLOCK DB command specifies the database to be unlocked.

Subsections:
- "Environment" on page 998
- "Syntax" on page 998
- "Usage notes" on page 998
- "Equivalent IMS type-2 commands" on page 999
- "Examples" on page 999

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 379. Valid environments for the /UNLOCK DB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
DB	X	X	

Syntax

```
►►──┬─/UNLOCK─┬──DB──▼──dbname──────────────────────────────────►◄
    └─/UNL────┘          └─(password)─┘
```
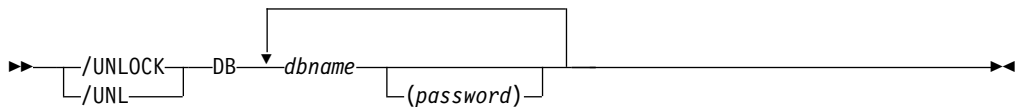
Usage notes

The output of the /UNLOCK DB command is changed when the command is entered through the OM API. In this case, the DFS058I message is not returned to OM. The command response returned to OM contains one or more of the following messages: DFS0488I, DFS3466I, DFS132

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB with the exception of the /START DB and the UPDATE DB START(ACCESS) command. A HALDB partition is not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

The HALDB partition reflects conditions such as STOPPED, LOCKED, or NOTOPEN. When a HALDB partition is stopped, it must be explicitly started again. Commands with the keyword ALL and commands against a HALDB master do not change the STOPPED and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is /DBR on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. Only the HALDB master displays STOPPED (each HALDB partition does not display STOPPED unless it was itself stopped). If a /DBR command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

Each partition inherits the access limitations of its HALDB master. If the /DBD command is issued against a HALDB master, all of its partitions close. A subsequent reference to any of the partitions results in the partition opening for input, although the partition's access might be UPDATE or EXCLUSIVE. The DBRC authorization state reflects the limited access.

This command can be issued by APPC and OTMA clients.

/UNLOCK DB is valid only if entered from the master terminal, the system console, a TCO script, or from an AOI application program.

While the database is being quiesced, this command cannot be processed successfully.

Equivalent IMS type-2 commands

The following table shows variations of the /UNLOCK DB command and the IMS type-2 commands that perform similar functions.

Table 380. Type-2 equivalents for the /UNLOCK DB command

Task	/UNLOCK DB command	Similar IMS type-2 command
Unlocks a database.	/UNLOCK DB *dbname*	UPDATE DB NAME(*dbname*) SET(LOCK(OFF))

Examples

The following is an example of the /UNLOCK DB command:

Entry ET:
```
/UNLOCK DB TREEFARM
```

Response ET:
```
DFS058I  UNLOCK COMMAND COMPLETED
```

Explanation: Database TREEFARM is unlocked and can be used.

Related reference:

"UPDATE DB command" on page 1038

/UNLOCK LTERM command

The /UNLOCK LTERM command specifies the logical terminal to be unlocked. This keyword applies only to the entering physical terminal and to logical terminals assigned to that physical terminal.

Subsections:
- "Environment"
- "Syntax" on page 1000
- "Usage notes" on page 1000

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 381. Valid environments for the /UNLOCK LTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
LTERM	X		X

Syntax

```
>>--+--/UNLOCK--+--LTERM--+--+-<--ltermname--+-------------+--+----------->-<
    +--/UNL----+          |  +--(password)--+  |
                          +--ALL---------------------------------+
```
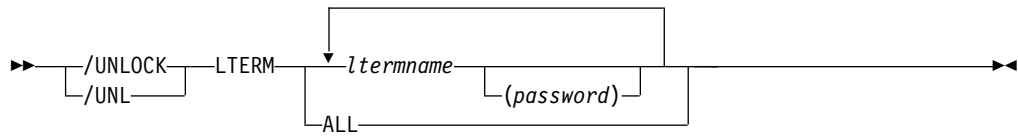
Usage notes

The /UNLOCK LTERM ALL command can only be used when all of the logical terminals associated with the entering physical terminal do not have passwords.

/UNLOCK LTERM applies only to the entering physical terminal.

/UNLOCK LTERM is not allowed from the OM API.

/UNLOCK NODE command

The /UNLOCK NODE command specifies the VTAM node to be unlocked. This keyword applies only to the entering physical terminal and to logical terminals assigned to that physical terminal.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 382. Valid environments for the /UNLOCK NODE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
NODE	X		X

Syntax

```
>>--+--/UNLOCK--+--NODE---------------------------------------------------->-<
    +--/UNL----+
```
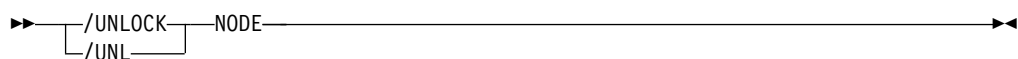
Usage notes

/UNLOCK NODE applies only to the entering physical terminal.

/UNLOCK NODE is not allowed from the OM API.

/UNLOCK PGM command

The /UNLOCK PGM command specifies the application program to be unlocked.

This command can be issued by APPC and OTMA clients.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands"
- "Examples"

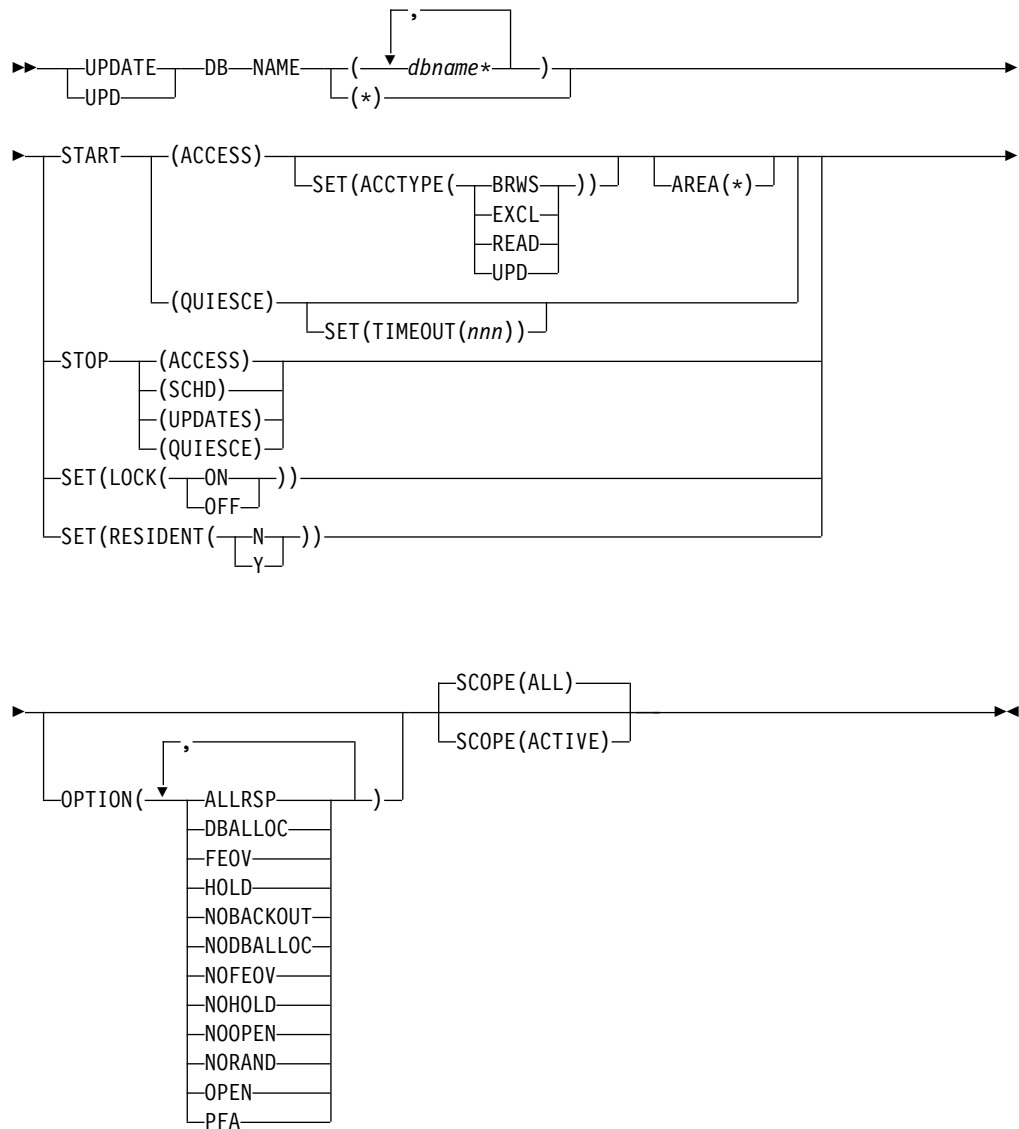## Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 383. Valid environments for the /UNLOCK PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
PGM	X	X	X

Syntax

```
►►──┬─/UNLOCK─┬──PGM──┬─►─pgmname─┬────────────────►◄
    └─/UNL────┘       └─(password)─┘
```
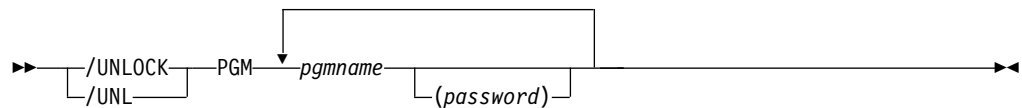
Usage notes

The /UNLOCK PGM command is valid only if entered from the master terminal, the system console, a TCO script, or from an AOI application program.

Equivalent IMS type-2 commands

The following table shows variations of the /UNLOCK PGM command and the IMS type-2 commands that perform similar functions.

Table 384. Type-2 equivalents for the /UNLOCK PGM command

Task	/UNLOCK PGM command	Similar IMS type-2 command
Unlocks a program.	/UNLOCK PGM *pgmname*	UPDATE PGM NAME(*pgmname*) SET(LOCK(OFF))

Examples

The following is an example of the /UNLOCK PGM command:

Entry ET:
```
/UNLOCK PGM APPLETRE
```

Response ET:
```
DFS058I  UNLOCK COMMAND COMPLETED
```

Explanation: Application program APPLETRE is unlocked and can be executed.

Related reference:

"UPDATE PGM command" on page 1259

/UNLOCK PTERM command

The /UNLOCK PTERM command specifies the physical terminal to be unlocked. This keyword applies only to the entering physical terminal and to logical terminals assigned to that physical terminal.

This command can be issued by APPC and OTMA clients.

Subsections:

- "Environment"
- "Syntax"
- "Usage notes"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 385. Valid environments for the /UNLOCK PTERM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
PTERM	X		X

Syntax

```
►►──┬─/UNLOCK─┬──PTERM───────────────────────────────────────────►◄
    └─/UNL────┘
```

Usage notes

/UNLOCK PTERM applies only to the entering physical terminal.

/UNLOCK PTERM is not allowed from the OM API.

Examples

The following is an example of the /UNLOCK PTERM command:

Entry ET:
```
/UNLOCK PTERM
```

Response ET:
```
DFS058I  UNLOCK COMMAND COMPLETED
```

Explanation: The physical terminal from which the command is entered is unlocked.

/UNLOCK SYSTEM command

The /UNLOCK SYSTEM command notifies a newly created active system in an XRF complex that I/O prevention is complete.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Examples"

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 386. Valid environments for the /UNLOCK SYSTEM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
SYSTEM	X		X

Syntax

```
►►──┬─/UNLOCK─┬──SYSTEM──────────────────────────────────────────────►◄
    └─/UNL────┘
```

Usage notes

I/O prevention is initiated at takeover to ensure that the failing active system cannot write to the databases. The alternate system then initiates I/O toleration to ensure database integrity and to enable new transaction processing as soon as possible. /UNLOCK SYSTEM ends the I/O toleration phase of processing.

/UNLOCK SYSTEM is valid only if it is entered from the master terminal or from the system console on an XRF system.

Examples

The following is an example of the /UNLOCK SYSTEM command:

The following three figures illustrate the use of the /UNLOCK SYSTEM command on a newly created active system. Each figure is a formatted master screen for the newly created active system IMSB.

The following figure shows a screen of a newly created active system in the I/O toleration phase of processing (awaiting I/O prevention). Database DD41M803 has an I/O toleration EEQE.

```
02/05/15 16:19:03  RSENAME: DFSRSENM  ACTIVE  AWAITING I/O PREVENTION  IMSB
 DFS994I *CHKPT 85135/152931**SIMPLE**
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:   11
DFS3804I LAST CHKPT ID VALID FOR RESTART: 85135/161847-BUILDQ: 85135/161213
DFS994I TAKEOVER COMPLETED.
DFS3859I 16:18:29 PRIORITY 4 SESSIONS SWITCHED.
DFS3860I 16:18:29 ALL TERMINAL SESSIONS SWITCHED.
----------------------------------------------------------------------------

    DATABASE
    DD41M803
        ERROR DD  TYPE    BLOCK
        DD41M803  IOT     0000003F
    *85135/161902*
                                              PASSWORD:
/dis db dd41m803 bkerr
```

Figure 3. I/O toleration phase of processing

The following figure shows a screen of the use of the /UNLOCK SYSTEM
command to notify the newly created active system that I/O prevention is
complete (the XRF system status line now indicates that processing is no longer
degraded by I/O toleration).

```
02/05/15 16:34:14  RSENAME: DFSRSENM  ACTIVE                         IMSB
 DFS994I *CHKPT 85135/161847**SIMPLE**
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:   11
DFS3804I LAST CHKPT ID VALID FOR RESTART: 85135/161847-BUILDQ: 85135/161213
 DFS058 16:34:14 UNLOCK COMMAND IN PROGRESS
DFS0488I - UNLOCK COMMAND COMPLETED.  RC = 00
DFS3860I 15:29:19 ALL TERMINAL SESSIONS SWITCHED.
----------------------------------------------------------------------------


                                              PASSWORD:
/unlock system
```

Figure 4. /UNLOCK SYSTEM command

The following figure is a screen that shows that the I/O toleration EEQE for
database DD41M803 has been deleted as part of /UNLOCK SYSTEM processing.

```
02/05/15 16:35:00  RSENAME: DFSRSENM  ACTIVE                         IMSB
 DFS994I *CHKPT 85135/161847**SIMPLE**
DFS3499I ACTIVE DDNAMES: MODBLKSA IMSACBA  FORMATA  MODSTAT ID:   11
DFS3804I LAST CHKPT ID VALID FOR RESTART: 85135/161847  -  BUILDQ: 85135/161213
DFS058 16:34:14 UNLOCK COMMAND IN PROGRESS
DFS0488I - UNLOCK COMMAND COMPLETED.  RC = 00
DFS3860I 16:18:29 ALL TERMINAL SESSIONS SWITCHED.
----------------------------------------------------------------------------

    DATABASE
    DD41M803
        NO EEQE OR INCOMPLETE BACKOUT INFORMATION AVAILABLE
    *85135/163500*
                                              PASSWORD:
/dis db dd41m803 bkerr
```

Figure 5. EEQE deleted as part of /UNLOCK SYSTEM processing

Related concepts:

➥ Takeover phase of the XRF process (System Administration)

/UNLOCK TRAN command

The /UNLOCK TRAN command specifies the transaction code to be unlocked.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Equivalent IMS type-2 commands"
- "Examples" on page 1006

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 387. Valid environments for the /UNLOCK TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
/UNLOCK	X	X	X
TRAN	X		X

Syntax
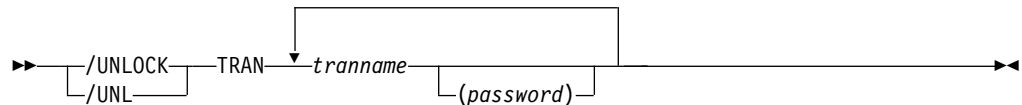
Usage notes

The /UNLOCK TRAN command cannot be used for Fast Path exclusive or CPI Communications driven transaction programs.

This command can be issued by APPC and OTMA clients.

The /UNLOCK TRAN command is valid only if entered from the master terminal, the system console, a TCO script, or from an AOI application program.

Equivalent IMS type-2 commands

The following table shows variations of the /UNLOCK TRAN command and the IMS type-2 commands that perform similar functions.

Table 388. Type-2 equivalents for the /UNLOCK TRAN command

Task	/UNLOCK TRAN command	Similar IMS type-2 command
Unlocks a transaction.	/UNLOCK TRAN *tranname*	UPDATE TRAN NAME(*tranname*) SET(LOCK(OFF))

Examples

The following is an example of the /UNLOCK TRAN command:

Entry ET:
```
/UNLOCK TRAN SEED
```

Response ET:
```
DFS058I  UNLOCK COMMAND COMPLETED
```

Explanation: Transaction SEED is unlocked and can be scheduled.
Related reference:
"UPDATE TRAN command" on page 1326

Chapter 30. UPDATE commands

Use the IMS UPDATE commands to update IMS resources or resource descriptors.

These commands can be issued through TSO SPOC, IBM Management Console for IMS and DB2 for z/OS, the Manage Resources options in the IMS Applications menu, or other user-written programs that issue commands through OM API. These commands can also be issued to an IMSplex by using the Batch SPOC utility.

UPDATE commands are:
- "UPDATE AREA command"
- "UPDATE DATAGRP command" on page 1021
- "UPDATE DB command" on page 1038
- "UPDATE DBDESC command" on page 1067
- "UPDATE IMS command" on page 1073
- "UPDATE IMSCON commands" on page 1085
- "UPDATE LE command" on page 1194
- "UPDATE MSLINK command" on page 1199
- "UPDATE MSNAME command" on page 1211
- "UPDATE MSPLINK command" on page 1218
- "UPDATE ODBM commands" on page 1228
- "UPDATE OLREORG command" on page 1244
- "UPDATE OTMADESC command" on page 1250
- "UPDATE PGM command" on page 1259
- "UPDATE PGMDESC command" on page 1276
- "UPDATE POOL command" on page 1286
- "UPDATE RM command" on page 1298
- "UPDATE RTC command" on page 1308
- "UPDATE RTCDESC command" on page 1315
- "UPDATE TRAN command" on page 1326
- "UPDATE TRANDESC command" on page 1360

UPDATE AREA command

Use the UPDATE AREA command to change the status of area resources.

When the UPDATE AREA command is issued, the command is processed only by the IMS to which it is routed. The command does not preload or preopen areas on other IMS systems in the IMSplex that share the area. The UPDATE AREA command is routed by OM. OM routes the command to all active DB/DC or DBCTL IMS systems, unless specific routing is specified. OM selects one IMS as the command master.

Subsections:
- "Environment" on page 1008
- "Syntax" on page 1008
- "Keywords" on page 1009

- "Usage notes" on page 1013
- "Equivalent IMS type-1 commands" on page 1013
- "Output fields" on page 1013
- "Return, reason, and completion codes" on page 1014
- "Examples" on page 1018

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE AREA command and keywords can be issued.

Table 389. Valid environments for the UPDATE AREA command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE AREA	X	X	
NAME	X	X	
OPTION	X	X	
SCOPE	X	X	
START	X	X	
STOP	X	X	

Syntax

```
                                            ┌─SCOPE(ALL)────┐
   ►─┬──────────────────────────────────┬───┼───────────────┼──►◄
     │              ┌─,────────┐         │   └─SCOPE(ACTIVE)─┘
     │              ▼          │         │
     └─OPTION(──┬───┬─ALLRSP─┬─┴─)──────┘
                    ├─FEOV──┤
                    ├─HOLD──┤
                    ├─NOFEOV┤
                    ├─NOHOLD┤
                    ├─OPEN──┤
                    └─PFA───┘
```
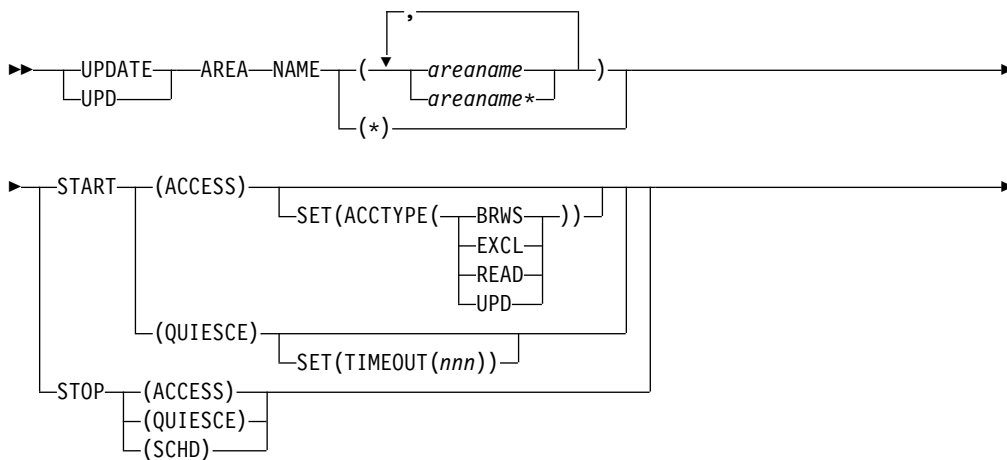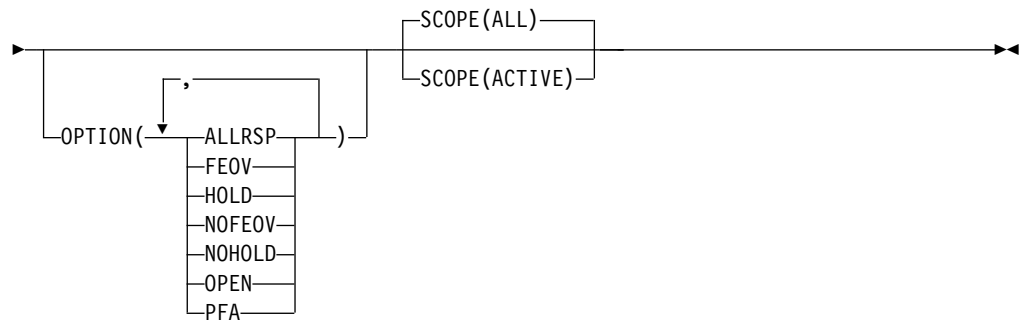
Keywords

The following keywords are valid for the UPDATE AREA command:

NAME()

Specifies the names of the specific areas that are to be processed or a group of areas to be processed.

Wildcard parameters can be specified. The area names that match the generic or wildcard parameter are processed. For specific or wildcard names, response lines are returned for all the area names that are processed.

NAME(*) indicates that the command is to be applied to all the areas in the system. When the NAME(*) is specified, the response lines are returned for only the area names that resulted in an error. If OPTION(ALLRSP) is specified with NAME(*), response lines are returned for all the area names that are processed.

OPTION()

Specifies the additional functions to be performed. Following is a list of additional functions:

ALLRSP

Indicates that the response lines are to be returned for all areas that are processed on the command. ALLRSP is only valid with NAME(*). The default action is to return response lines only for the areas that resulted in an error.

FEOV Indicates to force end of volume after the command processing is complete. The IMS log switches to the next OLDS, and a simple checkpoint is taken unless specified with STOP(QUIESCE). This switch is marked as a recovery point for log archiving purposes. A simple checkpoint is not taken when specified with STOP(QUIESCE).

FEOV is valid only with START(QUIESCE) OPTION(NOHOLD), STOP(ACCESS), or STOP(QUIESCE), and is not valid on an RSR tracker.

OPTION(FEOV) is the default when specified with START(QUIESCE) OPTION(NOHOLD) or STOP(QUIESCE).

For quiesce processing, the end of volume will be forced after the quiesce point has been reached, causing the logs to be switched before the quiesce is released, so that any new updates occur on the new IMS log. This occurs during the stop quiesce processing of

a START(QUIESCE) OPTION(NOHOLD) or when a STOP(QUIESCE) is issued to release a quiesce from a previous START(QUIESCE) OPTION(HOLD).

HOLD

Specifies that after the quiesce has been achieved successfully the area should remain quiesced. A subsequent STOP(QUIESCE) would be required to release the quiesce on the DEDB area. HOLD is valid only with the START(QUIESCE) keyword. This keyword is mutually exclusive with the NOHOLD keyword.

NOFEOV

Indicates to not force end of volume after the command processing is complete. The IMS log does not switch to the next OLDS and a simple checkpoint is not taken.

NOFEOV is valid only with START(QUIESCE) OPTION(NOHOLD), STOP(ACCESS), or STOP(QUIESCE). OPTION(NOFEOV) is the default when specified with STOP(ACCESS).

NOHOLD

Specifies that after the quiesce has been achieved successfully the area should automatically release the quiesce. If neither HOLD nor NOHOLD is specified, NOHOLD is assumed. NOHOLD is valid only with the START(QUIESCE) keyword. This keyword is mutually exclusive with the HOLD keyword.

OPEN Specifies to open the DEDB area data sets that are specified on the NAME() parameter, even if the area is not registered on DBRC as PREOPEN. The OPTION(OPEN) is processed locally by all IMS systems that receive the command and is not maintained as a global status in RM.

PFA Sets or resets the prevent further authorization (PFA) status in the RECON data set for the DEDB area. Use the PFA option with UPDATE AREA START(ACCESS) to enable access to a DEDB area. Use the PFA option with UPDATE AREA STOP (ACCESS|SCHD) to prevent access to a DEDB area. If you specify OPTION(PFA), the command master IMS updates the RECON only if the command is successful at the master IMS.

SCOPE()

Specifies where IMS should apply the change. The default is ALL.

ALL

Changes are applied to the active IMS systems to which the command is routed. If you specify that global area status is to be maintained, changes are also applied globally by updating the value maintained by RM. The RM status is updated only by the command master IMS. If global area status is not maintained, the command action is same as the SCOPE(ACTIVE) command.

This option does not apply to the quiesce function. The scope of a quiesce is always all instances of the area usage across the IMSplex. There is not a global status of QUIESCE for a DEDB area. Using SCOPE(ALL) for a START(QUIESCE) or STOP(QUIESCE) does not change the global status for the area.

ACTIVE

Changes are applied to the active IMS systems to which the command

is routed to. Any global status information maintained in the RM resource structure is not changed by the SCOPE(ACTIVE) command.

This option does not apply to the quiesce function. The scope of a quiesce is always all instances of the area usage across the IMSplex. There is no quiesce which would only apply to a subset of the IMSplex.

The UPDATE AREA command keywords that update information locally in the active IMS system and globally in RM when SCOPE(ALL) is specified include:
- START(ACCESS)
- STOP(SCHD)
- STOP(ACCESS)

SET() Specifies the attribute values to be changed.

ACCTYPE()

Specifies the access intent for the named area. This keyword can be specified only if START(ACCESS) is specified. This keyword overrides the database access intent of its DEDB.

You can specify one of the following keywords:

BRWS

Specifies that the named area is available for read-only processing on this IMS subsystem. The only programs that can use the area on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data by using the GO processing option might access uncommitted data because a sharing IMS subsystem could be updating the area. The area is opened for read-only processing.

EXCL

Specifies that the named area is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the area is registered to DBRC.

READ

Specifies that the named area is available for read processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the area. ACCTYPE of READ differs from ACCTYPE of BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The area is opened for read.

UPD

Specifies that the named area is available for update as well as read processing in the IMS subsystem.

TIMEOUT(*nnn*)

Specifies the number of seconds to wait before a timeout occurs in a DEDB area quiesce. The timeout value can be 1 - 999 seconds. The TIMEOUT parameter value can override the DBQUIESCETO parameter in the DFSCGxxx member of the IMS PROCLIB data set. If the TIMEOUT parameter is omitted and the DBQUIESCETO parameter is not specified, the default timeout value is 30 seconds. The TIMEOUT keyword is valid only with the START(QUIESCE) keyword.

START()

Specifies the attributes that are to be started.

ACCESS

Specifies the specific areas of a DEDB to be allocated or reallocated.

An UPDATE AREA START(ACCESS) can be issued on an RSR tracker to resume tracking for those areas that were stopped or had tracking errors. In addition, Online Forward Recovery (OFR) is started for areas that are not current with mainline tracking.

For Virtual Storage Option (VSO) areas, the UPDATE AREA START(ACCESS) can be used to preopen areas that are defined with the PREOPEN option. For VSO areas defined with the PRELOAD option, the command causes the areas to be loaded into the z/OS data space or an XES structure depending on the share level of the area. The command has no effect on VSO areas that are in virtual storage.

QUIESCE

Specifies that the DEDB areas named on the command are to be quiesced to establish a new recovery point. The scope of a quiesce is always all instances of the area usage across the IMSplex. There is no quiesce that would apply only to a subset of the IMSplex.

STOP()

Specifies the attributes that are to be stopped.

ACCESS

Stops the access and updating of the specified DEDB areas and closes them.

The UPDATE AREA STOP(ACCESS) command for VSO areas removes the areas from the data space or XES structure and forces updates to be written back to DASD.

An UPDATE AREA START(ACCESS) command is required to open and reallocate the areas closed by the UPDATE AREA STOP(ACCESS) command.

QUIESCE

Specifies that the DEDB areas named on the command should be made available again by releasing the quiesce on the areas.

Unlike START(QUIESCE), where each area resource listed must be quiesced in order for the command to complete successfully, STOP(QUIESCE) continues to process each listed resource even if some resources are not in quiesced state or cannot be released from quiesced state. For those resources that are not in quiesced state or cannot be released from quiesced state, the command returns a response line for each of those resources.

SCHD Specifies that the data sets associated to the areas are to be closed and deallocated.

Note: An UPDATE AREA START(ACCESS) is required to reallocate the areas stopped by the UPDATE AREA STOP(SCHD) command.

If UPDATE AREA STOP(SCHD) is processed during HSSP processing, the area will be released after the current commit

processing completes. Any image copy option in process when the command is issued can affect the continued system operation.

For VSO DEDB areas that are in a z/OS data space or XES structure, the UPDATE AREA STOP(SCHD) command action results in the removal of the VSO areas from the data space or XES structure and the writing of updates to DASD.

Usage notes

The UPDATE AREA command can only be specified using the OM API. The command is also not allowed on the XRF alternate.

The UPDATE AREA STOP(SCHD) command and the UPDATE AREA START(ACCESS) SET(ACCTYPE()) command are not allowed on any RSR tracker. However, the commands UPDATE AREA START(ACCESS) or UPDATE AREA STOP(ACCESS) are allowed only on a database level RSR tracker.

The UPDATE AREA START(ACCESS) and UPDATE AREA STOP(SCHD) commands are recoverable, and a X'22' log record is written, but the UPDATE AREA STOP(ACCESS) command is not recoverable.

A new log record, X'594C', is written for every DEDB area to which the UPDATE AREA command applies. It also includes global status and the global command time stamp.

The UPDATE AREA command is enhanced to return CCTXT with a non-zero completion code. The CCTXT can be up to 32 bytes, and it includes information about what the completion code means. The UPDATE AREA SCOPE(ALL) command returns a response line with the completion code for the global status update. The CCTXT for that code is GBL CC.

While the database is being quiesced, this command with keywords other than STOP(QUIESCE) cannot be processed successfully.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE AREA command and the type-1 IMS commands that perform similar functions.

Table 390. Type-1 equivalents for the UPDATE AREA command

UPDATE AREA command	Similar IMS type-1 commands
`UPDATE AREA NAME(name) STOP(ACCESS)`	`/DBR AREA areaname`
`UPDATE AREA NAME(name) START(ACCESS)`	`/START AREA areaname`
`UPDATE AREA NAME(name) STOP(SCHD)`	`/STOP AREA areaname`

Output fields

The following table shows the UPDATE AREA output fields. The columns in the table are as follows:

Short label
    Contains the short label generated in the XML output.

Keyword
    Identifies keyword on the command that caused the field to be generated.

N/A appears for output fields that are always returned. ERR appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 391. Output fields for the UPDATE AREA command

Short label	Keyword	Meaning
AREA	N/A	Area name. The area name is always returned.
CCTXT	*error*	The completion code text that briefly explains the meaning of the completion code.
CC	N/A	Completion code for the line of output. Completion code is always returned.
GBL	SCOPE(ALL)	Indicates that the response line is for the global update.
EERT	ERR	Error text returned to add more meaning to the completion code and may include a return code from a service. The error text may be returned for a non-zero completion code.
MBR	N/A	The IMSplex member that built the output line. The IMS identifier of the IMS for which the area information is displayed. The IMS identifier is always returned.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the UPDATE AREA command. The OM return and reason codes that may be returned as a result of the UPDATE AREA command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 392. Return and reason codes for the UPDATE AREA command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE AREA command completed successfully.
X'00000008'	X'00002014'	The UPDATE AREA command is not processed because an invalid character is found in the area name parameter.
X'00000008'	X'00002040'	Either more than one keyword or an invalid combination of filters was specified on the UPDATE AREA command. For example, the following keyword combinations are incorrect: • START(ACCESS) and STOP(ACCESS) • STOP(ACCESS) and OPTION(OPEN) • STOP(SCHD) and OPTION(OPEN) • OPTION(NOFEOV,FEOV) Check the input command and reenter the correct combinations.

Table 392. Return and reason codes for the UPDATE AREA command (continued)

Return code	Reason code	Meaning
X'00000008'	X'00005000'	The UPD AREA command processing terminated because IMODULE GETSTOR storage could not be obtained.
X'0000000C'	X'00003000'	The UPDATE AREA command is successful for at least one resource name. The UPDATE AREA command was not successful for one or more resource names. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the UPDATE AREA command are listed in Table 393 on page 1016.
X'0000000C'	X'00003004'	The UPDATE AREA command was not successful for all the resource name(s) specified. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the UPDATE AREA command are listed in Table 393 on page 1016.
X'00000010'	X'0000400C'	The UPDATE AREA command is not processed because the command variation entered is not valid on the XRF alternate.
X'00000010'	X'00004014'	The UPDATE AREA command is not processed because the function is not valid on the RSR tracker.
X'00000010'	X'00004024'	The UPDATE AREA command is not processed because Fast Path is not installed.
X'00000010'	X'00004025'	The UPDATE AREA command is rejected because no Fast Path areas are defined.
X'00000010'	X'00004200'	Commands are not processed because IMS shutdown is in progress.
X'00000010'	X'00004208'	Invalid command for RMENV=NO.
X'00000010'	X'00004400'	MINVERS in the RECON data sets is not 11.1.
X'00000014'	X'00005001'	The UPDATE AREA command processing terminated because WKAP or MAIN pool storage could not be obtained.
X'00000014'	X'00005004'	The UPDATE AREA command processing terminated because a DFSCOMD response buffer could not be obtained.
X'00000014'	X'00005008'	The UPDATE AREA command processing terminated because a DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	The UPDATE AREA command processing terminated because AWE could not be obtained.
X'00000014'	X'00005FFF'	The UPDATE AREA command processing terminated because of an internal error.
X'02000008'	X'0000203C'	An invalid TIMEOUT value is specified. The value must be a numeric value between 1 and 999.

The following table includes an explanation of the completion codes. Errors unique to the processing of UPDATE AREA command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 393. Completion codes for the UPDATE AREA command

Completion code	Completion code text	Meaning
0		The command completed successfully for the resource.
0C	COMMAND COMPLETE FOR NONE	This error is returned when all the areas could not be quiesced.
10	NO RESOURCE FOUND	No resource found.
11	DUPLICATE RESOURCE NAME	The resource name is specified multiple times on the command and is ignored.
17	ANOTHER CMD IN PROGRESS	This error is returned when the quiesce could not be started because another database command was in progress.
53	NO RM ADDRESS SPACE	This error is returned when the quiesce could not be started because the RM address space is not present.
55	NO FAST PATH INSTALLED	The command failed because Fast Path is not installed.
81	DBRC ERROR	This error is returned when an unexpected DBRC error occurs during the quiesce command.
90	INTERNAL ERROR	The command entered is not processed because of an internal error.
91	TIMEOUT ERROR	This error is returned when the quiesce could not be completed within the timeout period. The quiesce might be processing longer than the timeout value, or an IMSplex component involved in the process might have failed or hung. Check the state of the IMSplex components by, for example, issuing a QUERY IMSPLEX command or checking the z/OS system log.
92	COMMAND PROCESSING ERROR	The command entered is not processed because of a command error. A unique completion code could not be generated to explain the error. The message number and the return code that could not be converted to a completion code are listed in the error text.
A5	PREVENT FURTHER AUTH ON	The command entered is not processed, because the database or area is defined to DBRC as 'prevent further authorization'.
A9	DB OR AREA AUTHORIZATION ERROR	Area authorization to DBRC failed.
D1	DATABASE WRITE ERROR	This error is returned when the quiesce could not be completed because the database has a write error.
D2	DATABASE NEEDS BACKOUT	This error is returned when the quiesce could not be completed because the database needs backout.
D3	DATABASE OR AREA NEEDS RECOVERY	This error is returned when the quiesce could not be completed because the database needs recovery.

Table 393. Completion codes for the UPDATE AREA command (continued)

Completion code	Completion code text	Meaning
D9	COMMAND PROCESSING ERROR	The UPDATE AREA START(ACCESS) command could not be processed for the area name because the area open failed.
E0	DATABASE OR AREA IN RECOVERY	The command is not processed because the database or area is in recovery.
E5	PARTICIPANTS UNABLE TO QUIESCE	This error is returned when the quiesce could not be completed successfully across the IMSplex. The IMS with this completion code was the quiesce participant that was not able to be quiesced.
E6	QUIESCE COMMUNICATION FAILURE	This error is returned when the quiesce could not be completed because of a failure to communicate across the IMSplex. There could be a problem with RM, OM, or SCI that has caused the communication failure to occur.
E7	CMD NOT ALLOWED	This error is returned when the database command could not be processed because a quiesce command was in progress.
E8	DATABASE HAS INTENT TO REORGANIZE	This error is returned when the quiesce could not be started because the RECON data sets indicate that there is an intent to reorganize the database.
E9	DB IN WRONG STATE TO BE QUIESCED	This error is returned when the named resource is in the wrong state for quiesce processing to proceed.
EF	DATABASE IS IN ERROR	This error is returned when the quiesce could not be completed because the database is in error.
F0	NO AREA LOCK	The command processing failed because the area lock could not be obtained.
F1	AREA NOT STOPPED	The command entered is not processed because the area is not stopped.
F2	PRELOAD IS ACTIVE FOR AREA	The command is not processed because preload is active for the area.
F3	UNRESOLVED INDOUBTS FOR AREA	The command entered is not processed because unresolved indoubts exist for the area.
F4	ALLOCATION FAILED	Allocation failed for the area name.
F5	AREA NEEDS RECOVERY	The command processing failed because the area needs recovery.
F6	ADS NUMBER DISCREPANCY	The command failed for the AREA because there is a discrepancy between the number of ADS allocated by IMS and the number of ADS known to DBRC. Correct the discrepancy and reissue the command to deallocate the data sets.
F7	AREA IS NOT LOADED INTO CF	The area open failed and is not loaded into the Coupling Facility.
F8	AREA HAS I/O TOLERATED CI	The command is not processed because the area has an I/O tolerated CI.
F9	AREA HAS SECOND CI EEQE	The command is not processed because the area has a second CI EEQE.

Table 393. Completion codes for the UPDATE AREA command (continued)

Completion code	Completion code text	Meaning
FC	UTILITY ACTIVE ON AREA	This error is returned when the area is in use by a utility.
FD	AREA HAS EEQE	This error is returned when the area has an extended error queue element (EEQE).
FE	AREA HAS EQE	This error is returned when the area has an error queue element (EQE).
190	DB ACCESS LESS THAN AREA ACCESS.	The command fails for the resource, because the DEDB access is less than the area access.
191	AREA ACCESS ALREADY AT LEVEL.	The command fails for the resource, because the area access is already at the area access level.
192	AREA IN USE-UTIL.	The command fails for the resource, because a utility is running against the area.

Examples

The following are examples of the UPDATE AREA command:

Example 1 for UPDATE AREA command

TSO SPOC input:

```
UPD AREA NAME(DB21AR1*) STOP(ACCESS)
```

TSO SPOC output:

```
AreaName MbrName    CC
DB21AR1  IMS2        0
DB21AR1  SYS3        0
DB21AR10 IMS2        0
DB21AR10 SYS3        0
DB21AR11 IMS2        0
DB21AR11 SYS3        0
```

OM API input:

```
CMD(UPD AREA NAME(DB21AR1*) STOP(ACCESS))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2003.132 16:17:33.260435</statime>
<stotime>2003.132 16:17:33.281165</stotime>
<staseq>B968A333F5A93283</staseq>
<stoseq>B968A333FAB8DC83</stoseq>
<rqsttkn1>USRT005 10091733</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>AREA            </kwd>
<input>UPD AREA NAME(DB21AR1*) STOP(ACCESS) </input>
```

```
</cmd>
<cmdrsphdr>
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>AREA(DB21AR1 ) MBR(IMS2    ) CC(   0) </rsp>
<rsp>AREA(DB21AR10) MBR(IMS2    ) CC(   0) </rsp>
<rsp>AREA(DB21AR11) MBR(IMS2    ) CC(   0) </rsp>
<rsp>AREA(DB21AR1 ) MBR(SYS3    ) CC(   0) </rsp>
<rsp>AREA(DB21AR10) MBR(SYS3    ) CC(   0) </rsp>
<rsp>AREA(DB21AR11) MBR(SYS3    ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The command stops access to all the areas that match the area name specified and makes them unavailable. The areas data sets are closed and deallocated.

Example 2 of UPDATE AREA command

TSO SPOC input:

```
UPD AREA NAME(*) STOP(SCHD)
```

TSO SPOC output:

```
Log for . . : UPD AREA NAME(*) STOP(SCHD)
 IMSplex . . . . . : PLEX1
 Routing . . . . . :
 Start time. . . . : 2003.132 09:22:15.79
 Stop time . . . . : 2003.132 09:22:17.53
 Return code . . . : 00000000
 Reason code . . . : 00000000
 Command master. .  IMS2
```

OM API input:

```
CMD(UPD AREA NAME(*) STOP(SCHD))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1   </xmlvsn>
<statime>2003.132 16:22:15.799896</statime>
<stotime>2003.132 16:22:17.536107</stotime>
<staseq>B968A44169058C8E</staseq>
<stoseq>B968A44310E6BE2D</stoseq>
<rqsttkn1>USRT005 10092215</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
```

```
<kwd>AREA          </kwd>
<input>UPD AREA NAME(*) STOP(SCHD) </input>
</cmd>
</imsout>
```

Explanation: The command is routed to IMS2 and SYS3, and the command is successful at both IMS systems. No response lines are returned because the default action for NAME(*) is to return response lines only for the area names that resulted in an error. OPTION(ALLRSP) can be specified to obtain all the area names processed on the command.

Example 3 of UPDATE AREA command

TSO SPOC input:

```
UPD AREA NAME(DD01AR0) STOP(ACCESS) SCOPE(ALL)
```

TSO SPOC output:

```
AreaName  MbrName  CC  Global
DD01AR0   IMS1     0      Y
DD01AR0   IMS1     0
DD01AR0   IMS2     0
```

Explanation: The UPDATE AREA STOP(ACCESS) command for area DD01AR0 is successfully processed at IMS1 and IMS2. Command master IMS1 successfully updates the global status.

Example 4 of UPDATE AREA command

TSO SPOC input:

```
UPDATE AREA NAME(DB22AR0,DB22AR1,DB22AR2,DB22AR3) START(ACCESS) OPTION(OPEN)
```

TSO SPOC output:

```
AreaName MbrName   CC   CCText
DB22AR0  IMS1      0
DB22AR1  IMS1      0
DB22AR2  IMS1      A9   DB OR AREA AUTHORIZATION ERROR
DB22AR3  IMS1      A9   DB OR AREA AUTHORIZATION ERROR
```

Explanation: This command opens the DEDB area data sets that are specified on the NAME() parameter, even if the area is not registered on DBRC as PREOPEN. The OPTION(OPEN) is processed locally by all IMS systems that receive the command and is not maintained as a global status in RM.

Example 5 of UPDATE AREA command

TSO SPOC input:

```
UPDATE AREA NAME(AXYZ01) START(QUIESCE) OPTION(HOLD)
```

TSO SPOC output:

```
AreaName  MbrName  CC
AXYZ01    IM02     0
AXYZ01    IM01     0
AXYZ01    IM03     0
```

Explanation: This example is of a successful quiesce and hold for a DEDB area.

Example 6 of UPDATE AREA command

TSO SPOC input:
```
UPD AREA NAME(AXYZ01) STOP(QUIESCE)
```

TSO SPOC output:
```
AreaName   MbrName      CC
AXYZ01     IM03         0
AXYZ01     IM02         0
AXYZ01     IM01         0
```

Explanation: example is of releasing the quiesce for a DEDB area.

Example 7 of UPDATE AREA command

TSO SPOC input:
```
UPDATE AREA NAME(D0010001) START(ACCESS) SET(ACCTYPE(READ))
```

TSO SPOC output:
```
 AreaName MbrName     CC
 D0010001 IMS1         0
```

Explanation: DEDB DEDBJ0001 has database access of update. Area D0010001 of DEDB DEDBJ001 access is changed from update to read.

Related concepts:

➡ Maintaining global information for databases, DEDB areas, and transactions (System Administration)

➡ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

"/START AREA command" on page 828

"/STOP AREA command" on page 883

UPDATE DATAGRP command

Use the UPDATE DATAGRP command to allow changes to the status of the members of a data group.

Subsections:
- "Environment"
- "Syntax" on page 1022
- "Keywords" on page 1022
- "Usage notes" on page 1026
- "Equivalent IMS type-1 commands" on page 1026
- "Output fields" on page 1026
- "Return, reason, and completion codes" on page 1027
- "Examples" on page 1035

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE DATAGRP command and keywords can be issued.

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE DATAGRP	X	X	
NAME	X	X	
OPTION	X	X	
SCOPE	X	X	
START	X	X	
STOP	X	X	

Syntax
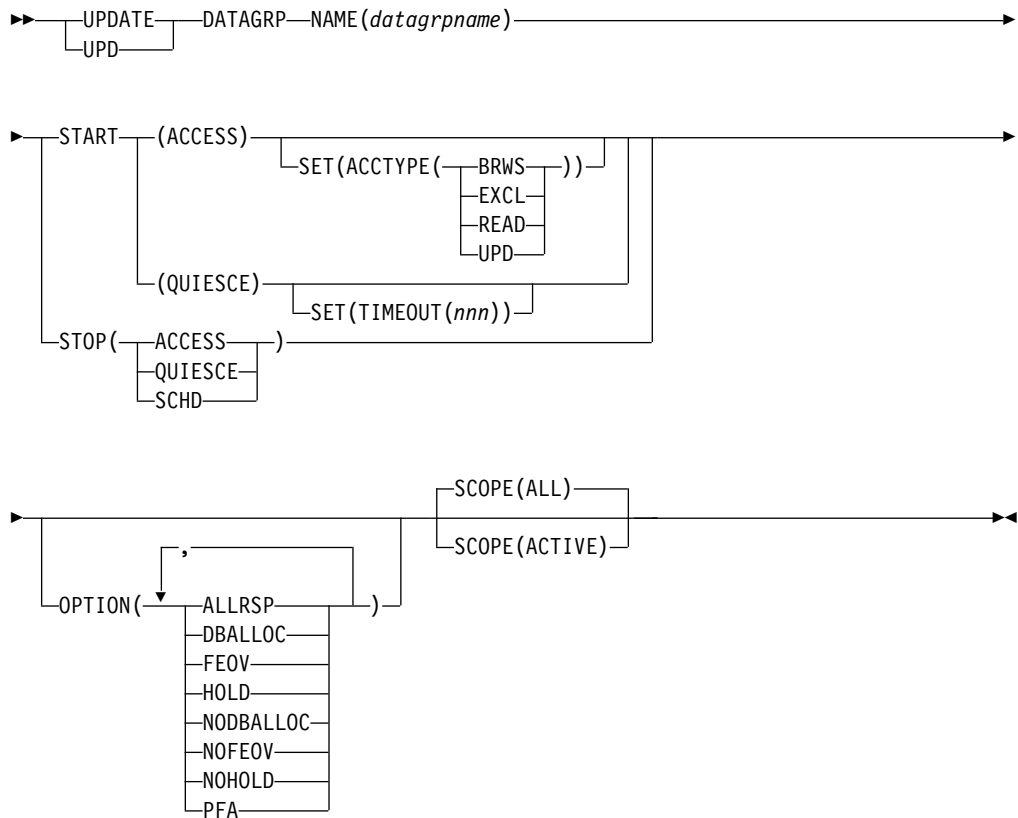
Keywords

The following keywords are valid for the UPDATE DATAGRP command:

NAME()
> Specifies the name of the data group to be processed.

OPTION()
> Specifies the additional functions to be performed. Following is a list of additional functions:

> **ALLRSP**
>> Indicates that the response lines are to be returned for all the members of the data group (all databases, areas, or both) that are processed on the command.

The default action is to return response lines only for the databases and areas or both that resulted in an error. A response line with the data group name is always returned.

DBALLOC

Indicates that the databases in the data group are to be allocated when they are started.

DBALLOC can only be specified with START(ACCESS).

Only one of DBALLOC or NODBALLOC can be specified.

FEOV Indicates to force end of volume after the command processing is complete. The IMS log switches to the next OLDS, and a simple checkpoint is taken unless specified with STOP(QUIESCE). This switch is marked as a recovery point for log archiving purposes. A simple checkpoint is not taken when specified with STOP(QUIESCE).

FEOV is valid only with START(QUIESCE) OPTION(NOHOLD) or STOP(ACCESS) or STOP(QUIESCE).

FEOV is not valid on an RSR tracker.

OPTION(FEOV) is the default when specified with START(QUIESCE) OPTION(NOHOLD) or STOP(QUIESCE).

For quiesce processing, the end of volume will be forced after the quiesce point has been reached, causing the logs to be switched before the quiesce is released, so that any new updates occur on the new IMS log. This occurs during the stop quiesce processing of a START(QUIESCE) OPTION(NOHOLD) or when a STOP(QUIESCE) is issued to release a quiesce from a previous START(QUIESCE) OPTION(HOLD).

HOLD

Specifies that after the quiesce has been achieved successfully the data group members should remain quiesced. A subsequent STOP(QUIESCE) would be required to release the quiesce on the data group members. This keyword is only valid with the START(QUIESCE) keyword. HOLD is mutually exclusive with NOHOLD.

NODBALLOC

Indicates that the databases in the data group are not to be allocated when they are started. The databases will be allocated when they are scheduled.

NODBALLOC can only be specified with START(ACCESS). NODBALLOC is the default action for UPDATE DATAGRP command if OPTION(DBALLOC) is not specified.

Only one of DBALLOC or NODBALLOC can be specified.

NOFEOV

Indicates to not force end of volume after the command processing is complete. The IMS log does not switch to the next OLDS, and a simple checkpoint is not taken.

NOFEOV is valid only with START(QUIESCE) OPTION(NOHOLD), STOP(ACCESS), or STOP(QUIESCE). OPTION(NOFEOV) is the default when specified with STOP(ACCESS).

NOHOLD
Specifies that after the quiesce has been achieved successfully the data group members should automatically release the quiesce. If you do not specify either HOLD or NOHOLD, NOHOLD is assumed. This keyword is only valid with the START(QUIESCE) keyword. NOHOLD is mutually exclusive with HOLD.

SCOPE()
Specifies where IMS should apply the change.

ACTIVE
SCOPE(ACTIVE) specifies that the changes are to be applied to the IMS systems that are currently active and to which the command is routed.

This option does not apply to the quiesce function. The scope of a quiesce is always all instances of the data group usage across the IMSplex. There is no quiesce which would only apply to a subset of the IMSplex.

ALL SCOPE(ALL) specifies that the changes are to be applied to the active IMS systems.

This option does not apply to the quiesce function. The scope of a quiesce is always all instances of the data group members usage across the IMSplex. There is not a global status of QUIESCE for a data group or its members. Using SCOPE(ALL) for a START(QUIESCE) or STOP(QUIESCE) does not change the global status for the data group members.

Currently, ACTIVE and ALL produce the same results.

SET() Specifies the attribute values to be changed or sets the database state.

ACCTYPE
Specifies the access intent for the named database. This keyword can be specified only if START(ACCESS) is specified. This keyword overrides the database access intent specified at system definition.

BRWS Specifies that the named database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data because another program could be updating the database. The database is opened for read-only processing.

EXCL Specifies that the named database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC.

READ Specifies that the named database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. ACCTYPE of READ differs from ACCTYPE of BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UPD Specifies that the named database is available for update as well as read processing in this IMS subsystem.

TIMEOUT(*nnn*)
Specifies the number of seconds to wait before a timeout occurs in a data group quiesce. The timeout value can be 1 - 999 seconds. The TIMEOUT parameter value can override the DBQUIESCETO parameter in the DFSCGxxx member of the IMS PROCLIB data set. If the TIMEOUT parameter is omitted and the DBQUIESCETO parameter is not specified, the default timeout value is 30 seconds. The TIMEOUT keyword is valid only with the START(QUIESCE) keyword.

START()
Specifies the attributes that are to be started.

ACCESS
Specifies that the members of the specified data group name are to be started. See UPDATE AREA START(ACCESS) and UPDATE DB START(ACCESS) for a description of the actions performed on the databases and areas in the data group.

Additional functions to be performed along with START(ACCESS) can be specified using the OPTION keyword.

An UPDATE DATAGRP START(ACCESS) command with SET(ACCTYPE) or OPTION(DBALLOC | NODBALLOC | NOBACKOUT | OPEN | NOOPEN) is invalid on the RSR tracker.

QUIESCE
Specifies that the data group named on the command is to be quiesced to establish a new recovery point. The scope of a quiesce is always all instances of the data group usage across the IMSplex.

STOP()
Specifies the attributes to be stopped.

ACCESS
Specifies that offline processing is to be done for the members of the specified data group. See UPDATE AREA STOP(ACCESS) and UPDATE DB STOP(ACCESS) for a description of the actions performed on the databases and areas in the data group.

QUIESCE
Specifies that the data group named on the command should be made available again by releasing the quiesce on the data group members.

Unlike START(QUIESCE), where each data group resource listed must be quiesced in order for the command to complete successfully, STOP(QUIESCE) continues to process each listed resource even if some resources are not in quiesced state or cannot be released from quiesced state. For those resources that are not in quiesced state or cannot be released from quiesced state, the command returns a response line for each of those resources.

SCHD Specifies that the members of the specified data group name are to be stopped. See UPDATE AREA STOP(ACCESS) and UPDATE DB STOP(ACCESS) for a description of the actions performed on the databases and areas in the data group.

Usage notes

The UPDATE DATAGRP command can only be specified through the OM API and can be processed only by the DB/DC and DBCTL environments. In addition, the UPDATE DATAGRP command is not allowed on the XRF alternate.

When the UPDATE DATAGRP command is issued, it only applies to the IMS system to which it is routed. It does not apply to the other IMS systems in the IMSplex that share the database or areas in the data group. The routing of the UPDATE DATAGRP command is done by OM. OM routes the command to all active DB/DC or DBCTL IMS systems, unless specific routing is specified. OM selects one IMS as the command master.

A response line is returned for the data group name from each IMS. Response lines are also returned for each data group member that resulted in an error. No response lines are returned for the data group members that are processed successfully unless OPTION(ALLRSP) is specified.

A *data group* is defined in the RECON data set by using the INIT.DBDSGRP command with the parameters GRPNAME and DBGRP (to define a DB group), MEMBERS (to define a DBDS group), or RECOVGRP (to define a recovery group). The DATAGRP keyword on the UPDATE command can specify either a DBDS group or a DB group (DL/I databases or DEDB areas).

If the ACCESS keyword is specified on the UPDATE DATAGRP command along with the DBDS group name, the ACCESS keyword is not applied to the Fast Path DEDB databases associated with the Fast Path DEDB areas in the DBDS group. If the intent is to use the ACCESS keyword for Fast Path DEDB databases, the DATAGRP parameter must be a data group name that does not contain area names.

While the database is being quiesced, this command with keywords other than STOP(QUIESCE) cannot be processed successfully.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE DATAGRP command and the type-1 IMS commands that perform similar functions.

Table 394. Type-1 equivalents for the UPDATE DATAGRP command

UPDATE DATAGRP command	Similar IMS type-1 command
UPDATE DATAGRP NAME(name) STOP(ACCESS)	/DBR DATAGRP datagrpname
UPDATE DATAGRP NAME(name) START(ACCESS)	/START DATAGRP datagrpname
UPDATE DATAGRP NAME(name) STOP(SCHD)	/STOP DATAGRP datagrpname

Output fields

The following table shows the output fields for the UPDATE DATAGRP. The columns in the table are as follows:

Short label
>   Contains the short label generated in the XML output.

Keyword
>   Identifies the keyword on the command that caused the field to be

generated. N/A appears for output fields that are always returned. ERR
appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 395. Output fields for UPDATE DATAGRP command

Short label	Keyword	Meaning
AREA	N/A	Area name. The Area name is returned if there are one or more areas in the data group.
CC	N/A	Completion code for the line of output. Completion code is always returned.
CCTXT	*error*	The completion code text that briefly explains the meaning of the completion code.
DB	N/A	Database name. The database name is returned if there are one or more databases in the data group.
DG	N/A	Data group name. The data group name is always returned.
ERRT	ERR	Error text returned to add more meaning to the completion code and may include a return code from a service. The error text is only returned if the completion code is non-zero.
MBR	N/A	The IMSplex member that built output line. The IMS identifier of the IMS for which the database information is displayed. The IMS identifier is always returned.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the UPDATE DATAGRP
command. The OM return and reason codes that may be returned as a result of the
UPDATE DATAGRP command are standard for all commands entered through the
OM API.

The following table includes the return and reason codes and a brief explanation of
the codes. The return or reason code returned for the command might also indicate
an error from a CSL request.

Table 396. Return and reason codes for the UPDATE DATAGRP command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE DATAGRP command completed successfully.
X'00000008'	X'00002004'	An invalid keyword or more than one keyword is specified on the UPDATE DATAGRP command.
X'00000008'	X'00002014'	The UPDATE DATAGRP command is not processed because an invalid character is found in the data group name.

Table 396. Return and reason codes for the UPDATE DATAGRP command (continued)

Return code	Reason code	Meaning
X'00000008'	X'00002040'	More than one filter or keyword value is specified on the UPDATE DATAGRP command. Either more than one keyword or an invalid combination of filters was specified. For example, START(ACCESS) and STOP(ACCESS) was specified, or OPTION(DBALLOC,NODBALLOC) was specified. Check the input command and reenter the correct combinations.
X'00000008'	X'00005000'	The UPD DATAGRP command processing terminated because IMODULE GETSTOR storage could not be obtained.
X'00000008'	X'0000501C'	The UPD DATAGRP command processing terminated because IMODULE GETMAIN storage could not be obtained.
X'0000000C'	X'00003000'	The UPDATE DATAGRP command is successful for at least one member in the data group. The UPDATE DATAGRP command is not successful for one or more members in the data group. The completion code indicates the reason for the error with the data group member name. The completion codes that can be returned by the UPDATE DATAGRP command are listed in the UPDATE DATAGRP completion code table.
X'0000000C'	X'00003004'	The UPDATE DATAGRP command is not successful for all the members in the data group. The completion code indicates the reason for the error with the data group member name. The completion codes that can be returned by the UPDATE DATAGRP command are listed in the UPDATE DATAGRP completion code table.
X'00000010'	X'0000400C'	The UPDATE DATAGRP command is not processed because it is not valid on the XRF alternate.
X'00000010'	X'00004014'	The UPDATE DATAGRP command is not processed because it is not valid on the RSR tracker.
X'00000010'	X'00004200'	Commands are not processed because IMS shutdown is in progress.
X'00000010'	X'00004208'	Invalid command for RMENV=NO.
X'00000010'	X'00004400'	MINVERS in the RECON data sets is not 11.1.
X'00000014'	X'00005004'	The UPDATE DATAGRP command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	The UPDATE DATAGRP command processing terminated because DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	The UPDATE DATAGRP command processing terminated because AWE could not be obtained.
X'00000014'	X'00005014'	The UPDATE DATAGRP command processing terminated because BCB storage could not be obtained.
X'00000014'	X'00005FFF'	The UPDATE DATAGRP command processing terminated because of an internal error.

Table 396. Return and reason codes for the UPDATE DATAGRP command (continued)

Return code	Reason code	Meaning
X'02000008'	X'0000203C'	An invalid TIMEOUT value is specified. The value must be a numeric value between 1 and 999.

The following table includes an explanation of the completion codes. Errors unique to the processing of UPDATE DATAGRP command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 397. Completion codes for the UPDATE DATAGRP command

Completion code	Completion code text	Meaning
0		The command completed successfully for the resource.
8	COMMAND COMPLETE FOR SOME	Some. The command completed with error for some of the AREAS of the DEDB. Response lines for the area names in error are returned.
C	COMMAND COMPLETE FOR NONE	None. The command completed with error for all the AREAs of the DEDB. Response lines for the area names in error are returned.
10	NO RESOURCES FOUND	No resource found. Database name is invalid, or the wildcard parameter specified does not match any database names.
11	DUPLICATE RESOURCE NAME	Duplicate resource name. The resource name is specified multiple times on the command and is ignored.
17	ANOTHER CMD IN PROGRESS	This error is returned when the quiesce could not be started because another database command was in progress.
23	DB STOP ACCESS IN PROGRESS	A /DBRECOVERY, or UPDATE DB STOP(ACCESS) command to stop database access is in progress. This takes the database offline.
25	DB STOP UPDATES IN PROGRESS	A /DBDUMP or UPDATE DB STOP(UPDATES) command to stop database updates is in progress.
26	DEDB STOP IN PROGRESS	/DBRECOVERY, /STOP, or UPDATE DB STOP(SCHD) command to stop database scheduling is in progress for a DEDB.
31	NOT ALLOWED FOR A DEDB	Database is a DEDB. The command entered is not valid for the DEDB in the IMS environment.
32	NOT ALLOWED FOR AN MSDB	Database is a MSDB. The command entered is not valid for the MSDB in the IMS environment.

Table 397. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
33	NOT ALLOWED FOR A HALDB MASTER	Command invalid HALDB master. The command OPTION is invalid for the HALDB master but partition structure rebuild will be done if structure rebuild is needed and if only one HALDB master was specified in the command. No rebuild will be attempted if there is more than one database name listed in the command. If there are multiple database names listed in the command and all are invalid except the HALDB master, then rebuild will be attempted if needed.
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for an IMS descriptor or resource. DFSDSDB1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
53	NO RM ADDRESS SPACE	This error is returned when the quiesce could not be started because the RM address space is not present.
55	NO FAST PATH INSTALLED	The command failed because Fast Path is not installed.
56	FF DB + LSO=Y + TRK = ERROR	Command is invalid on the RSR tracker because of the LSO=Y option.
65	DMB POOL STORAGE ERROR	DMB pool storage error. The command failed because of DMB pool storage request failure.
66	DMB POOL FULL	DMB pool full. The command failed because the DMB pool was full.
6C	NOT ALLOWED FOR A HALDB PARTITION	An UPDATE command specified a change to the residency option for a HALDB partition. The residency option is valid only for the master and not the partitions.
6F	REFERENCED BY PROGRAM	An UPDATE DB command is issued to change the resident option. There is a currently scheduled program that is referencing that database. The UPD command fails.
76	RECOVER CMD ACTIVE	/RECOVER START command is in progress to recover one or more databases with the database recovery services.
81	DBRC ERROR	DBRC error.
90	INTERNAL ERROR	The command entered is not processed because of an internal error.
91	TIMEOUT ERROR	This error is returned when the quiesce could not be completed within the timeout period.

Table 397. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
92	COMMAND PROCESSING ERROR	Command processing error. The command entered is not processed because of a command error. A unique completion code could not be generated to explain the error. The message number and the return code that could not be converted to a completion code are listed in the error text.
A0	DYNAMIC ALLOCATION FAILED	Dynamic allocation failed. The command entered is not processed because the dynamic allocation failed for the DB.
A1	DB IS AUTHORIZED BY BATCH	Database is authorized by Batch. The command entered is not processed because the database is authorized by batch.
A2	DB IS AUTHORIZED BY ANOTHER IMS	Database is authorized by another IMS. The command entered is not processed because the database is authorized by another active or abnormally terminated IMS and its authorization state is incompatible with the current authorization request.
A3	AUTHORIZATION CHANGE FAILED	Authorization change failed. The DBRC CHNGAUTH request resulted in an error.
A4	DATABASE NOT REGISTERED TO DBRC	Database not registered to DBRC. The command processing failed as the database is not registered to DBRC.
A5	PREVENT FURTHER AUTH ON	Prevent Further Auth ON. The command entered is not processed because the database or area is defined to DBRC as prevent further Auth.
A6	INVALID DATABASE RECORD IN RECON	Invalid database record in RECON. The command entered is not processed because an invalid parameter was found during the evaluation process of the database usage compatibility. The database record might be invalid in the RECON data set.
A7	DBRC UNAUTH FAILED FOR CHNGAUTH	DBRC unauth failed during change authorization. The command is not processed because of an error during UNAUTH processing during change authorization request.
A8	INVALID DB RECORD IN RECON	Invalid database record in RECON. An UPDATE DB SET(ACCTYPE) command is entered to change the database authorization level. An encoded state of zero is returned by DBRC during the change authorization processing.
A9	DB OR AREA AUTHORIZATION ERROR	Database or area authorization error. For a database, the command entered is not processed because of a database authorization error. For an area, area authorization to DBRC failed.
AA	DB IN USE-BMP	The UPDATE DB command is rejected because the database is in use by a BMP.

Table 397. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
AB	DB IN USE-DBCTL LONG THREAD	The UPDATE DB command is rejected because the database is in use by a long-running DBCTL thread.
AC	FP AREA HELD-LONG BUSY WAIT	The UPDATE DB command is rejected because the area of the DEDB is in long-busy wait.
AD	DYNAMIC UNALLOCATION FAILED	The UPDATE DB command is not successful because of a dynamic unallocation error.
AE	DYNAMIC ALLOCATION ERROR	The UPDATE DB command is not successful because of a dynamic allocation error. No SVC99 is issued.
C1	OLR DDIR MISSING OR DFSPNT ZERO	Unknown DMB referenced for database. The command cannot be processed because an unknown data management block is referenced for the database. Refer to the DFS564I message put out to the system console to identify the DMB name that cannot be referenced.
CC	OLR IS ACTIVE FOR DATABASE	OLR is active for database. The command failed as OLR is active for the database.
D0	DATABASE CLOSE ERROR	Database close error. The command processing failed because of a database close error.
D1	DATABASE WRITE ERROR	Database write error. The command processing failed because of a database write error.
D2	DATABASE NEEDS BACKOUT	Database needs backout. The command processing failed as the database needs backout.
D3	DATABASE OR AREA NEEDS RECOVERY	Database or AREA needs recovery. The command processing failed as the database or area needs recovery.
D4	DATABASE NEEDS IMAGE COPY	Database needs image copy. The command processing failed as the database needs image copy.
D5	DATABASE HAS NO BACKOUTS	Database has no backouts. The command processing failed as there are no backouts for the database.
D6	DATABASE IN USE	Database in use. A SET(ACCTYPE) is specified for the DEDB and the authorization level cannot be changed as the DEDB is in use in a region.
D7	DB I/O PREVENTION NOT COMPLETE	Database I/O prevention not complete. The database cannot be started as it is extended because of a XRF takeover and the I/O prevention is not complete.
D8	DATABASE BACKOUTS PENDING	Database backouts pending. The access type specified for the database cannot be changed as restartable backouts are pending for the database.
D9	DATABASE/AREA OPEN FAILED	Database or area open failed. The command failed because an error occurred while opening the database or area.

Table 397. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
DA	DATABASE BEING RECALLED BY HSM	Database being recalled from HSM. The command processing failed because the database is being recalled from HSM.
DB	PARTITION OPEN FAILED	Partition open failed. The partition open failed because the master is offline. This can also occur if the partition has been deleted and partition structure rebuild has occurred. Partition structure rebuild can be accomplished by issuing an UPD DB NAME(haldbmst) START(ACCESS) OPTION(OPEN) command, where haldbmst is the partition's master, or by issuing a qualified GU call for a key in the key range of the partition. List.recon can be used to determine if the partition exists or has been deleted.
DC	HALDB PARTITION BUILD FAILURE	Database partition build failure. The database partition build for the DDIR or DMB failed. Refer to the DFS0415I message sent to the system console to determine the reason of the failure.
DD	HALDB PARTITION INIT FAILURE	Database partition initialization failed. The database partition initialization for the DDIR or DMB failed. Refer to the DFS0415 message sent to the system console for the details.
DE	ACBLIB READ FAILURE	ACBLIB read failure. The command is not processed because there was an error reading the ACBLIB.
DF	DB DIRECTORY INIT FAILURE	Database directory initialization failed. The command is not processed because of a database directory initialization failure.
E0	DATABASE OR AREA IN RECOVERY	Database or area in recovery. The command is not processed because the database or area is in recovery.
E1	DATABASE HAS NOT BEEN DBR'ED	An UPDATE command changing the residency option of a database was not issued for a database that has not been DBR'ed.
E2	PARALLEL DB OPEN NOT COMPLETE	Restart parallel DB open not complete. The command is not processed because the restart parallel DB open is not complete for the database.
E5	PARTICIPANT UNABLE TO QUIESCE	This error is returned when the quiesce could not be completed successfully across the IMSplex. The IMS with this completion code was the quiesce participant which was not able to be quiesced.
E6	QUIESCE COMMUNICATION FAILURE	This error is returned when the quiesce could not be completed because of a failure to communicate across the IMSplex. There could be a problem with RM, OM, or SCI that has caused the communication failure to occur.

Table 397. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
E7	CMD NOT ALLOWED	This error is returned when the database command could not be processed because a quiesce command was in progress.
E8	DATABASE HAS INTENT TO REORGANIZE	This error is returned when the quiesce could not be started because the RECON data sets indicates that there is intent to reorganize the database.
E9	DB IN WRONG STATE TO BE QUIESCED	This error is returned when the named resource is in the wrong state for quiesce processing to proceed.
EE	DATABASE BACKOUT ERROR	Database backout error. The command processing failed because of a database backout error.
EF	DATABASE IS IN ERROR	Database is in error. The command entered is not processed because the database is in error.
F0	NO AREA LOCK	No AREA lock. The command processing failed because the area lock could not be obtained.
F1	AREA NOT STOPPED	Area not stopped. The command entered is not processed because the AREA is not stopped.
F2	PRELOAD IS ACTIVE FOR AREA	Preload is active for AREA. The command is not processed because preload is active for the AREA.
F3	UNRESOLVED INDOUBTS FOR AREA	Unresolved indoubts for AREA. The command entered is not processed because unresolved indoubts exist for the AREA.
F4	ALLOCATION FAILED	Allocation failed. Allocation failed for the AREA name.
F5	AREA NEEDS RECOVERY	AREA needs recovery. The command processing failed because the area needs recovery.
F6	ADS NUMBER DISCREPANCY	ADS number discrepancy. The command failed for the AREA because there is a discrepancy between the number of ADS allocated by IMS and the number of ADS known to DBRC. Correct the discrepancy and reissue the command to deallocate the data sets.
F7	AREA IS NOT LOADED INTO CF	Area is not loaded into CF. The AREA OPEN failed and is not loaded into the Coupling Facility.
F8	AREA HAS I/O TOLERATED CI	Area has I/O tolerated CI. The command is not processed because the AREA has an I/O tolerated CI.
F9	AREA HAS 2ND CI EEQE	Area has second CI EEQE. The command is not processed because the AREA has a second CI EEQE.
FC	UTILITY ACTIVE ON AREA	This error is returned when the area is in use by a utility.
FD	AREA HAS EEQE	This error is returned when the area has an extended error queue element (EEQE).

Table 397. Completion codes for the UPDATE DATAGRP command (continued)

Completion code	Completion code text	Meaning
FE	AREA HAS EQE	This error is returned when the area has an error queue element (EQE).

Examples

The following are examples of the UPDATE DATAGRP command:

Example 1 for UPDATE DATAGRP command

TSO SPOC input:

```
UPD DATAGRP NAME(GROUP1) STOP(ACCESS)
```

TSO SPOC output:

```
DataGroup DBName    AreaName MbrName    CC
GROUP1                       IMSA        0
GROUP1                       IMS1        8
GROUP1    DEDBJN03           IMS1        C
GROUP1    DEDBJN03 DB3AREA0 IMS1        F0
```

OM API input:

```
CMD(UPD DATAGRP NAME(GROUP1) STOP(ACCESS))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2003.129 22:45:55.898843</statime>
<stotime>2003.129 22:45:55.990001</stotime>
<staseq>B965346AB45DB229</staseq>
<stoseq>B965346ACA9F1263</stoseq>
<rqsttkn1>USRT005 10154555</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>

</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntext>At least one request successful</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMSA    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>DATAGRP         </kwd>
<input>UPD DATAGRP NAME(GROUP1) STOP(ACCESS) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DG" llbl="DataGroup" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="3" scroll="no"
```

```
len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DG(GROUP1  ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>DB(DEDBJN03) AREA(DB3AREA0) DG(GROUP1  ) MBR(IMS1    ) CC(  F0)
</rsp>
<rsp>DB(DEDBJN03) DG(GROUP1  ) MBR(IMS1    ) CC(   C) </rsp>
<rsp>DG(GROUP1  ) MBR(IMS1    ) CC(   8) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The command stops access to all the members of data group GROUP1 and takes them offline. The command response lines are returned for all the databases or areas that resulted in an error. No response lines are returned for members for which the command was successful.

Example 2 for UPDATE DATAGRP command

TSO SPOC input:
```
UPD DATAGRP NAME(GROUP1) START(ACCESS) OPTION(ALLRSP)
```

TSO SPOC output:

DataGroup	DBName	AreaName	MbrName	CC
GROUP1			IMSA	0
GROUP1		DB21AR1	IMSA	0
GROUP1	BE3PARTS		IMSA	0
GROUP1	DEDBJN03		IMSA	0
GROUP1			IMS1	0
GROUP1		DB21AR1	IMS1	0
GROUP1	BE3PARTS		IMS1	0
GROUP1	DEDBJN03		IMS1	0

OM API input:
```
CMD(UPD DATAGRP NAME(GROUP1) START(ACCESS) OPTION(ALLRSP))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2003.129 22:46:51.490484</statime>
<stotime>2003.129 22:46:51.672332</stotime>
<staseq>B965349FB88B4445</staseq>
<stoseq>B965349FE4F0C36A</stoseq>
<rqsttkn1>USRT005 10154651</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMSA    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>DATAGRP          </kwd>
<input>UPD DATAGRP NAME(GROUP1) START(ACCESS) OPTION(ALLRSP) </input>
```

```
</cmd>
<cmdrsphdr>
<hdr slbl="DG" llbl="DataGroup" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DB(DEDBJN03) DG(GROUP1 ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>DB(BE3PARTS) DG(GROUP1 ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>AREA(DB21AR1 ) DG(GROUP1 ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>DG(GROUP1 ) MBR(IMSA    ) CC(   0) </rsp>
<rsp>DB(DEDBJN03) DG(GROUP1 ) MBR(IMS1    ) CC(   0) </rsp>
<rsp>DB(BE3PARTS) DG(GROUP1 ) MBR(IMS1    ) CC(   0) </rsp>
<rsp>AREA(DB21AR1 ) DG(GROUP1 ) MBR(IMS1    ) CC(   0) </rsp>
<rsp>DG(GROUP1 ) MBR(IMS1    ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The command starts access of all the members of data group, GROUP1, and makes them available. The command response lines are returned for all the databases or areas that are processed with OPTION(ALLRSP) specified.

Example 3 for UPDATE DATAGRP command

TSO SPOC input:
```
UPDATE DATAGRP NAME(DBGXYZ) START(QUIESCE) OPTION(HOLD)
```

TSO SPOC output:

DataGroup	DBName	MbrName	CC
DBGXYZ	DB1XYZ	IM02	0
DBGXYZ	DB2XYZ	IM02	0
DBGXYZ		IM02	0
DBGXYZ	DB1XYZ	IM01	0
DBGXYZ	DB2XYZ	IM01	0
DBGXYZ		IM01	0
DBGXYZ	DB1XYZ	IM03	0
DBGXYZ	DB2XYZ	IM03	0
DBGXYZ		IM03	0

Explanation: This example is of a successful quiesce-and-hold for a data group.

Example 4 for UPDATE DATAGRP command

TSO SPOC input:
```
UPD DATAGRP NAME(DBGXYZ) START(QUIESCE)
```

TSO SPOC output:

DataGroup	DBName	MbrName	CC
DBGXYZ	DB1XYZ	IM02	0
DBGXYZ	DB2XYZ	IM02	0
DBGXYZ		IM02	0
DBGXYZ	DB1XYZ	IM01	0
DBGXYZ	DB2XYZ	IM01	0

```
DBGXYZ              IM01       0
DBGXYZ    DB1XYZ    IM03       0
DBGXYZ    DB2XYZ    IM03       0
DBGXYZ              IM03       0
```

Explanation: This example is of a successful quiesce-and-go for a data group.

Example 5 for UPDATE DATAGRP command

TSO SPOC input:
```
UPD DATAGRP NAME(DBGXYZ)  STOP(QUIESCE)
```

TSO SPOC output:
```
DataGroup  DBName    MbrName    CC
DBGXYZ     DB1XYZ    IM02       0
DBGXYZ     DB2XYZ    IM02       0
DBGXYZ               IM02       0
DBGXYZ     DB1XYZ    IM01       0
DBGXYZ     DB2XYZ    IM01       0
DBGXYZ               IM01       0
DBGXYZ     DB1XYZ    IM03       0
DBGXYZ     DB2XYZ    IM03       0
DBGXYZ               IM03       0
```

Explanation: This example is of releasing the quiesce for a data group.

Related concepts:

How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

Command keywords and their synonyms (Commands)

"/START DATAGRP command" on page 833

"/STOP DATAGRP command" on page 889

UPDATE DB command

Use the UPDATE DB command to update status or definition information about
databases. For example, UPDATE DB can make a database available, take the
database offline, stop scheduling, stop updates, lock, and unlock a database.

Subsections:
- "Environment"
- "Syntax" on page 1039
- "Keywords" on page 1040
- "Usage notes" on page 1050
- "Equivalent IMS type-1 commands" on page 1052
- "Output fields" on page 1052
- "Return and reason codes" on page 1053
- "Examples" on page 1063

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from
which the UPDATE DB command and keywords can be issued.

Table 398. Valid environments for the UPDATE DB command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE DB	X	X	
NAME	X	X	
OPTION	X	X	
SCOPE	X	X	
SET	X	X	
START	X	X	
STOP	X	X	

Syntax

```
►►──┬─UPDATE─┬──DB──NAME──┬─(─┬──────dbname*──────┬─)─┬────────────────────────────►
    └─UPD────┘            │   └─────────,◄────────┘   │
                          └─(*)───────────────────────┘

►──┬─START──┬─(ACCESS)──┬──────────────────────────────────┬──┬──────────┬───┬────►
   │        │           └─SET(ACCTYPE(─┬─BRWS─┬─))─┘        │  └─AREA(*)──┘   │
   │        │                          ├─EXCL─┤             │                 │
   │        │                          ├─READ─┤             │                 │
   │        │                          └─UPD──┘             │                 │
   │        └─(QUIESCE)──┬───────────────────────┬─────────────────────────────┤
   │                     └─SET(TIMEOUT(nnn))─┘                                  │
   │                                                                           │
   ├─STOP──┬─(ACCESS)───┬──────────────────────────────────────────────────────┤
   │       ├─(SCHD)─────┤                                                       │
   │       ├─(UPDATES)──┤                                                       │
   │       └─(QUIESCE)──┘                                                       │
   ├─SET(LOCK(─┬─ON──┬─))────────────────────────────────────────────────────────┤
   │           └─OFF─┘                                                          │
   └─SET(RESIDENT(─┬─N─┬─))──────────────────────────────────────────────────────┘
                   └─Y─┘


                                          ┌─SCOPE(ALL)────┐
►──┬────────────────────────────────┬──┬──┴───────────────┴──┬─────────────────►◄
   │                                │  └─SCOPE(ACTIVE)─┘
   │              ┌─────,◄──────┐    │
   └─OPTION(──┬──┬─ALLRSP───┬──┬─)─┘
              │  ├─DBALLOC───┤
              │  ├─FEOV──────┤
              │  ├─HOLD──────┤
              │  ├─NOBACKOUT─┤
              │  ├─NODBALLOC─┤
              │  ├─NOFEOV────┤
              │  ├─NOHOLD────┤
              │  ├─NOOPEN────┤
              │  ├─NORAND────┤
              │  ├─OPEN──────┤
              │  └─PFA───────┘
```

Keywords

The following keywords are valid for the UPDATE DB command:

AREA(*)

> Specifies the areas of the databases that are to be processed or, if you specify AREA(*), indicates that the command applies to all the areas in the database.
>
> The UPDATE DB START(ACCESS) AREA(*) command starts the DEDB and all of its areas and allows scheduling of new applications against the DEDB.
>
> The AREA(*) keyword is ignored for non-DEDB databases. You cannot specify both the AREA(*) and the NAME(*) keywords. If you specify the AREA(*) keyword with the START(ACCESS) SET(ACCTYPE) keyword, all the DEDB areas are restarted after they are stopped by the START(ACCESS) SET(ACCTYPE) keyword.

NAME()

> Specifies the 1-8 character name of the database (DBD name). Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the resource names that are processed. For NAME(*) command responses are returned only for the resource names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the resource names that are processed.
>
> NAME(*) is the designed method on IMSplex commands to enable the user to specify that the command applies to all the database resources.
>
> The database names that match the generic or wildcard parameter are processed. For specific or wildcard names, response lines are returned for all the database names that are processed.
>
> When the database specified is a DEDB, a response line is returned for the DEDB name. A response line is also returned for all the areas of a DEDB that resulted in errors. A response line is not returned for the areas of a DEDB for which the command action is successful.

OPTION()

> Specifies additional functions to be performed along with the UPDATE DB command.

> **ALLRSP**
>
> > Indicates that the response lines are to be returned for all resources that are processed on the command. The default action is to return response lines only for the resources that resulted in an error. It is only valid with NAME(*) or AREA(*). ALLRSP is ignored for other NAME values.

> **DBALLOC**
>
> > Indicates that the database is to be allocated when it is started.
> >
> > DBALLOC can be specified with START(ACCESS) only. DBALLOC is the default action when specific database names are specified on the command. Only one of DBALLOC or NODBALLOC can be specified.
> >
> > OPTION(DBALLOC) is not valid for a HALDB master.

FEOV Indicates to force end of volume after the command processing is complete.

The IMS log switches to the next OLDS, and a simple checkpoint is taken unless specified with STOP(QUIESCE). This switch is marked as a recovery point for log archiving purposes. A simple checkpoint is not taken when specified with STOP(QUIESCE).

FEOV is valid only with START(QUIESCE) OPTION(NOHOLD), STOP(QUIESCE), STOP(UPDATES), or STOP(ACCESS), and does not apply to an RSR tracker.

OPTION(FEOV) is the default when specified with START(QUIESCE) OPTION(NOHOLD) or STOP(QUIESCE).

When START(QUIESCE) or STOP(QUIESCE) is specified, OPTION(FEOV) is the default; that is, the end of volume will be forced after the quiesce point has been reached. The logs are switched before the quiesce is released so that any new updates occur on the new IMS log. Switching of the logs occurs when STOP(QUIESCE) is issued to release a quiesce from a previous START(QUIESCE) OPTION(HOLD).

HOLD

Specifies that after the quiesce has been achieved successfully, the database should remain quiesced. A subsequent STOP(QUIESCE) would be required to release the quiesce on the database.

OPTION(HOLD) is valid only with the START(QUIESCE) keyword. OPTION(HOLD) is mutually exclusive with OPTION(NOHOLD).

NOBACKOUT

Indicates to suppress backout restart for a database not registered in DBRC.

NOBACKOUT can be specified with START(ACCESS) only and is not valid with NAME(*). OPTION(NOBACKOUT) is not valid for a HALDB master.

NODBALLOC

Indicates that the database is not to be allocated when it is started.

The database is allocated when it is scheduled. NODBALLOC can be specified with START(ACCESS) only. NODBALLOC is the default action when NAME(*) is specified. Only one of DBALLOC or NODBALLOC can be specified. OPTION(NODBALLOC) is ignored for HALDB masters.

NOFEOV

Indicates to not force end of volume after the command processing is complete. The IMS log does not switch to the next OLDS and a simple checkpoint is not taken.

OPTION(NOFEOV) can be specified only with START(QUIESCE) OPTION(NOHOLD), STOP(ACCESS), STOP(QUIESCE), or STOP(UPDATES). OPTION(NOFEOV) is the default except when START(QUIESCE) or STOP(QUIESCE) is specified.

NOHOLD

Specifies that after the quiesce has been achieved successfully the

database should automatically release the quiesce. If either HOLD or NOHOLD is not specified, NOHOLD is assumed.

OPTION(NOHOLD) is valid only with the START(QUIESCE) keyword. OPTION(NOHOLD) is mutually exclusive with OPTION(HOLD).

NOOPEN

Indicates that the database is not to be opened when it is started.

NOOPEN can only be specified with START(ACCESS). NOOPEN is the default action unless the database has EEQE's.

Only one of OPEN or NOOPEN can be specified. OPTION(NOOPEN) is not valid for a HALDB master.

NORAND

Indicates that the randomizer stays loaded while the UPDATE DB STOP(ACCESS) command is being processed.

This option solves the problem of the Extended Common Service Area (ECSA) becoming fragmented by continuous unloading and reloading of randomizers.

Restriction: OPTION(NORAND) only works for Fast Path DEDBs. It is ignored for full-function databases.

OPTION(NORAND) status is not maintained in the RM resource structure if global database status is maintained. If an UPD DB STOP(ACCESS) OPTION(NORAND) command is issued for a Fast Path DEDB, the STOACC status is saved in the RM resource structure. The randomizer is unloaded when the global STOACC status is applied.

OPEN Indicates that the database is to be opened when it is started. OPEN can be specified only with START(ACCESS). Specifying START(ACCESS) OPTION(OPEN) causes the randomizer routine or the selection partition routine to be loaded.

The randomizer is loaded into memory the first time it is referenced by a database. If no other database references the randomizer and the randomizer is deleted from memory, the randomizer is deleted from memory.

OPEN cannot be specified with NAME(*) or if NODBALLOC is specified. Either OPEN or NOOPEN can be specified. OPTION(OPEN) is not valid for a HALDB master, but the partition structure rebuild is done if structure rebuild is needed and if only one HALDB master was specified in the command. No rebuild is attempted if there is more than one database name listed in the command.

The UPDATE DB AREA(*) START(ACCESS) OPTION(OPEN) command starts and opens all areas under one or more DEDBs specified in the NAME() keyword even if the areas are not registered on DBRC as PREOPEN. The OPTION(OPEN) is processed locally by all IMS systems that receive the command and is not maintained as a global status in RM.

PFA Sets or resets the "prevent further authorization" (PFA) or the "read-only" status in the RECON for the database. Use the PFA option with UPDATE DB START(ACCESS) to enable access to a

database. Use the PFA option with the UPDATE DB STOP (ACCESS|SCHD|UPDATES) command to prevent access to a database. PFA is optional. If you specify OPTION(PFA), the command master updates the RECON data set only if the command is successful at the master IMS. All other IMS systems that receive the command process is locally. OPTION(PFA) is not valid with SET(LOCK(ON|OFF)).

SCOPE()

Specifies where IMS should apply the change.

ALL

Changes are applied to the IMS systems that are currently active and to which the command is routed. Changes are also applied globally by updating the value maintained by RM. The global status that RM saves can be propagated across some IMS restarts. When an IMS system starts, it obtains global status information from RM. Maintaining global status in RM allows IMS systems to start without a specific setting or status for an area.

If the status or attribute that this command is updating is not maintained globally by RM, then the command is processed as if SCOPE(ACTIVE) is specified. If this command updates status or attributes that are both global and local, RM only updates the global status or attributes.

You must specify if the global area status must be maintained in RM. You can specify this during IMS initialization in either the DFSDFxxx or DFSCGxxx PROCLIB member with PLEXPARM(GSTSDB(Y)). You can also change it dynamically using the UPD IMS SET(PLEXPARM(GSTSDB(Y)) command. If you do not specify that global database status is to be maintained, the GLOBAL keyword is processed as in prior releases, and the global status is not updated.

Restriction: SCOPE(ALL) does not apply to the quiesce function. The scope of a quiesce is always all instances of the database usage across the IMSplex. There is not a global status of QUIESCE for a database. For START(QUIESCE) and STOP(QUIESCE), SCOPE(ALL) is the same as SCOPE(ACTIVE).

ACTIVE

Changes are applied to the active IMS systems to which the command is routed to. Any global status information maintained in the RM resource structure is not changed by the SCOPE(ACTIVE) command.

Restriction: SCOPE(ACTIVE) does not apply to the quiesce function. The scope of a quiesce is always all instances of the database usage across the IMSplex.

RM maintains global status information for some database attributes and values. IMS updates those values in RM based on commands or other internal events. When SCOPE(ALL) is specified, every IMS system that processes the command updates information in its local control blocks. Only the IMS command master calls RM to update the information globally.

The UPDATE DB command keywords that update information locally in the active IMS system and globally in RM when SCOPE(ALL) is specified include:

- START(ACCESS)
- START(ACCESS) SET(ACCTYPE())
- STOP(ACCESS|SCHD|UPDATES)
- SET(LOCK(ON|OFF))

If global status is maintained, UPD DB START(ACCESS) STOP(ACCESS,SCHD,UPDATES) SCOPE(ALL) writes an X'4C' log record to include global status and global time stamp.

SET() Specifies the attribute values to be changed or sets the database state.

ACCTYPE

Specifies the access intent for the named database. ACCTYPE() can only be specified if START(ACCESS) is also specified. This keyword overrides the database access intent specified at system definition.

If an UPDATE DB START(ACCESS) SET(ACCTYPE) command is issued for a HALDB master database, the access intent for the master database is set. The access intent settings for the partitions in the database are not affected. If an UPDATE DB START(ACCESS) SET(ACCTYPE) command is issued for one or more HALDB partitions, the access intent for the partitions is set. The access intent for the master database is not affected.

Because each partition has the access limitation of both itself and the master database, you might have to update the access intent of the partitions when you update the access intent of the master database. For example, if the master database and the partitions all have an access intent of read (READ) and you want to change the access intent for the master database and all of the partitions to update (UPD), issue the UPDATE DB START(ACCESS) SET(ACCTYPE) command for the master database and each of the partitions.

BRWS The database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those databases that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data since a sharing IMS subsystem could be updating the database. The database is opened for read-only processing.

EXCL The database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC.

READ The database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. Access type READ differs from access type BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UPD The database is for update as well as read processing in the IMS subsystem.

RESIDENT

> Specifies the resident option. The RESIDENT(N) option takes effect right away. The RESIDENT(Y) option takes effect at the next restart, unless the database was updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart. A database defined as a DEDB in ACBLIB always sets the RESIDENT(Y) attribute when the DEDB is loaded, regardless of the RESIDENT value specified. RESIDENT(N) is rejected for a DEDB.

> **N** The DMB associated with the named database resource is not made resident in storage. The DMB is loaded at scheduling time.

> **Y** The DMB associated with the named database resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the DMB and initializes it. A resident database is accessed in local storage, which eliminates I/O to the ACBLIB. In an online environment, the DMB control blocks are stored in the ACBLIB. If the DLI/SAS address space exists, DLI/SAS loads the DMB, otherwise, it is the IMS control region that loads it. This makes the DMB dependent on the existence of the corresponding database resource.

> The database must be stopped in order for this attribute to be changed. You may need to issue a /DBR DB command or an UPDATE DB STOP(ACCESS) command to stop the database before issuing an UPDATE DB SET(RESIDENT(Y|N)) command.

> An UPDATE command specified with SET(RESIDENT) is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined). The UPDATE DB command changes a MODBLKS database to dynamic, if the RESIDENT attribute is changed. These commands are recoverable.

LOCK Locks and unlocks the specified database. NAME(*) cannot be specified with SET(LOCK(ON|OFF)).

> **ON** Locks and prevents subsequently scheduled programs from accessing the database, without affecting currently scheduled programs. The database is not closed.

> For a shared secondary index database, an UPDATE DB SET(LOCK(ON)) on the first secondary index or subsequent secondary indexes affects only the named database.

> **OFF** Unlocks the specified databases and resets the effect of an UPDATE DB SET(LOCK(ON)) command. An UPDATE DB SET(LOCK(OFF)) on the first secondary index or subsequent secondary indexes affects only the named database.

TIMEOUT(*nnn*)

> Specifies the number of seconds to wait before a timeout occurs in a database quiesce. The timeout value can be 1 - 999 seconds. The TIMEOUT parameter value can override the DBQUIESCETO parameter in the DFSCGxxx member of the IMS PROCLIB data set. If the TIMEOUT parameter is omitted and the DBQUIESCETO

parameter is not specified, the default timeout value is 30 seconds. The TIMEOUT keyword is valid only with the START(QUIESCE) keyword.

START()

Specifies the attributes that are to be started.

ACCESS

The UPDATE DB START(ACCESS) command starts the database and permits access from transactions or programs. The UPDATE DB START(ACCESS) command resets the actions done by a prior UPDATE DB STOP(ACCESS), UPDATE DB STOP(SCHD), or UPDATE DB STOP(UPDATE). The access intent is set to the database access intent specified at system definition unless SET(ACCTYPE) is specified.

The UPDATE DB START(ACCESS) command can be used to allocate or reallocate all databases other than DEDBs. For a DEDB, an UPDATE AREA command can be used to allocate or reallocate the DEDB areas.

For a DEDB, the UPDATE DB START(ACCESS) command also causes any unloaded randomizer that was specified in the DBD source to be reloaded.

When the UPDATE DB START(ACCESS) command is specified for a HALDB partition, the partition is not allocated unless it has EEQEs, the OPEN keyword is specified, or it was previously authorized but not allocated. The action taken to allocate the data sets is dependent on the status of the master database and its availability.

If the UPDATE DB START(ACCESS) command is issued for a HALDB master database, access to the partitions is enabled if the partitions are not stopped or offline. The command updates the status of the master database. The command does not update the status of the partitions. If a partition is stopped or offline, it remains stopped or offline and must be explicitly started. If the ACCTYPE parameter is used, the access intent setting for the master database is set, but the access intent settings for the partitions in the database are not affected.

If the UPDATE DB START(ACCESS) command is issued for a HALDB partition, the command updates the status of the partition. The command does not update the status of the master database. If the ACCTYPE parameter is used, the access intent setting for the partition is set, but the access intent setting for the master database is not affected.

The UPDATE DB START(ACCESS) command may reset the USTOPPED status for transactions that are suspended. If the transaction is suspended and its processing program has access to the started database, the UPDATE DB START(ACCESS) command results in the USTOPPED attribute being reset. Any messages on the suspend queue for that transaction is transferred to the normal processing queue.

If there was a prior dynamic backout or emergency restart backout failure, then the UPDATE DB START(ACCESS) command attempts to perform the backout again.

If the database is registered to DBRC, then DBRC is informed when batch backout is successfully executed, and the failing backout is not attempted again when an UPDATE DB START(ACCESS) command is issued.

The UPDATE DB START(ACCESS) command is not processed for the databases being accessed by batch programs.

For a HIDAM database, the UPDATE DB START(ACCESS) command must be issued for both the index and the data area DBD. If a backout failure occurs for this database, the command causes the backout to be attempted again.

On an RSR tracker, the UPDATE DB START(ACCESS) command can be used to resume tracking for those databases that were stopped by a tracking subsystem processing. The command can also be used to start online forward recovery (OFR) for those databases that are not current with mainline tracking.

For shared secondary index databases, the UPDATE DB START(ACCESS) can be issued on the first secondary index or subsequent secondary indexes to undo the actions of the prior UPDATE DB STOP(SCHD) and UPDATE DB STOP(UPDATES) commands.

Additional functions can be performed with START(ACCESS) by specifying the OPTION keyword. OPTION(DBALLOC | NODBALLOC), OPTION(NOBACKOUT), OPTION(NOOPEN | OPEN) can be specified along with START(ACCESS). The OPTION keyword is not valid for a HALDB master.

QUIESCE
Specifies that the databases named on the command are to be quiesced to establish a new recovery point. The scope of a quiesce is always all instances of the database usage across the IMSplex. There is no quiesce that would apply only to a subset of the IMSplex.

STOP()
Specifies the attributes that are to be stopped.

ACCESS
Starts offline processing of the database. This processing closes and deallocates the database and deauthorizes the database to DBRC. An UPDATE DB START(ACCESS) command is required to reset the effect of an UPDATE DB STOP(ACCESS).

Specifying UPDATE DB STOP(ACCESS), UPDATE DB STOP(UPDATES), or UPDATE DB STOP(SCHD) can cause the randomizer routine or the selection partition routine to be deleted from memory.

OPTION(NOFEOV) is the default action for STOP(ACCESS). The IMS log is not switched to the next OLDS and a simple checkpoint is not taken. OPTION(FEOV) can be specified on the command to switch to the next OLDS and take an IMS simple checkpoint.

If the UPDATE DB STOP(ACCESS) command is issued for a HALDB master database, processing acts on the entire database. That is, IMS closes and deallocates all partition data sets and

unauthorizes all partitions in the database. The command updates the status of the master database. The command does not update the status of the partitions.

If the UPDATE DB STOP(ACCESS) command is issued for a partition, all data sets in the partition are closed and deallocated and the partition is deauthorized. The command also updates the status of the partition. The command does not update the status of the master database. Issue the UPDATE DB STOP(ACCESS) for a partition only if you are stopping access to a subset of the partitions in a HALDB database. If you are stopping access to the entire database, issue the command for the master database, not for the partitions.

The UPDATE DB STOP(ACCESS) command can be used on a database readiness level (DLT) tracker to take shadow areas and databases offline for image copy and recovery. The command can also be used to stop online forward recovery (OFR) in progress for the specified database.

The UPDATE DB STOP(ACCESS) command does not deallocate a data set if a VSAM data set hardware error occurred. For shared secondary index databases, an UPDATE DB STOP(ACCESS) on the first secondary index affects all databases sharing the secondary index data set. An UPDATE DB STOP(ACCESS) command on the subsequent secondary indexes affects only the named database.

QUIESCE
Specifies that the databases named on the command should be made available again by releasing the quiesce on the databases.

Unlike START(QUIESCE), where each database resource listed must be quiesced in order for the command to complete successfully, STOP(QUIESCE) continues to process each listed resource even if some resources are not in quiesced state or cannot be released from quiesced state. For those resources that are not in quiesced state or cannot be released from quiesced state, the command returns a response line for each of those resources.

SCHD Stops or prevents subsequently scheduled programs from accessing the database, without affecting currently scheduled programs. The database is not closed.

An UPDATE DB START(ACCESS) command can be used to reset the effect of an UPDATE DB STOP(SCHD) command.

If the command is issued for a DEDB or MSDB, programs using the database will not be scheduled. For other databases, the programs will still be scheduled. If the INIT call was issued, however, a call against the database will result in either a 3303 pseudoabend or a BA status code.

When the UPDATE DB STOP(SCHD) command is issued for a database that is in use by an MPP region, the command is processed after the region completes processing the current message. After the current message processing is complete, the application program receives a QC status indicating no more messages even if there are messages to be processed.

When the UPDATE DB STOP(SCHD) command is issued for a database that is in use by a BMP region, the command is rejected.

If the UPDATE DB STOP(SCHD) command is issued for a HALDB master database, access to all partitions in the database is stopped for subsequently scheduled programs. The command updates the status for the master database. The command does not update the status of the partitions.

If the UPDATE DB STOP(SCHD) command is issued for a HALDB partition, the status of the partition is updated. The command does not update the status of the master database. Use the UPDATE DB STOP(SCHD) command only for a partition or a subset of partitions in a HALDB database that you want to stop. To stop the entire database, issue the command for the master database, not for the partition.

In a DBCTL system, CCTL can specify LONG or SHORT when it schedules a PSB. When the UPDATE DB STOP(SCHD) command is issued for a database that is in use by a LONG thread, the command is rejected. When the command is issued for a database that is in use by a SHORT thread, the thread completes before the command is processed.

For a shared secondary index database, an UPDATE DB STOP(SCHD) on the first secondary index or subsequent secondary indexes affects only the named databases.

UPDATES

Stops or prevents transactions or programs from updating the specified DL/I database.

STOP(UPDATES) is not valid for DEDBs or MSDBs.

An UPDATE DB START(ACCESS) command is required to reset the effect of an UPDATE DB STOP(UPDATES) command.

When the UPDATE DB STOP(UPDATES) command is processed, the message processing regions using the specified database are terminated at the conclusion of processing their transactions in preparation to close the database and enable the databases to be opened input only. As the message processing regions terminate programs, the data sets of the database are closed.

OPTION(FEOV) forces the IMS log to switch to the next OLDS. This switch is marked as a recovery point for log archiving purposes. IMS also issues a simple checkpoint. OPTION(NOFEOV), which is the default, overrides this action.

After the command is processed, the scheduling of transactions is resumed. No transactions will be allowed to update the specified databases. Programs with update intent will be scheduled, but update calls to DL/I databases will result in a 3303 pseudoabend, a BA, or BB status code. The pseudoabend or status codes appear only if the application program informed IMS through the INIT STATUS GROUPA or GROUPB call that it is prepared to accept status codes regarding data unavailability.

An UPDATE DB STOP(UPDATES) on the first secondary index affects all databases sharing the secondary index data set. An UPDATE DB STOP(UPDATES) on subsequent secondary indexes affects only the named database.

If the UPDATE DB STOP(UPDATES) command is issued for a HALDB master database, all partition data sets are closed and reopened and all partitions in the database are reauthorized. The command also updates the status of the master database. It does not update the status of the partitions.

If the UPDATE DB STOP(UPDATES) command is issued for a partition, all data sets in the partition are closed and reopened and the partition is reauthorized. The command also updates the status of the partition. The command does not update the status of the master database. Issue the UPDATE DB STOP(UPDATES) for a partition only if you want to stop updates for a subset of the partitions in a HALDB database. To stop updates for the entire database, issue the command for the master database, not for the partition.

Usage notes

The UPDATE DB command can only be specified through the OM API and can only be processed by the DB/DC and DBCTL environments. When the UPDATE DB command is issued, it only applies to the IMS system to which it is routed. The command is not processed by other IMS systems in the IMSplex that share the database but do not receive the command. OM selects one IMS as the command master.

The command is not allowed on the XRF alternate system, RSR tracker, or FDBR region.

Resources exist for the life of IMS unless they are deleted using a DELETE command or online change for MODBLKS. Resource updates are recoverable across an IMS warm start or emergency restart. Updates to database runtime resource definition attributes such as RESIDENT are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

Each database is updated individually, unlike the online change process where either all databases are updated or no databases are updated. Some runtime resource definition values for a database can only be updated if the database is not in use. If the database is in use, the update fails. An exception to this rule is status. You can update the status of a database while it is in use. In a sysplex environment with multiple IMS systems, the update might succeed on some IMS systems and fail on others.

Runtime resource definition attributes include the following: RESIDENT.

An UPDATE DB command that changes runtime resource definition attributes such as the RESIDENT attribute is rejected for MSDBs and DEDBs.

The UPDATE DB command returns CCTXT with a nonzero completion code. The CCTXT can be up to 32 bytes, and it includes information about what the completion code means. The UPDATE DB SCOPE(ALL) command returns a response line with the completion code for the global status update. The CCTXT for that code is GBL CC.

If all the attributes specified by the UPDATE command are already defined for the resource, no update is actually made, no resources are quiesced, no log record is

created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

When you enter this command, the database name can be an existing non-HALDB, a HALDB master, or a HALDB partition. A command against a HALDB partition operates exactly like a command against a non-HALDB with the exception of the /START DATABASE and the UPDATE DB START(ACCESS) command. A HALDB partition is not allocated during the command unless it was previously authorized but not allocated, the OPEN keyword was specified, or the partition has EEQEs. The partition is allocated at first reference.

For HALDB databases, IMS tracks partition statuses and master database statuses separately. For example, a partition can be stopped, but its master database can be started. Alternatively, the partition can be started, but its master database can be stopped. Before opening, authorizing, or scheduling a partition, IMS always examines the status of the partition and the master database. If either the partition or the master database has a status that prevents the action, IMS does not perform the action.

Each partition has the access limitations of both itself and its master database. For example, if the master database has an access intent of read (READ) and one of its partitions has an access intent of update (UPD), the partition cannot be updated. Alternatively, if the master database has an access intent of update (UPD) and one of its partitions has an access intent of read (READ), the partition cannot be updated. Similar considerations apply to other statuses that affect access limitations, such as being stopped or locked.

Exception: If the HALDB master database has update access (UPD), the partitions can have an access intent of exclusive (EXCL), exceeding the access of the master.

Commands that are issued with a partition name affect only the status of the partition. Commands that are issued against the master database affect only the status of the master database. Therefore, a start of a master database does not update the status of its partitions. If the partitions are stopped, they remain stopped. When a HALDB partition is explicitly stopped, it must be explicitly started again. The type-1 commands with the keyword ALL, type-2 commands with NAME(*), and commands against a HALDB master do not change the STOPPED (shown as STOACC, STOSCHD, or STOUPDS on QUERY DB) and LOCKED indicators in each HALDB partition.

When the command target is a HALDB master, processing acts on all HALDB partitions. For example, if the IMS command is UPDATE DB STOP(ACCESS) on the HALDB master, all of the HALDB partitions are closed, deallocated, and deauthorized. However, the stopped status is only set in the master database. If a QUERY DB command is issued, only the HALDB master displays a status of STOACC (each HALDB partition does not display STOACC unless it was itself stopped). If a UPDATE DB STOP(ACCESS) command was issued against a HALDB master, the display output of a /DISPLAY DB command shows the HALDB master (as STOPPED), but does not display the status of the partitions.

If received during an UPD DB NAME(partname) START(ACCESS) command, where *partname* is a HALDB partition that was added and partition structure rebuild has not been done, then issue an UPD DB NAME(haldbmst) START(ACCESS) OPTION(OPEN), where *haldbmst* is the partition's master, followed by an UPD DB NAME(partname) START(ACCESS) command or issue a

call for a key in the key range of the new or redefined partition. This will invoke partition structure rebuild and allow the partition to be used.

Restrictions:
- The UPDATE DB START(ACCESS | QUIESCE) and UPDATE DB STOP(ACCESS | QUIESCE | SCHD | UPDATES) commands cannot be processed against a HALDB partition on an IMS system while HALDB Online Reorganization (OLR) is running against that partition on the same IMS system.
- The UPDATE DB START(ACCESS) SET(ACCTYPE(UPD)) and UPDATE DB STOP(ACCESS | UPDATES) commands cannot be issued against a HALDB master while OLR is reorganizing any of its partitions.
- While the database is being quiesced, this command cannot be processed successfully.
- If the /DBRECOVERY DB or UPDATE DB STOP(ACCESS) commands are issued for a DEDB database that has newly added areas using the DEDB Alter utility, you might need to start the DEDB database using the ACCESS option if your PSB has a PCB that is not accessing the newly added areas by their DEDB name. In this situation, issue one of the following commands:
  - /STA DB ACCESS=UP
  - UPDATE DB START(ACCESS) SET(ACCTYPE(UPD))

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE DB command and the type-1 IMS commands that perform similar functions.

Table 399. Type-1 equivalents for the UPDATE DB command

UPDATE DB command	Similar IMS type-1 command
UPDATE DB NAME(*name*) STOP(UPDATES)	/DBD DB *dbname*
UPDATE DB NAME(*name*) STOP(ACCESS)	/DBR DB *dbname*
UPDATE DB NAME(*name*) START(ACCESS)	/START DB *dbname*
UPDATE DB NAME(*name*) STOP(SCHD)	/STOP DB *dbname*
UPDATE DB NAME(*name*) SET(LOCK(ON))	/LOCK DB *dbname*
UPDATE DB NAME(*name*) SET(LOCK(OFF))	/UNLOCK DB *dbname*

Output fields

The following table shows the output fields for the UPDATE DB. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 400. UPDATE DB output fields

Short label	Keyword	Meaning
AREA	*error*	Area name of the DEDB that resulted in an error during the processing of the command.
CC	N/A	Completion code.
CCTXT	*error*	The completion code text that briefly explains the meaning of the completion code.
DB	DB	Database name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
GBL	GBL	Indicates that the response line is for the global update.
MBR	N/A	IMSplex member that built output line. IMS identifier of the IMS for which the database information is displayed. IMS identifier is always returned.

Return and reason codes

An IMS return and reason code is returned to OM by the UPDATE DB command. The OM return and reason codes that may be returned as a result of the UPDATE DB command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 401. Return and reason codes for the UPDATE DB command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000008'	X'00002004'	An invalid keyword or filter, or more than one keyword is specified on the UPDATE DB command.
X'00000008'	X'00002008'	Invalid number of keywords. Either a SET, START, or STOP keyword is required.
X'00000008'	X'00002011'	UPDATE DB command processing terminated because you cannot specify more than one keyword with the NAME(*) keyword.
X'00000008'	X'00002014'	The UPDATE DB command is not processed because an invalid character is found in the database name parameter.
X'00000008'	X'00002040'	More than one filter or keyword value is specified on the UPDATE DB command. Either more than one keyword or an invalid combination of filters was specified. For example, UPDATE DB NAME(dbname) START(ACCESS) OPTION(NORAND) or UPDATE DB NAME(dbname) STOP(UPDATES) OPTION(NORAND)
X'00000008'	X'00002048'	Invalid SET attribute.

Table 401. Return and reason codes for the UPDATE DB command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), output lines are only returned for resources with nonzero completion codes. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. The completion code indicates the reason for the error with the resource name. The completion codes that can be returned by the UPDATE DB command are listed in the UPDATE DB completion code table.
X'00000010'	X'00004000'	The UPDATE DB command is not processed as restart is in progress.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	The UPDATE DB command is not processed because Fast Path is not installed.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004124'	An UPDATE DB command that specified either START(QUIESCE) or STOP(QUIESCE) was rejected, because another process of the same type, such as another command with QUIESCE specified, was already in progress. Wait until the other process has completed.
X'00000010'	X'000041F0'	The UPDATE DB command is not processed because an IMS Checkpoint is in progress.
X'00000010'	X'000041F4'	The UPDATE DB command is not processed because an MSDB Checkpoint is in progress.
X'00000010'	X'000041F8'	The UPDATE DB command is not processed because a takeover is in progress.
X'00000010'	X'00004200'	Commands are not processed because IMS shutdown is in progress.
X'00000010'	X'00004208'	Invalid command for RMENV=NO.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004320'	The UPDATE DB command is not processed because another UPDATE DB START(QUIESCE) or UPDATE DB STOP(QUIESCE) command is in progress.
X'00000010'	X'00004325'	DB quiesce phase is in progress.
X'00000010'	X'00004400'	MINVERS in the RECON data sets is not 11.1.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.

Table 401. Return and reason codes for the UPDATE DB command (continued)

Return code	Reason code	Meaning
X'00000014'	X'00005000'	The UPD DB command processing terminated because IMODULE GETSTOR storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.
X'00000014'	X'00005014'	The UPDATE DB command processing terminated because BCB storage could not be obtained.
X'00000014'	X'0000501C'	The UPD DB command processing terminated because IMODULE GETMAIN storage could not be obtained.
X'00000014'	X'00005FFF'	The UPDATE DB command processing terminated because of an internal error.

Completion codes

The following table includes an explanation of the completion codes. Errors unique to the processing of UPDATE DB command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 402. Completion codes for the UPDATE DB command

Completion code	Completion code text	Meaning
0		The command completed successfully for the resource.
8	COMMAND COMPLETE FOR SOME	Some. The command completed with error for some of the AREAS of the DEDB. Response lines for the area names in error are returned.
C	COMMAND COMPLETE FOR NONE	None. The command completed with error for all the AREAs of the DEDB. Response lines for the area names in error are returned. This error is returned when all the databases could not be quiesced.
10	NO RESOURCES FOUND	Database name is invalid, or the wildcard parameter specified does not match any database names.
17	ANOTHER CMD IN PROGRESS	This error is returned when the quiesce could not be started because another database command was in progress.
23	DB STOP ACCESS IN PROGRESS	A /DBRECOVERY, or UPDATE DB STOP(ACCESS) command to stop database access is in progress. This takes the database offline.
25	DB STOP UPDATES IN PROGRESS	A /DBDUMP or UPDATE DB STOP(UPDATES) command to stop database updates is in progress.

Table 402. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
26	DEDB STOP IN PROGRESS	/DBRECOVERY, /STOP, or UPDATE DB STOP(SCHD) command to stop database scheduling is in progress for a DEDB.
31	NOT ALLOWED FOR A DEDB	Database is a DEDB. The command entered is not valid for the DEDB in the IMS environment.
32	NOT ALLOWED FOR AN MSDB	Database is an MSDB. The command entered is not valid for the MSDB in the IMS environment.
33	NOT ALLOWED FOR A HALDB MASTER	Command invalid HALDB master. The command OPTION is invalid for the HALDB master but partition structure rebuild will be done if structure rebuild is needed and if only one HALDB master was specified in the command. No rebuild will be attempted if there is more than one database name listed in the command.

If there are multiple database names listed in the command and all are invalid except the HALDB master, then rebuild will be attempted if needed. |
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for an IMS descriptor or resource. DFSDSDB1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
53	NO RM ADDRESS SPACE	This error is returned when the command could not be processed because the RM address space is not present.
55	NO FAST PATH INSTALLED	No Fast Path installed. The command failed because Fast Path is not installed.
56	FF DB + LSO=Y + TRK = ERROR	Command is invalid on the RSR tracker because of the LSO=Y option.
65	DMB POOL STORAGE ERROR	DMB pool storage error. The command failed because of DMB pool storage request failure.
66	DMB POOL FULL	DMB pool full. The command failed because the DMB pool was full.
6C	NOT ALLOWED FOR A HALDB PARTITION	An UPDATE command specified a change to the residency option for a HALDB partition. The residency option is valid only for the master and not the partitions.

Table 402. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
6F	REFERENCED BY PROGRAM	An UPDATE DB command is issued to change the resident option. There is a currently scheduled program that is referencing that database. The UPDATE command fails.
76	RECOVER CMD ACTIVE	/RECOVER START command is in progress to recover one or more databases with the database recovery services.
81	DBRC ERROR	DBRC error.
8C	RESOURCE IS STOPPED	The database is stopped. The database might be stopped by a command, such as UPDATE DB, /STOP DB, or /DBRECOVERY, or for some other reason, such as an error.
90	INTERNAL ERROR	Internal error. The command entered is not processed because of an internal error.
91	TIMEOUT ERROR	This error is returned when the quiesce could not be completed within the timeout period.
92	COMMAND PROCESSING ERROR	Command processing error. The command entered is not processed because of an error. A unique completion code could not be generated to explain the error. The message number and the return code that could not be converted to a completion code are listed in the error text.
A0	DYNAMIC ALLOCATION FAILED	Dynamic allocation failed. The command entered has not completed processing because the dynamic allocation failed for the DB. Some of the command processing might be completed before the error is detected.
A1	DB IS AUTHORIZED BY BATCH	Database is authorized by batch. The command entered has not completed processing because the database is authorized by batch. Some of the command processing might be completed before the error is detected.

Table 402. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
A2	DB IS AUTHORIZED BY ANOTHER IMS	Database is authorized by another IMS. The command entered has not completed processing because the database is authorized by another active or abnormally terminated IMS and its authorization state is incompatible with the current authorization request.
A3	AUTHORIZATION CHANGE FAILED	Authorization change failed. The DBRC CHNGAUTH request resulted in an error.
A4	DATABASE NOT REGISTERED TO DBRC	Database not registered to DBRC. The command processing has not completed because the database is not registered to DBRC. Some of the command processing might be completed before the error is detected.
A5	PREVENT FURTHER AUTH ON	Prevent further authorization ON. The command entered has not completed processing because the database or area is defined to DBRC as "prevent further authorization". Some of the command processing might be completed before the error is detected.
A6	INVALID DATABASE RECORD IN RECON	Invalid database record in the RECON data set. The command entered has not completed processing because an invalid parameter was found during the evaluation process of the database usage compatibility. The database record might be invalid in the RECON data set. Some of the command processing might be completed before the error is detected.
A7	DBRC UNAUTH FAILED FOR CHNGAUTH	DBRC unauthorization failed during change authorization. The command entered has not completed processing because of an error during UNAUTH processing for the change authorization request.
A8	INVALID DB RECORD IN RECON	Invalid database record in RECON. An UPDATE DB SET(ACCTYPE) command is entered to change the database authorization level. An encoded state of zero is returned by DBRC during the change authorization processing. Some of the command processing might be completed before the error is detected.

Table 402. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
A9	DB OR AREA AUTHORIZATION ERROR	Database or area authorization error. For a database, the command entered has not completed processing because of a database authorization error. For an area, area authorization to DBRC failed. Some of the command processing might be completed before the error is detected.
AA	DB IN USE-BMP	The UPDATE DB command was not successful because the database is in use by a BMP. Some of the command processing might be completed before the error is detected.
AB	DB IN USE-DBCTL LONG THREAD	The UPDATE DB command was not successful because the database is in use by a long-running DBCTL thread. Some of the command processing might be completed before the error is detected.
AC	FP AREA HELD-LONG BUSY WAIT	The UPDATE DB command was not successful because the AREA of the DEDB is in long-busy wait. Some of the command processing might be completed before the error is detected.
AD	DYNAMIC UNALLOCATION FAILED	The UPDATE DB command was not successful due to a dynamic unallocation error. Some of the command processing might be completed before the error is detected.
AE	DYNAMIC ALLOCATION ERROR	The UPDATE DB command was not successful due to a dynamic allocation error. No SVC99 is issued. Some of the command processing might be completed before the error is detected.
C1	OLR DDIR MISSING OR DFSPNT ZERO	Unknown DMB referenced for database. The command cannot be processed because an unknown data management block is referenced for the database. Refer to the DFS564I message put out to the system console to identify the DMB name that cannot be referenced.
CC	OLR IS ACTIVE FOR DATABASE	OLR is active for database. The command failed as OLR is active for the database.
D0	DATABASE CLOSE ERROR	Database close error. The command processing failed because of a database close error.

Table 402. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
D1	DATABASE WRITE ERROR	Database write error. The command processing failed because of a database write error.
D2	DATABASE NEEDS BACKOUT	Database needs backout. The command processing failed as the database needs backout.
D3	DATABASE OR AREA NEEDS RECOVERY	Database or AREA needs recovery. The command processing failed as the database or area needs recovery.
D4	DATABASE NEEDS IMAGE COPY	Database needs image copy. The command processing failed as the database needs image copy.
D5	DATABASE HAS NO BACKOUTS	Database has no backouts. The command processing failed as there are no backouts for the database.
D6	DATABASE IN USE	Database in use. A SET(ACCTYPE) is specified for the DEDB and the authorization level cannot be changed as the DEDB is in use in a region.
D7	DB I/O PREVENTION NOT COMPLETE	Database I/O prevention not complete. The database cannot be started as it is extended because of an XRF takeover and the I/O prevention is not complete.
D8	DATABASE BACKOUTS PENDING	Database backouts pending. The access type specified for the database cannot be changed as restartable backouts are pending for the database.
D9	DATABASE/AREA OPEN FAILED	Database open failed. The command failed because of an error opening the database. Refer to DFS0730I messages to determine the reason of the failure.
DA	DATABASE BEING RECALLED BY HSM	Database being recalled from HSM. The command processing failed because the database is being recalled from HSM.

Table 402. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
DB	PARTITION OPEN FAILED	Partition open failed. The partitions open failed because the master is offline. This can also occur if the partition has been deleted and partition structure rebuild has occurred. Partition structure rebuild can be accomplished by issuing an UPD DB NAME(haldbmst) START(ACCESS) OPTION(OPEN) command, where haldbmst is the partition's master, or by issuing a qualified GU call for a key in the key range of the partition. List.recon can be used to determine if the partition exists or has been deleted.
DC	HALDB PARTITION BUILD FAILURE	Database partition build failure. The database partitions build for the DDIR or DMB failed. Refer to the DFS0415I message sent to the system console to determine the reason of the failure.
DD	HALDB PARTITION INIT FAILURE	Database partition initialization failed. The database partition initialization for the DDIR or DMB failed. Refer to the DFS0415 message sent to the system console for the details.
DE	ACBLIB READ FAILURE	ACBLIB read failure. The command is not processed because there was an error reading the ACBLIB.
DF	DB DIRECTORY INIT FAILURE	Database directory initialization failed. The command is not processed because of a database directory initialization failure.
E0	DATABASE OR AREA IN RECOVERY	Database or area in recovery. The command is not processed because the database or area is in recovery.
E1	DB MUST BE STOPPED AND OFFLINE	The database must be stopped and taken offline in order for the attribute to be changed. You might need to issue a /DBR DB command or an UPDATE DB STOP(ACCESS) command to stop the database and take it offline before issuing the UPDATE DB SET(RESIDENT(Y|N)) command.
E2	PARALLEL DB OPEN NOT COMPLETE	Restart parallel DB open not complete. The command is not processed because the restart parallel DB open is not complete for the database.

Table 402. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
E5	PARTICIPANT UNABLE TO QUIESCE	This error is returned when the quiesce could not be completed successfully across the IMSplex. The IMS with this completion code was the quiesce participant that was not able to be quiesced.
E6	QUIESCE COMMUNICATION FAILURE	This error is returned when the quiesce could not be completed because of a failure to communicate across the IMSplex. There could be a problem with RM, OM, or SCI that has caused the communication failure to occur.
E7	CMD NOT ALLOWED	This error is returned when the database command could not be processed because a quiesce command was in progress.
E8	DATABASE HAS INTENT TO REORGANIZE	This error is returned when the quiesce could not be started because the RECON data sets indicate that there is an intent to reorganize the database.
E9	DB IN WRONG STATE TO BE QUIESCED	This error is returned when the named resource is in the wrong state for quiesce processing to proceed.
EE	DATABASE BACKOUT ERROR	Database backout error. The command processing failed because of a database backout error.
EF	DATABASE IS IN ERROR	Database is in error. The command entered is not processed because the database is in error.
F0	NO AREA LOCK	No AREA lock. The command processing failed as the area lock could not be obtained.
F1	AREA NOT STOPPED	Area not stopped. The command entered is not processed because the AREA is not stopped.
F2	PRELOAD IS ACTIVE FOR AREA	Preload is active for AREA. The command entered is not processed because the AREA is not stopped.
F3	UNRESOLVED INDOUBTS FOR AREA	Unresolved indoubts for AREA. The command entered is not processed because unresolved indoubts exist for the AREA.
F4	ALLOCATION FAILED	Allocation failed. Allocation failed for the AREA name.
F5	AREA NEEDS RECOVERY	AREA needs recovery. The command processing failed because the area needs recovery.

Table 402. Completion codes for the UPDATE DB command (continued)

Completion code	Completion code text	Meaning
F6	ADS NUMBER DISCREPANCY	ADS number discrepancy. The command failed for the AREA because there is a discrepancy between the number of ADS allocated by IMS and the number of ADS known to DBRC. Correct the discrepancy and reissue the command to deallocate the data sets.
F7	AREA IS NOT LOADED INTO CF	Area is not loaded into CF. The AREA OPEN failed and is not loaded into the Coupling Facility.
F8	AREA HAS I/O TOLERATED CI	Area has I/O tolerated CI. The command is not processed because the AREA has an I/O tolerated CI.
F9	AREA HAS 2ND CI EEQE	Area has 2nd CI EEQE. The command is not processed because the AREA has a 2nd CI EEQE.
FC	UTILITY ACTIVE ON AREA	This error is returned when the area is in use by a utility.
FD	AREA HAS EEQE	This error is returned when the area has an extended error queue element (EEQE).
FE	AREA HAS EQE	This error is returned when the area has an error queue element (EQE).
196	DB has an in-doubt EEQE	This error is returned when an UPDATE DB STOP(ACCESS) command is issued against a database that has an in-doubt EEQE and the NODBR keyword was specified in the DFSVSMxx member of the IMS PROCLIB data set.
1D7	IMPORT CHANGE LIST ERROR	The resource is in the process of being imported from the change list in the IMSRSC repository or was not successfully imported from the change list. The resource cannot be updated until it is successfully imported from the repository.

Examples

The following are examples of the UPDATE DB command:

Example 1 for UPDATE DB command

TSO SPOC input:
```
UPDATE DB NAME(DEDBJ00%,BADNAME,BAD*) SET(RESIDENT(Y))
```

TSO SPOC output:

```
Response for: UPDATE DB NAME(DEDBJ00%,BADNAME,BAD*) SET(RESIDENT(Y))
DBName    MbrName    CC CCText
BAD*      IMS1       10 NO RESOURCES FOUND
BADNAME   IMS1       10 NO RESOURCES FOUND
DEDBJ001  IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ002  IMS1        0
DEDBJ003  IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ004  IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ005  IMS1        0
DEDBJ006  IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ007  IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ008  IMS1       E1 DATABASE HAS NOT BEEN DBR"D
DEDBJ009  IMS1       E1 DATABASE HAS NOT BEEN DBR"D
```

OM API input:

```
CMD(UPDATE DB NAME(DEDBJ00%,BADNAME,BAD*) SET(RESIDENT(Y)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2006.311 00:34:47.028100</statime>
<stotime>2006.311 00:34:47.031559</stotime>
<staseq>BFAADA74E4584E87</staseq>
<stoseq>BFAADA74E5307115</stoseq>
<rqsttkn1>USRT011 10163446</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC     </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1     </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>DB              </kwd>
<input>UPDATE DB NAME(DEDBJ00%,BADNAME,BAD*) SET(RESIDENT(Y)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DB" llbl="DBName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="AREA" llbl="AreaName" scope="LCL" sort="a" key="4"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="GBL" llbl="Global" scope="GBL" sort="d" key="2" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="y" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
```

```
<rsp>DB(DEDBJ001) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(BADNAME ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>DB(BAD*    ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp
<rsp>DB(DEDBJ002) MBR(IMS1) CC(   0) </rsp>
<rsp>DB(DEDBJ003) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ004) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ005) MBR(IMS1) CC(   0) </rsp>
<rsp>DB(DEDBJ006) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ007) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ008) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
<rsp>DB(DEDBJ009) MBR(IMS1) CC(  E1) CCTXT(DATABASE HAS NOT BEEN DBR"D)
 </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update some databases to be resident. The update succeeds for databases DEDBJ002 and DEDBJ005, as shown by completion code 0. The update fails for several databases with completion code E1, because the databases need to be taken offline by use of the /DBRECOVERY command in order to change a runtime resource definition attribute. The update fails for database BADNAME and for parameter BAD* with completion code 10, since database BADNAME does not exist and no database name starts with BAD.

Example 2 for UPDATE DB command

TSO SPOC input:
```
UPDATE DB NAME(DEDBJN22) AREA(*) START(ACCESS) OPTION(OPEN)
```

TSO SPOC output:
```
DBName    MbrName     CC    CCtext
DEDBJN22  IMS1         C
DB22AR0   IMS1         0
DB22AR1   IMS1         A9   DB OR AREA AUTHORIZATION ERROR
```

Explanation: In this example, CC=A9 means that DBRC returned a nonzero return code for authorization request.

Example 3 for UPDATE DB command

TSO SPOC input:
```
UPD DB NAME(DEDBJN22) START(QUIESCE) OPTION(HOLD) SET(TIMEOUT(60))
```

TSO SPOC output:
```
 DBName    MbrName  CC
 DEDBJN22  IMS1      0
 DEDBJN22  IMS2      0
 DEDBJN22  IMS3      0
```

Explanation: This example is of a successful quiesce and hold for Fast Path DEDB DEDBJN22, which is coordinated across three IMS systems.

Example 4 for UPDATE DB command

TSO SPOC input:

```
UPD DB NAME(DEDBJN22) STOP(QUIESCE)
```

TSO SPOC output:

```
DBName    AreaName MbrName  CC CCText
DEDBJN22           IMS1      C COMMAND COMPLETE FOR NONE
DEDBJN22 DB22AR0  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR1  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR2  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR3  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR4  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
```

Explanation: This example attempts to release a quiesce on a DEDB. DEDB
DEDBJN22 consists of five areas. The command failed because none of the areas
were held in quiesced state. A response line is returned for the DEDB name with
CC=C ("COMMAND COMPLETE FOR NONE") because none of the areas under
the DEDB were in a state to be released. A response line is also returned for each
area.

Example 5 for UPDATE DB command

TSO SPOC input:

```
UPD DB NAME(DEDBJN22) STOP(QUIESCE)
```

TSO SPOC output:

```
DBName    AreaName MbrName  CC CCText
DEDBJN22           IMS1      8 COMMAND COMPLETE FOR SOME
DEDBJN22 DB22AR0  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR3  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
DEDBJN22 DB22AR4  IMS1     E9 DB IN WRONG STATE TO BE QUIESCED
```

Explanation: This example attempts to release a quiesce on a DEDB. DEDB
DEDBJN22 consists of five areas. The command was partially successful because it
was able to release the quiesce on DB22AR1 and DB22AR2 only. In this example,
DB22AR0, DB22AR3, and DB22AR4 were not in quiesced state. A response line is
returned for the DEDB name with CC=8 ("COMMAND COMPLETE FOR SOME")
because only some of the areas under the DEDB were in a state to be released. A
response line is also returned for each area that could not be processed.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming
APIs)

Related tasks:

➡ Updating runtime database resource and descriptor definitions with the
UPDATE command (System Definition)

Related reference:

➡ /LOCK command (Commands)

➡ Command keywords and their synonyms (Commands)

UPDATE DBDESC command

Use the UPDATE DBDESC command to update database descriptors. A descriptor is a model that can be used to create descriptors or resources.

Updating a descriptor changes only the attributes explicitly specified on the UPDATE command. Attributes not specified retain their existing values. Any database resource or descriptor can be created using this descriptor as a model, by specifying the CREATE command with LIKE(DESC(*descriptor_name*)). Any descriptor or resource that was already created using this descriptor is not updated.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1068
- "Usage notes" on page 1069
- "Output fields" on page 1070
- "Return, reason, and completion codes" on page 1070
- "Examples" on page 1072

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 403. Valid environments for the UPDATE DBDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE DBDESC	X	X	
NAME	X	X	
OPTION	X	X	
SET	X	X	

Syntax

A:

B:

```
               ┌─,──────────────────────────┐
├──SET(──▼──┬──RESIDENT(──┬─N─┬─)──┬──)──────────────────────────┤
           │              └─Y─┘    │
           └──DEFAULT(Y)───────────┘
```
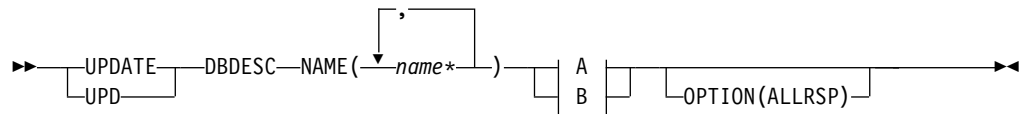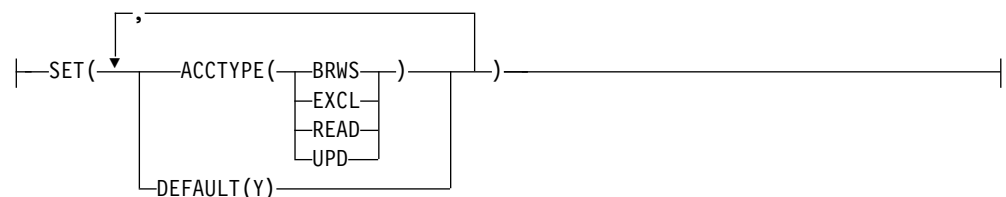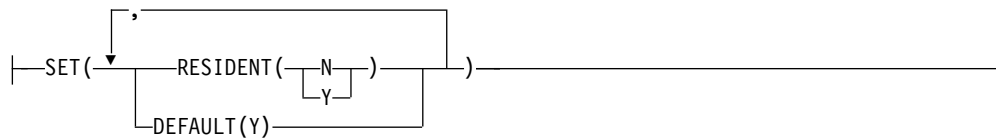
Keywords

The following keywords are valid for the UPDATE DBDESC command:

NAME

Specifies the 1-8 character name of the database descriptor name. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the descriptor names that are processed. For NAME(*) command responses are returned only for the descriptor names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the descriptor names that are processed.

OPTION

Specifies additional functions to be performed along with the command.

ALLRSP

Indicates that the response lines are to be returned for all descriptors that are processed on the command. The default action is to return response lines only for the descriptors that resulted in an error. It is only valid with names that include a wildcard character. ALLRSP is ignored for other NAME values.

SET

Specifies the attribute values to be changed.

ACCTYPE

Specifies the access intent for a database created using this descriptor.

BRWS

The database is available for read-only processing on this IMS subsystem. The only programs that can use the database on this subsystem are those that have a PCB processing option of GO (PROCOPT=GO). Programs that access the data using the GO processing option might see uncommitted data since a sharing IMS subsystem could be updating the database. The database is opened for read-only processing.

EXCL

The database is to be used exclusively by this IMS subsystem. This exclusive access is guaranteed only when the database is registered to DBRC.

READ

The database is available for read-only processing in this IMS subsystem. Programs with update intent can be scheduled, but cannot update the database. Access type READ differs from access type BRWS in that the data is read with integrity (locking is performed) and all programs can access the data, not just those with a processing option of GO. The database is opened for read-only processing.

UPD

> The database is for update as well as read processing in the IMS subsystem.

DEFAULT(Y)

> Specifies this descriptor as the default, which resets the existing default descriptor to DEFAULT(N). When a descriptor is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a database descriptor called DFSDSDB1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor. Since only one database descriptor can be the default at one time, only one database name can be specified with DEFAULT(Y).

RESIDENT

> Specifies the resident option of a database created using this descriptor. The RESIDENT(Y) option takes effect at the next restart, unless the database was updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart. A database defined as a DEDB in ACBLIB always sets the RESIDENT(Y) attribute when the DEDB is loaded, regardless of the RESIDENT value specified. RESIDENT(N) is rejected for a DEDB.

> **N** The DMB associated with the named database resource is not made resident in storage. The DMB is loaded at scheduling time.

> **Y** The DMB associated with the named database resource is made resident in storage at the next IMS restart. At the next IMS restart, IMS loads the DMB and initializes it. A resident database is accessed from local storage, which eliminates I/O to the ACBLIB. In an online environment, the DMB control blocks are stored in the ACBLIB. If the DLI/SAS address space exists, DLI/SAS loads the DMB, otherwise, it is the IMS control region that loads it. This makes the DMB dependent on the existence of the corresponding database resource.

Usage notes

Descriptors exist for the life of the IMS unless they are deleted using a DELETE command. Descriptors are recoverable across an IMS warm start or emergency restart. Descriptors are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

If all the attributes specified by the UPDATE command are already defined for the descriptor, no update is actually made, no descriptors are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

The UPDATE DBDESC command can be issued only through the OM API. This command applies to DB/DC and DBCTL systems.

The UPDATE DBDESC commands are not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE DBDESC commands are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

This command is recoverable.

Each descriptor is updated individually. Individual updating does not work like online change where either all descriptors are updated or no descriptors are updated. Descriptors can be successfully updated if they are not currently in use. If a descriptor is in use, the update fails. In a sysplex environment, the update might succeed on some IMS systems and fail on others. A descriptor is in use if another command is in progress that references the command.

If the descriptor is the IMS-defined database descriptor (DFSDSDB1), the only attribute that can be updated is the DEFAULT attribute.

Output fields

The following table shows the UPDATE DBDESC output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
> Provides a brief description of the output field.

Table 404. Output fields for the UPDATE DBDESC command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the non-zero completion code.
DESC	DBDESC	Database descriptor name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a non-zero completion code and contains information that further explains the completion code.
OLDDEF	DBDESC	Old default descriptor name, if this descriptor is updated to be the default by specifying DEFAULT(Y). The old default descriptor is no longer the default.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 405. Return and reason codes for the UPDATE DBDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.

Table 405. Return and reason codes for the UPDATE DBDESC command (continued)

Return code	Reason code	Meaning
X'00000004'	X'00002008'	Invalid number of keywords. Either a SET keyword is required.
X'00000008'	X'00002048'	Invalid SET attribute.
X'00000008'	X'00002133'	Multiple name parameters were specified with DEFAULT(Y). Only one descriptor can be the default at a time.
X'0000000C'	X'00003000'	Command was successful for some descriptors but failed for others. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), output lines are only returned for descriptors with non-zero completion codes. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004310'	Storage could not be obtained for the Transaction Input edit routine table. A cold start is required to fix this error.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 406. Completion codes for the UPDATE DBDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for database resource or database resource descriptor.
17	ANOTHER CMD IN PROGRESS	None. The command completed with error for all the Areas of the DEDB. Response lines for the area names in error are returned.
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for an IMS descriptor or resource. DFSDSDB1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).

Table 406. Completion codes for the UPDATE DBDESC command (continued)

Completion code	Completion code text	Meaning
6C	NOT ALLOWED FOR A HALDB PARTITION	The UPDATE command specified a change to the residency option for a HALDB partition. The residency option is valid only for the master and not the partitions.
8A	WILDCARD PARAMETER NOT SUPPORTED	A wildcard parameter was specified with DEFAULT(Y). Only one descriptor can be the default at a time.
1D7	IMPORT CHANGE LIST ERROR	The descriptor is in the process of being imported from the change list in the IMSRSC repository or was not successfully imported from the change list. The descriptor cannot be updated until it is successfully imported from the repository.

Examples

The following are examples of the UPDATE DBDESC command:

Example 1 for UPDATE DBDESC command

TSO SPOC input:
```
UPDATE DBDESC NAME(*) SET(RESIDENT(Y)) OPTION(ALLRSP)
```

TSO SPOC output:
```
Response for: UPDATE DBDESC NAME(*) SET(RESIDENT(Y)) OPTION(ALLRSP)
DescName MbrName    CC CCText
BRWSDESC IMS1        0
DESC001  IMS1        0
DESC002  IMS1        0
DESC003  IMS1        0
DESC004  IMS1        0
DESC005  IMS1        0
DFSDSDB1 IMS1       48 NOT ALLOWED FOR IMS RESOURCE
EXCLDESC IMS1        0
RESDESC  IMS1        0
```

OM API input:
```
CMD(UPDATE DBDESC NAME(*) SET(RESIDENT(Y)) OPTION(ALLRSP))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.312 23:27:37.656606</statime>
<stotime>2006.312 23:27:37.661922</stotime>
<staseq>BFAD4F2D1B11EC8E</staseq>
<stoseq>BFAD4F2D1C5E294E</stoseq>
<rqsttkn1>USRT011 10152737</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
```

```
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1     ">
<typ>IMS     </typ>
<styp>DBDC     </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1     </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>DBDESC          </kwd>
<input>UPDATE DBDESC NAME(*) SET(RESIDENT(Y)) OPTION(ALLRSP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1" scroll=
 "no" len="8" dtype="CHAR" align="left" skipb="no"  />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(EXCLDESC) MBR(IMS1) CC(    0) </rsp>
<rsp>DESC(DESC004 ) MBR(IMS1) CC(    0) </rsp>
<rsp>DESC(DESC005 ) MBR(IMS1) CC(    0) </rsp>
<rsp>DESC(DESC001 ) MBR(IMS1) CC(    0) </rsp>
<rsp>DESC(RESDESC ) MBR(IMS1) CC(    0) </rsp>
<rsp>DESC(BRWSDESC) MBR(IMS1) CC(    0) </rsp>
<rsp>DESC(DESC002 ) MBR(IMS1) CC(    0) </rsp>
<rsp>DESC(DESC003 ) MBR(IMS1) CC(    0) </rsp>
<rsp>DESC(DFSDSDB1) MBR(IMS1) CC(   48) CCTXT(NOT ALLOWED FOR IMS
RESOURCE) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update all database descriptors to be resident. The update completed successfully for most database descriptors, as shown by completion code 0. The update failed for IMS-defined descriptor DFSDSDB1 with completion code 48 NOT ALLOWED FOR IMS RESOURCE, since the only attribute that can be updated for DFSDSDB1 is DEFAULT(Y).

Related concepts:

➷ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➷ Command keywords and their synonyms (Commands)

UPDATE IMS command

Use the UPDATE IMS command to update local and global IMS attributes, which include global status for areas, databases, and transactions in an IMSplex. You can also use the UPDATE IMS command to enable IMS to use the IMSRSC repository.

To change global PLEXPARM values in an IMSplex, use the UPDATE IMS
SET(PLEXPARM()) command. To change local LCLPARM values in one or more
IMS systems in an IMSplex, use the UPDATE IMS SET(LCLPARM()) command.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1075
- "Usage notes" on page 1078
- "Output fields" on page 1079
- "Return, reason, and completion codes" on page 1080
- "Examples" on page 1083

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from
which the UPDATE IMS command and keywords can be issued.

Table 407. Valid environments for the UPDATE IMS command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE IMS	X	X	X
SET	X	X	X
OPTION	X	X	X

Syntax

```
>>─┬─UPDATE─┬─IMS───────────────────────────────────────────────────────>
   └─UPD────┘

>─┬─SET(LCLPARM(─┬─AUTOEXPORT(N)──────────────────┬─))─┬──────────────────><
  │              ├─FPBP64STAT(─┬─N─┬─)─────────────┤    │
  │              │             └─Y─┘               │    │
  │              │             ┌──────,──────────┐ │    │
  │              │             │                 │ │    │
  │              ├─LOCKTIME(─▼─┬─BMP(xx)────────┬─┴─)─┤  │
  │              │            ├─BMPOPT(─┬─ABEND──┬─)─┤   │
  │              │            │         └─STATUS─┘    │   │
  │              │            ├─MSG(yy)──────────────┤    │
  │              │            └─MSGOPT(─┬─ABEND──┬─)──┘    │
  │              │                      └─STATUS─┘         │
  │              └─REPO(Y)─REPOTYPE(IMSRSC)───────────────┘
  │              ┌──────,──────────┐
  │              │                 │
  ├─SET(PLEXPARM(─▼─┬─GSTSAREA(─┬─N─┬─)─┬─┴─))─────────────
  │                │          └─Y─┘  │
  │                ├─GSTSDB(─┬─N─┬─)──┤
  │                │         └─Y─┘    │
  │                └─GSTSTRAN(─┬─N─┬─)┘
  │                           └─Y─┘
  └─SET(PLEXPARM())─OPTION(─┬─CONTINUE─┬─)─
                           └─FORCE────┘
```
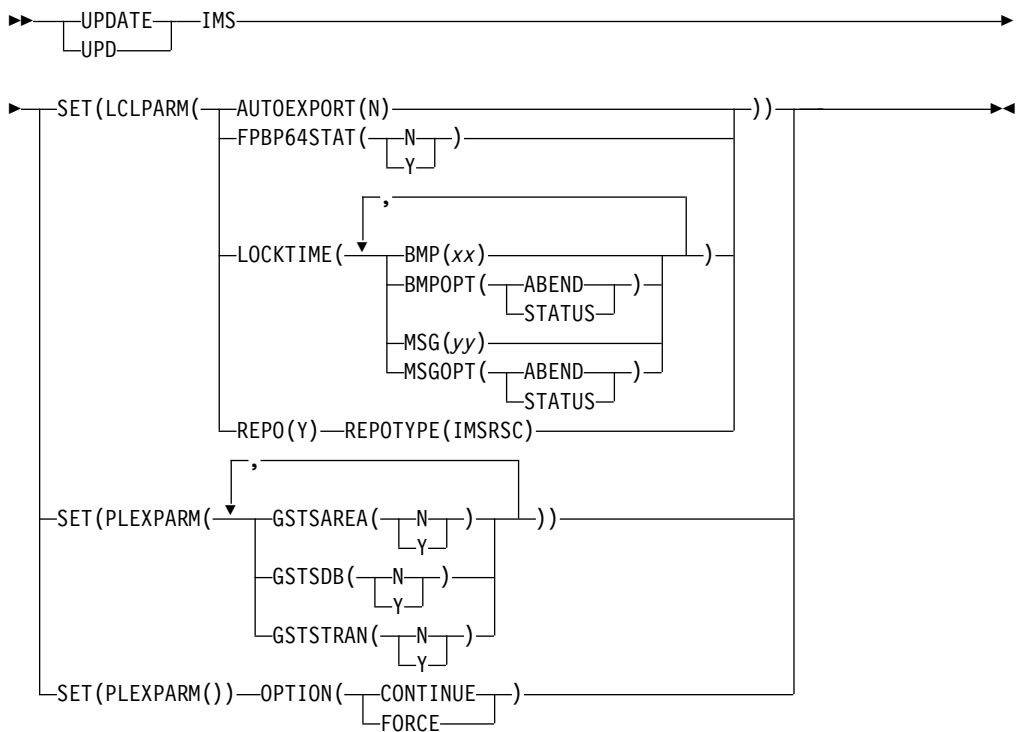
Keywords

The following keywords are valid for the UPDATE IMS command:

LCLPARM()

Specifies the values for the local parameters to be modified. The LCLPARM() keyword is mutually exclusive with the PLEXPARM() keyword in the UPDATE IMS SET() command.

The UPDATE IMS SET(LCLPARM()) command can be used to modify local parameters such as FPBP64STAT at each IMS system.

For the UPDATE IMS SET(LCLPARM()) command, all IMS systems that received the command from OM process the command.

Specify one of the following options:

AUTOEXPORT(N)

Specifies that the automatic export processing during system checkpoint should be disabled. The AUTOEXPORT keyword is valid in a DRD environment with or without the IMSRSC repository.

For the DRD users who are enabled with RDDS data sets, the UPDATE IMS SET(LCLPARM(AUTOEXPORT(N))) command can be used after successful migration to the repository. This command reduces the processor usage of automatic export at system checkpoints.

AUTOEXPORT(N) turns off automatic export to both the IMSRSC repository and RDDS if both are enabled. If there is an error during automatic export to the repository during IMS shutdown, you can use the UPDATE IMS SET(AUTOEXPORT(N)) command to turn off automatic export to the repository and try the IMS shutdown again.

Changes made by the UPDATE IMS SET(AUTOEXPORT(N)) command only remain in effect until IMS restarts. To permanently change the AUTOEXPORT settings in IMS, modify the AUTOEXPORT setting in the DFSDFxxx PROCLIB member.

After the UPDATE IMS command with the AUTOEXPORT(N) keyword is issued, you can continue using system RDDS data sets on the EXPORT and IMPORT commands and during automatic import. However, during import and automatic import, make sure that the RDDS contains the current data especially if AUTOEXPORT is turned off.

Automatic export to the RDDS, the IMSRSC repository, or both, can be enabled by a cold start of IMS with AUTOEXPORT=RDDS | REPO | AUTO specified for the DFSDFxxx PROCLIB member.

Automatic export to the RDDS can be enabled by specifying AUTOEXPORT=N for the DFSDFxxx PROCLIB member and then starting IMS.

FPBP64STAT

Specifies whether Fast Path 64-bit buffer usage statistics per unit of work (UOW) for dependent regions are to be written to OLDS. The Fast Path 64-bit buffer usage is recorded in X'5945' log records.

N   Does not write Fast Path 64-bit buffer usage statistics per unit of work for dependent regions in X'5945' log records to OLDS.

Y   Writes Fast Path 64-bit buffer usage statistics per unit of work for dependent regions in X'5945' log records to OLDS.

LOCKTIME
Specifies the IMS LOCKTIME values. You can specify LOCKTIME values in any combination of the following keywords and parameters:

BMP(*xx*)
Specifies the amount of time that IMS waits before lock requests for BMP regions are timed out. BMP regions include IMS BMP and JBP regions. The value, which represents time in seconds, can range 1 - 32767.

BMPOPT(ABEND | STATUS)
Specifies whether IMS ends a timed-out task abnormally (ABEND) or returns a status code to the application (STATUS).

MSG(*yy*)
Specifies the amount of time that IMS waits before lock requests for MSG regions are timed out. MSG regions include IMS MPP, JMP, and IFP regions, as well as DRA threads. The value, which represents time in seconds, can range 1 - 32767.

MSGOPT(ABEND | STATUS)
Specifies whether IMS ends a timed-out task abnormally (ABEND) or returns a status code to the application (STATUS).

Updating IMS LOCKTIME values does not affect the lock timeout value in an IRLM. To change the lock timeout value in an IRLM, use the existing MODIFY IRLM command.

Recommendation: Generally, use the same timeout values for both IMS and IRLM. When using two values in IMS, setting the IRLM timeout value to the lower of the two IMS values allows IMS and IRLM to act together. When more than one IMS is identified to the same IRLM, workload conditions might require using different timeout values across IMS systems. In such a case, note the following:

• Using a lower timeout value in IRLM results in lock requests to wait until the time spent for waiting equals the lowest IMS LOCKTIME value.

• Using a higher timeout value in IRLM results in lock requests to wait beyond the IMS time because IRLM does not call IMS until the IRLM timeout value is exceeded.

REPO(Y)
Enables IMS to use the repository.

REPOTYPE(IMSRSC)
The repository type that is to be enabled. REPOTYPE is required with REPO(Y).

During command processing, IMS connects to RM for the repository services to enable IMS to use the repository. If RM is not enabled to use the repository, the UPDATE IMS command results in an error. You must first enable RM to use the repository by issuing the UPDATE RM command.

To disable use of the repository, IMS must be cold started.

When IMS is initialized, repository usage is enabled only if the REPOSITORY=(TYPE=IMSRSC) statement is defined in the REPOSITORY section of the DFSDFxxx PROCLIB member. Therefore, when the UPDATE IMS SET(LCLPARM(REPO(Y)

REPOTYPE(IMSRSC)) command is successfully processed, the REPOSITORY=(TYPE=IMSRSC) statement must be added before any IMS restarts because we can disable usage of the repository only across an IMS cold start.

During IMS restart, if the IMS log records indicate that the IMSRSC repository was enabled before the restart and the REPOSITORY=(TYPE=IMSRSC) statement is not defined, IMS terminates with a U0168 abend code and X'1C' error code.

For an XRF complex, entering the command on the IMS active system results in the command being processed on the IMS alternate system. Successful completion of the command requires that both the IMS active and alternate systems add the REPOSITORY=(TYPE=IMSRSC) statement in the REPOSITORY section of their respective DFSDFxxx PROCLIB members.

After IMS connects to the IMSRSC repository, autoexport to the IMSRSC repository is also enabled if AUTOEXPORT=AUTO or REPO is explicitly specified in the DFSDFxxx member that was used during IMS initialization.

The autoexport is done based on the AUTOEXPORT_IMSID= value specified in the DFSDFxxx member. If the AUTOEXPORT_IMSID= is not specified in the DFSDFxxx member then AUTOEXPORT_IMSID is set to THIS_IMS for the autoexport.

If you would like AUTOEXPORT_IMSID=ALL then the DFSDFxxx member must be modified to add the repository section and the AUTOEXPORT_IMSID=ALL and IMS must be restarted.

If AUTOEXPORT=AUTO or REPO was not explicitly specified in the DFSDFxxx member then autoexport to the IMSRSC repository is not enabled by the UPD IMS command. You must modify the DFSDFxxx member to add AUTOEXPORT=AUTO or REPO and the AUTOEXPORT_IMSID= keyword and restart IMS.

PLEXPARM()
Specifies the values for the global parameters to be modified. The LCLPARM keyword is mutually exclusive with the PLEXPARM keyword in the UPDATE IMS SET() command.

The UPDATE IMS SET(PLEXPARM()) command can be used to modify global status for areas, databases, and transactions that is maintained in an IMSplex. If the command is successful, the values in the RM global PLEXPARM entry along with the values maintained in each IMS system are updated.

For the UPDATE IMS SET(PLEXPARM()) command, all IMS systems in an IMSplex receive the command, but only the command master IMS processes the command. The non-master IMS systems do not process the command that is sent from OM. Instead, the command master IMS coordinates the change across all IMS systems in the IMSplex.

Specify one or more of the following resources:

GSTSAREA
Specifies how area status is maintained in the IMSplex.

N No global status is maintained for area resources in RM. All global status in RM for area resources is deleted.

Y Global status is maintained for area resources in RM. Global status is

maintained on subsequent UPD AREA SCOPE(ALL) commands or type-1 area commands that include a GLOBAL keyword.

GSTSDB
Specifies how database status is maintained in the IMSplex.

N   No global status is maintained for database resources in RM. All global status in RM for database resources is deleted.

Y   Global status is maintained for database resources in RM. Global status is maintained on subsequent UPD DB SCOPE(ALL) commands or type-1 database commands that include a GLOBAL keyword.

GSTSTRAN
Specifies how transaction status is maintained in the IMSplex.

N   No global status is maintained for transaction resources in RM. All global status in RM for transaction resources is deleted.

Y   Global status is maintained for transaction resources in RM. Global status is maintained on subsequent UPD TRAN SCOPE(ALL) commands.

OPTION()
Specifies options for the UPDATE IMS SET command.

CONTINUE
Indicates that IMS should continue to process a previously entered UPDATE IMS SET command that encountered an error before it completed. The command must be at a point that allows processing to continue, otherwise the command is aborted.

If OPTION(CONTINUE) is specified, no parameters can be passed on the PLEXPARM parameter.

FORCE
OPTION(FORCE) can be used in a DBCTL warm-standby environment when the UPD IMS command fails because the standby did not finish restart processing. If OPTION(FORCE) is specified, the UPD IMS command can be completed even when an IMS is in restart mode. When OPTION(FORCE) is specified, the IMS in restart mode does not participate and update the UPD IMS values; this option only allows the UPD IMS command to finish at other IMS systems. The DBCTL standby rereads the global entry when it restarts to become the active in order to obtain the current PLEXPARM values. If OPTION(FORCE) is used when a non-DBCTL standby system did not finish restart, the results might not be as expected because the IMS does not run the UPD IMS command. This could lead to errors because of mismatch in the PLEXPARM values.

Usage notes

The UPDATE IMS SET command is a type-2 command that must be issued through the OM API. The command is defined to OM as ROUTE=ALL. OM routes the command to all IMS systems in an IMSplex. This command is not a recoverable command.

The only case in which the command is processed from a log record is on an XRF alternate system, which reads an X'220E' log record of the XRF active system to stay in step with the XRF active. Even in the XRF environment, if a takeover occurs and an IMS restart is required (/ERE or /NRE), the command is not recovered. It requires that the DFSDFxxx PROCLIB member be changed to include

a REPOSITORY=(TYPE=IMSRSC) statement in the REPOSITORY section. Otherwise, IMS terminates with a U0168 abend code and X'1C' error code.

In general, when a nonzero return code is received for the UPDATE IMS command, you must enter the UPD IMS SET(PLEXPARM()) OPTION(CONTINUE) command, which tells IMS to either complete the command or to cancel the command and clean up as required. The reason code from the UPD IMS SET(PLEXPARM()) OPTION(CONTINUE) command might indicate the state of the original UPD IMS command. If the command is unable to determine the state of the original UPD IMS command, the reason code indicates that fact. The user can use the QUERY IMS command to find out the state of the global IMSplex parameters.

If an error is encountered during the UPD IMS command processing, another new UPD IMS command cannot be processed until the UPDATE IMS SET(PLEXPARM()) OPTION(CONTINUE) command is issued and completes successfully.

If the error condition that caused the nonzero return code to be received for the UPDATE IMS command is not resolved, the UPDATE IMS SET(PLEXPARM()) OPTION(CONTINUE) command continues to encounter the same error condition. The error condition must be resolved before the command can be cleaned up and another UPDATE IMS command can be entered.

This command is not valid on an RSR tracker, an FDBR system, or an XRF alternate system. However, an FDBR system and an XRF alternate system maintain global PLEXPARM values internally and report their values with a QUERY IMS SHOW (PLEXPARM) command. IMS systems on RST tracker or FDBR systems return a completion code of ICC_NA or 1. The UPD IMS command is allowed to complete at other IMS systems.

The UPD IMS SET command is not allowed if an IMS is in restart. The command fails with a 'B0' return code. The UPD IMS command must be issued after the IMS completed restart processing.

When REPO(Y) is specified, some error conditions result in a DFS3308E, DFS4400E, DFS4457E, or other error message being sent as message output in the command reply. These messages can be used to help diagnose the cause of the error.

Output fields

The following table shows the UPDATE IMS output fields. The columns in the table are as follows:

Short label
      Contains the short label generated in the XML output.

Long label
      Contains the long label generated in the XML output.

Keyword
      Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Scope Identifies the scope of the output field.

Meaning
    Provides a brief description of the output field.

Table 408. Output fields for the UPDATE IMS command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	n/a	n/a	The completion code for the line of output. The completion code is always returned.
LPARM	LclParmName	LCLPARM	LCL	The name of the local parameter that is being changed. Multiple subparameters of this parameter might be changed depending on what was entered in the command.
MBR	MbrName	n/a	n/a	The IMS identifier of the IMS that built the output line. The IMS identifier is always returned.
CCTXT	CCText	n/a	LCL	Completion code text returned to provide more information about the completion code. CC text could include a return code from a service. CC text is returned only if the completion code is nonzero.
PARMNAME	ParmName	PARMNAME	LCL	The name of the global parameter that is being changed. Multiple sub-parameters of this parameter may be changed depending on what was entered in the command.
REPOTP	RepositoryType	LCLPARM, REPO	LCL	The type of the repository.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMS command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 409. Return and reason codes for the UPDATE IMS command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMS command completed successfully.

Table 409. Return and reason codes for the UPDATE IMS command (continued)

Return code	Reason code	Meaning
X'00000004'	X'00001000'	The UPDATE IMS command was not processed on the IMS system because the IMS system is not the command master. No resource information is returned.
X'00000004'	X'00001020'	UPD IMS OPTION(CONTINUE) finished cleanup, but state is not known. User must redo or reissue the command.
X'00000008'	X'00002004'	Both LCLPARM and PLEXPARM keywords are specified on the UPDATE IMS SET() command. LCLPARM and PLEXPARM are mutually exclusive keywords.
X'00000008'	X'00002008'	Neither the PLEXPARM keyword nor the LCLPARM keyword is specified on the UPDATE IMS command. Specify PLEXPARM for global parameters or LCLPARM for local parameters.
X'00000008'	X'00002040'	More than one filter value is specified on the UPDATE IMS command.
X'0000000C'	X'00003004'	No requests were successful.
X'00000010'	X'00004000'	The UPDATE IMS command is not processed because IMS restart is not complete.
X'00000010'	X'00004004'	No CQS address space.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004018'	No resource structure, or resource structure is not available.
X'00000010'	X'0000401C'	Command is not valid on the FDBR region.
X'00000010'	X'00004100'	The resource structure is full.
X'00000010'	X'00004104'	The UPDATE IMS command is not processed because RM is not available.
X'00000010'	X'00004108'	The UPDATE IMS command is not processed because SCI is not available.
X'00000010'	X'00004124'	The UPDATE IMS command is not processed because a process step is in progress.
X'00000010'	X'00004128'	The UPDATE IMS command is not processed because no RM process step is in progress.
X'00000010'	X'00004200'	The UPDATE IMS command is not processed because IMS shutdown is in progress.
X'00000010'	X'00004208'	Invalid command for RMENV=NO.
X'00000010'	X'0000420C'	Error on non-command master. The UPD IMS could not complete because of an error on the non master IMS.

Table 409. Return and reason codes for the UPDATE IMS command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004300'	For REPO(Y) or AUTOEXPORT(N), the command is rejected because dynamic resource definition (MODBLK=DYN) is not enabled.
X'00000010'	X'00004501'	RM is not enabled with the repository.
X'00000014'	X'00005000'	The UPDATE IMS command is not processed as the IMODULE GETSTOR storage could not be obtained.
X'00000014'	X'00005004'	The UPDATE IMS command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005100'	The UPDATE IMS command is not processed because of a RM request error.
X'00000014'	X'00005104'	The UPDATE IMS command is not processed because of a CQS error.
X'00000014'	X'00005108'	The UPDATE IMS command is not processed because of a SCI request error.
X'00000014'	X'0000510C'	The UPDATE IMS command is not processed as another process is in progress.
X'00000014'	X'00005110'	Repository error.
X'00000014'	X'00005FFF'	The UPDATE IMS command is not processed because of an IMS internal error.

Errors that are unique to the processing of the UPDATE IMS command are
returned as completion codes. A completion code is returned for each action
against an individual resource.

Table 410. Completion codes for the UPDATE IMS command

Completion code	Completion code text	Meaning
0		The UPDATE IMS command completed successfully.
1		Command is not applicable to the IMS.
55	NO FASTPATH INSTALLED	Command is not applicable because Fast Path is not installed.
B0		IMS restart is not complete. The UPD IMS command could not complete because an IMS did not complete restart. You need to terminate the UPD IMS command with an UPD IMS SET(PLEXPARM()) OPTION(CONTINUE). The UPD IMS SET(PLEXPARM(...)) command must be issued after the IMS finishes restart processing.
BB		Command is not applicable as RMENV=NO.
146	INVALID KEYWORD FOR DCCTL	Command is not applicable in a DCCTL system.
147	FPBP64 NOT ENABLED	Command is not applicable because Fast Path 64-bit buffer manager is not enabled.

Examples

The following are examples of the UPDATE IMS command:

Example 1 for UPDATE IMS command

TSO SPOC input:
```
UPDATE IMS SET(PLEXPARM(GSTSAREA(Y),GSTSDB(Y),GSTSTRAN(N)))
```

TSO SPOC output:
```
Response for UPDATE IMS SET(PLEXPARM(GSTSAREA(Y),GSTSDB(Y),GSTSTRAN(N)))
ParmName  MbrName    CC
PLEXPARM  PRODIMS1   0
PLEXPARM  PRODIMS2   0
PLEXPARM  SYS3       0
```

Explanation: The UPD IMS command is processed successfully at the three active IMS systems (PRODIMS1, PRODIMS2, and SYS3) in the IMSplex and is updated successfully in RM.

Example 2 for UPDATE IMS command

TSO SPOC input:
```
UPDATE IMS SET(PLEXPARM(GSTSTRAN(Y)))
```

TSO SPOC output:
```
Response for UPDATE IMS SET(PLEXPARM(GSTSTRAN(Y)))
ParmName   MbrName  CC       GSTSTRAN
PLEXNAME   PRODIMS1 0        Y
PLEXNAME   PRODIMS2 0        Y
PLEXNAME   SYS3     0        Y
```

Explanation: The UPD IMS command is processed successfully at the three active IMS systems (PRODIMS1, PRODIMS2, and SYS3) in the IMSplex and is updated successfully in RM.

Example 3 for UPDATE IMS command

TSO SPOC input:
```
UPDATE IMS SET(PLEXPARM(GSTSTRAN(N)))
```

TSO SPOC output:
```
Response for: UPDATE IMS SET(GSTSTRAN(Y))
MbrName    CC   GSTSTRAN CCText
SYS3       0    Y
PRODIMS1   0    Y
PRODIMS2   16   Y        IMS TIMED OUT
```

Example 4 for UPDATE IMS command

TSO SPOC input:
```
UPDATE IMS SET(LCLPARM(FPBP64STAT(Y)))
```

TSO SPOC output:
```
    ParmName LclParmName MbrName    CC
    LCLPARM  FPBP64STAT  IMS1       0
    LCLPARM  FPBP64STAT  IMS2       0
```

Explanation: In this example, the UPDATE IMS command turns on FPBP64STAT logging.

Example 5 for UPDATE IMS command

TSO SPOC input:

```
UPDATE IMS SET(LCLPARM(FPBP64STAT(N)))
```

TSO SPOC output:

```
    ParmName LclParmName MbrName    CC
    LCLPARM  FPBP64STAT  IMS1        0
    LCLPARM  FPBP64STAT  IMS2        0
```

Explanation: In this example, the UPDATE IMS command turns off FPBP64STAT logging.

Example 6 for UPDATE IMS command

TSO SPOC input:

```
UPD IMS SET(LCLPARM(LOCKTIME(MSG(40),MSGOPT(ABEND),BMP(30),
BMPOPT(STATUS))))
```

TSO SPOC output:

```
  ParmName LclParmName MbrName     CC
  LCLPARM  LOCKTIME    IMS1         0
  LCLPARM  LOCKTIME    IMS2         0
```

Explanation: In this example, the UPDATE IMS command updates IMS LOCKTIME values.

Example 7 for UPDATE IMS command

TSO SPOC input:

```
UPD IMS SET(LCLPARM(REPO(Y) REPOTYPE(IMSRSC)))
```

TSO SPOC output:

```
ParmName LclParmName RepositoryType MbrName     CC
LCLPARM  REPO        IMSRSC         IMS1         0
LCLPARM  REPO        IMSRSC         IMS2         0
```

OM API input:

```
CMD(UPD IMS SET(LCLPARM(REPO(Y) REPOTYPE(IMSRSC))))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.190 01:45:07.574341</statime>
<stotime>2011.190 01:45:07.618879</stotime>
<staseq>C80A9624D1245C48</staseq>
<stoseq>C80A9624DC03F148</stoseq>
<rqsttkn1>USRT005 10184507</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
```

```
<master>IMS1    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>IMS            </kwd>
<input>UPD IMS SET(LCLPARM(REPO(Y) REPOTYPE(IMSRSC))) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PARM" llbl="ParmName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="LPARM" llbl="LclParmName" scope="LCL" sort="a" key="2"
 scroll="no" len="10" dtype="CHAR" align="left" />
<hdr slbl="REPOTP" llbl="RepositoryType" scope="LCL" sort="n" key="0"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PARM(LCLPARM) LPARM(REPO      ) REPOTP(IMSRSC  ) MBR(IMS1) CC(
 0) </rsp>
<rsp>PARM(LCLPARM) LPARM(REPO      ) REPOTP(IMSRSC  ) MBR(IMS2) CC(
 0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: In this example, IMS1 and IMS2 are enabled to use the repository for
REPOTYPE=IMSRSC. The user needs to issue the QUERY IMS command to obtain
the repository name.

Example 8 for UPDATE IMS command

TSO SPOC input:

```
UPDATE IMS SET(LCLPARM(AUTOEXPORT(N))
```

TSO SPOC output:

```
ParmName LclParmName MbrName CC
LclParm  AUTOEXPORT  IMS1     0
LclParm  AUTOEXPORT  IMS2     0
```

Explanation: In this example, IMS1 and IMS2 have AUTOEXPORT disabled at the
next checkpoint.

Related concepts:

[➡] How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

[➡] Command keywords and their synonyms (Commands)

[➡] DFSDFxxx member of the IMS PROCLIB data set (System Definition)

UPDATE IMSCON commands

Use the UPDATE IMSCON commands to update status or configuration definitions
of one or more IMS Connect resources.

The TYPE keyword is a required keyword that specifies the type of IMS Connect
resource to update. There is no default TYPE parameter.

The UPDATE IMSCON command is processed by every IMS Connect to which OM routes the command, whether or not OM has designated a particular IMS Connect as the command master.

Subsections:
- "UPDATE IMSCON TYPE(ALIAS) command"
- "UPDATE IMSCON TYPE(CLIENT) command" on page 1091
- "UPDATE IMSCON TYPE(CONFIG) command" on page 1095
- "UPDATE IMSCON TYPE(CONVERTER) command" on page 1108
- "UPDATE IMSCON TYPE(DATASTORE) command" on page 1112
- "UPDATE IMSCON TYPE(IMSPLEX) command" on page 1126
- "UPDATE IMSCON TYPE(ISC) command" on page 1133
- "UPDATE IMSCON TYPE(ISCUSER) command" on page 1139
- "UPDATE IMSCON TYPE(LINK) command" on page 1143
- "UPDATE IMSCON TYPE(MSC) command" on page 1148
- "UPDATE IMSCON TYPE(ODBM) command" on page 1154
- "UPDATE IMSCON TYPE(PORT) command" on page 1159
- "UPDATE IMSCON TYPE(RACFUID) command" on page 1170
- "QUERY IMSCON TYPE(RMTCICS) command" on page 244
- "UPDATE IMSCON TYPE(RMTIMSCON) command" on page 1180
- "UPDATE IMSCON TYPE(SENDCLNT) command" on page 1187

Related reference:

➡ Equivalent IMS Connect WTOR, z/OS, and type-2 commands (Commands)

UPDATE IMSCON TYPE(ALIAS) command

Use the UPDATE IMSCON TYPE(ALIAS) command to set or change the attributes of an IMS alias that is associated with an IMS Open Database Manager (ODBM).

Subsections:
- "Environment"
- "Syntax" on page 1087
- "Keywords" on page 1087
- "Usage notes" on page 1088
- "Equivalent WTOR and z/OS commands" on page 1088
- "Output fields" on page 1088
- "Return, reason, and completion codes" on page 1089
- "Examples" on page 1090

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
>>--+--UPDATE--+--IMSCON--TYPE(ALIAS)--NAME(--+<----,----+--+------)--------------->
    +--UPD-----+                              +--alias_name---+
                                              +--alias_name*--+

>--+--------------------------------+--+--START(ROUTE)--+---------------------><
   |              +<-----,-----+     |  +--STOP(ROUTE)---+
   +--ODBM(--+----+--odbm_name--+-+--)--+
             |    +--odbm_name*--+ |
             +------------------------+
```
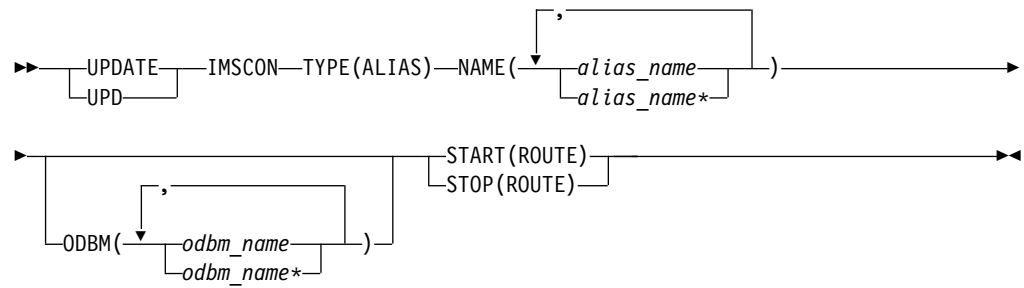
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(ALIAS) command.

NAME
> Specifies the name of the IMS alias. You can specify a single alias name or a list of alias names separated by commas. Wildcards can be used in the names.

ODBM
> Specifies the name of the IMS ODBM to be updated. You can specify a single ODBM name or a list of ODBM names separated by commas. Wildcards can be used in the names. This keyword is optional.

START(ROUTE) | STOP(ROUTE)
> Mutually exclusive keywords that you use to enable or disable message routing to an IMS ODBM that is associated with the specified IMS alias.

> **START(ROUTE)**
> > Enables message routing to an IMS ODBM that is associated with the specified IMS alias. The alias represents the IMS data store that the client wants to send a message to. The START(ROUTE) keyword sets the specified IMS alias to active so that IMS Connect can route to that alias.

> > Whereas the UPDATE IMSCON TYPE(ODBM) command starts communication between the IMS Connect and an IMS ODBM, the UPDATE IMSCON TYPE(ALIAS) command sets the specified IMS alias to active so that IMS Connect can route to that alias.

> **STOP(ROUTE)**
> > Disables message routing to an IMS ODBM that is associated with the specified IMS alias. The STOP(ROUTE) keyword sets the specified alias to inactive so that IMS Connect cannot route to that alias. You can resume routing by specifying the START(ROUTE) keyword.

> > If you stop routing messages to a particular alias by using the STOP(ROUTE) keyword, and then issue the UPDATE IMSCON TYPE(ODBM) STOP(COMM) command followed by UPDATE IMSCON TYPE(ODBM) START(COMM), the alias status is lost. In other words, the alias will be active again after the UPDATE IMSCON TYPE(ODBM) commands.

Usage notes

You can issue the UPDATE IMSCON TYPE(ALIAS) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(ALIAS) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 411. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(ALIAS) command

UPDATE IMSCON TYPE(ALIAS) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UDPATE IMSCON TYPE(ALIAS) NAME(*alias_name*) ODBM(*odbm_name*) START(COMM)	STARTIA *alias_name* *odbm_name*	UPDATE ALIAS NAME(*aliasName*) ODBM(*odbmName*) START(ROUTE)
UDPATE IMSCON TYPE(ALIAS) NAME(*alias_name*) ODBM(*odbm_name*) STOP(COMM)	STOPIA *alias_name* *odbm_name*	UPDATE ALIAS NAME(*aliasName*) ODBM(*odbmName*) STOP(ROUTE)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) is displayed for output fields that are always returned. *error* is displayed for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 412. Output fields for the UPDATE IMSCON TYPE(ALIAS) command

Short label	Long label	Keyword	Meaning
ALIAS	AliasName	N/A	The alias name of an IMS data store defined to the instance of ODBM. The alias name is always returned.
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 414.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
ODBM	ODBMName	ODBM	Name of the ODBM associated with the alias.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(ALIAS) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 413. Return and reason codes for the UPDATE IMSCON TYPE(ALIAS) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(ALIAS) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 414. Completion codes for the UPDATE IMSCON TYPE(ALIAS) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(ALIAS) command completed successfully for the resources.

Table 414. Completion codes for the UPDATE IMSCON TYPE(ALIAS) command (continued)

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for UPDATE IMSCON TYPE(ALIAS) command

TSO SPOC input:

```
UDPATE IMSCON TYPE(ALIAS) NAME(IMS1) ODBM(ODBMA) START(ROUTE)
```

TSO SPOC output:

```
AliasName   MbrName   CC      ODBMName
IMS1        HWS1       0      ODBMA
```

OM API input:

```
CMD ( UDPATE IMSCON TYPE(ALIAS) NAME(IMS1) ODBM(ODBMA) START(ROUTE) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 15:34:49.371591</statime>
<stotime>2010.298 15:34:49.372641</stotime>
<staseq>C6C83044FA3C7630</staseq>
<stoseq>C6C83044FA7E1E70</stoseq>
<rqsttkn1>USRT001 10083449</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(ALIAS) NAME(IMS1) ODBM(ODBMA) START(ROUTE)
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="ALIAS" llbl="AliasName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ODBM" llbl="ODBMName" scope="LCL" sort="a" key="3"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
</cmdrsphdr>
<cmdrspdata>
```

```
<rsp>ALIAS(IMS1   ) MBR(HWS1          ) CC(   0) ODBM(ODBMA )
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: Message routing to the IMS Open Database (ODBM), ODBMA, associated with the IMS alias, IMS1, has been enabled.

Related reference:

➡️ STARTIA command (Commands)

➡️ STOPIA command (Commands)

UPDATE IMSCON TYPE(CLIENT) command

Use the UPDATE IMSCON TYPE(CLIENT) command to terminate communication with a client that uses a specific TCP/IP port.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 1092
- "Equivalent WTOR and z/OS commands" on page 1094
- "Output fields" on page 1092
- "Return, reason, and completion codes" on page 1093
- "Examples" on page 1094

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►──┬─UPDATE─┬──IMSCON──TYPE(CLIENT)──NAME(client_name)──────────────────►
    └─UPD────┘


►──┬──────────────────────────────┬──STOP(COMM)──────────────────────►◄
   │            ┌─,─────────┐      │
   └─PORT(──▼──┬─LOCAL───┬──┴──)───┘
              ├─portnum──┤
              └─portnum*─┘
```
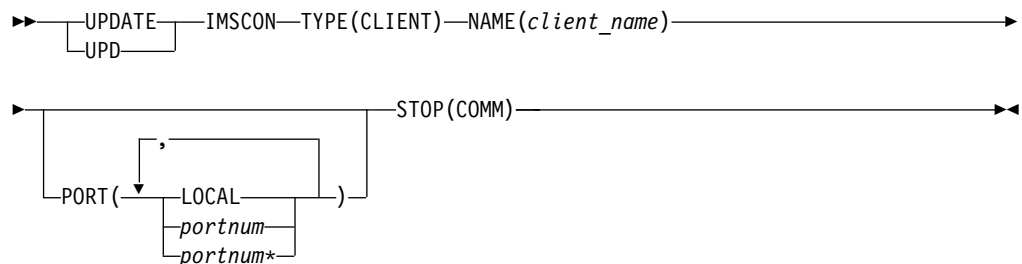
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(CLIENT) command.

NAME
Specifies the name of the client.

PORT
Specifies the port that the client is using for the TCP/IP connection with the IMS Connect.

This port number must match the port number defined in the PORT, PORTID, or CICSPORT substatement of the TCPIP configuration statement, or the DRDAPORT substatement of the ODACCESS configuration statement in the HWSCFGxx configuration member.

To filter on the local port used by the IMS TM Resource Adapter, specify PORT(LOCAL). You can also use wildcards in the port number.

The SSL port is displayed with the character "S" appended to the end of the port number. To filter on the SSL port, specify the port number either with or without the character "S" appended to the end of the port number.

The port defined for ODBM use is displayed with the character "D" appended to the end of the port number. To filter on the ODBM port, specify the port number either with or without the character "D" appended to the end of the port number.

The port defined for ISC over TCPIP use (CICSPORT) is displayed with the character "C" appended to the end of the port number. To filter on the CICSPORT, specify the port number either with or without the character "C" appended to the end of the port number.

STOP(COMM)
Stops communication with a client that uses a specific TCP/IP port.

Usage notes

You can issue the UPDATE IMSCON TYPE(CLIENT) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

Use the UPDATE IMSCON TYPE(CLIENT) command whenever a client is unable to accept response messages being sent to it, or when a client is waiting for a nonexistent response message (for example, when an error occurred that caused a response message to be lost before it was sent back to the client).

Work currently in progress for that client is ended.

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be

generated. N/A (not applicable) is displayed for output fields that are always returned. *error* is displayed for output fields that are returned only in the case of an error.

Meaning

Provides a brief description of the output field.

Table 415. Output fields for the UPDATE IMSCON TYPE(CLIENT) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 417 on page 1094.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
CLID	ClientID	N/A	Name of the client. The client name is always returned.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
PORT	Port	N/A	The port number on which the client is active.
			If one of the following characters is appended to the end of the port number, it indicates that the port is dedicated to a particular purpose:
			C Identifies a CICS port.
			D Identifies an ODBM port.
			S Identifies an SSL port.
			If "LOCAL" is displayed instead of a port number, the port is a local port that is used by the IMS TM Resource Adapter.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(CLIENT) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 416. Return and reason codes for the UPDATE IMSCON TYPE(CLIENT) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(CLIENT) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.

Table 416. Return and reason codes for the UPDATE IMSCON TYPE(CLIENT) command (continued)

Return code	Reason code	Meaning
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 417. Completion codes for the UPDATE IMSCON TYPE(CLIENT) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(CLIENT) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(CLIENT) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 418. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(CLIENT) command

UPDATE IMSCON TYPE(CLIENT) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(CLIENT) NAME(*client_name*) PORT(*portid*) STOP(COMM)	STOPCLNT *portid clientid*	DELETE PORT NAME(*portName*) CLIENT(*clientName*)

Examples

Example 1 for UPDATE IMSCON TYPE(CLIENT) command

TSO SPOC input:
```
UPD IMSCON TYPE(CLIENT) NAME(CLIENT01) PORT(9999)  STOP(COMM)
```

TSO SPOC output:

```
ClientName    Port    MbrName   CC
CLIENT01      9999    HWS1       0
```

OM API input:

```
CMD(UPD IMSCON TYPE(CLIENT) NAME(CLIENT01) PORT(9999)  STOP(COMM))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 02:09:29.445456</statime>
<stotime>2010.298 02:09:29.446600</stotime>
<staseq>C6C77C43814502A2</staseq>
<stoseq>C6C77C43818C84E2</stoseq>
<rqsttkn1>USRT001 10190929</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(CLIENT) NAME(CLIENT01) PORT(9999)  STOP(COMM)
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="CLID" llbl="ClientID" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left"skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>CLID(CLIENT01) PORT(9999    ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The communication with client CLIENT01 on port 9999 has been terminated.

Related reference:

➡ STOPCLNT command (Commands)

➡ IMS Connect DELETE CLIENT command (Commands)

UPDATE IMSCON TYPE(CONFIG) command

Use the UPDATE IMSCON TYPE(CONFIG) command to terminate IMS Connect with the option QUIESCE or FORCE, to open and close the line trace data set, and to enable and disable features such as ODBM registration, password support, RACF, and z/OS Resource Recovery Services (RRS).

Subsections:

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax
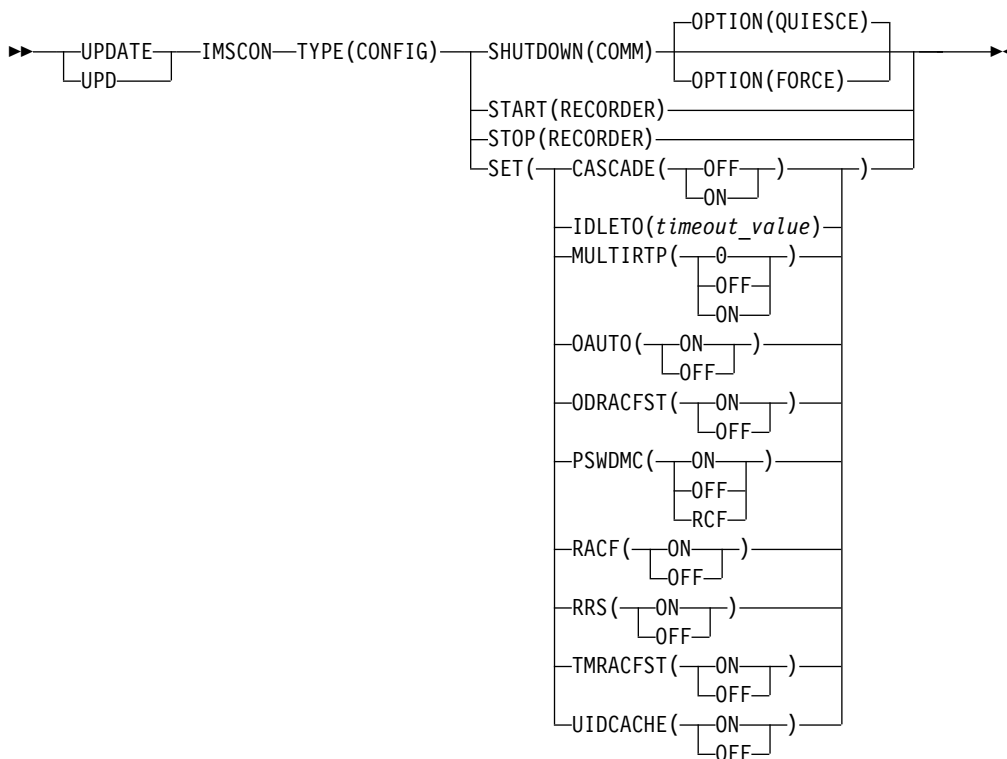
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(CONFIG) command.

SHUTDOWN(COMM)

Specifies that IMS Connect ends all client and data store connections in a controlled manner.

OPTION

Specifies an option for SHUTDOWN.

QUIESCE

Ends all client and data store connections in a controlled manner. If no parameter is specified for the SHUTDOWN keyword, QUIESCE is used by default.

All work that is in progress, or that is queued for processing, is completed before IMS Connect is stopped. No new work is accepted after this command is entered and accepted.

IMS Connect shuts down in the following sequence:
1. All active units of work for clients and browsers are completed.
2. Communication between IMS Connect and IMS is terminated.
3. IMS Connect terminates.

FORCE

Ends all client and data store connections immediately. This keyword forces any IMS applications that are being run for the connected clients to end abnormally.

SET

Specifies the attribute values to be changed.

CASCADE=

Defines the IMS Connect default setting for support for cascading global, two-phase commit (synclevel=syncpoint) transactions from IMS TM Resource Adapter via a TCP/IP connection to an IMS TM or IMS DB/DC system that is on a different z/OS image than IMS Connect. Cascading transactions that are received via a LOCAL option connection is not supported.

A CASCADE specification in the definition of a data store connection overrides the CASCADE specification in the IMS Connect system configuration. Any data store connection definition that does not specify a value for CASCADE inherits the CASCADE specification from the IMS Connect system configuration.

Until the next restart of IMS Connect, a CASCADE specification made by the UPDATE IMSCON TYPE(CONFIG) SET(CASCADE()) command overrides the CASCADE specification on the HWS configuration statement.

The CASCADE parameter applies only to the support for cascading global, RRS transactions between IMS Connect and IMS TM when they are on different z/OS images. The specification of the IMS Connect CASCADE parameter does not affect support between other components or subsystems, such as between IMS subsystems in a shared queues environment.

OFF

Specifies that, by default, data store connections do not support the cascading of global, two-phase commit (synclevel=syncpoint) transactions to IMS TM and IMS DB/DC systems that are running on a different z/OS image than IMS Connect.

If a synclevel=syncpoint transaction is routed to an IMS system that resides on a different LPAR than IMS Connect, IMS Connect rejects the transaction with return code 4 and reason code NCASCADE.

ON Specifies that, by default, data store connections support the cascading of global, two-phase commit (synclevel=syncpoint transactions to IMS TM and IMS DB/DC systems that are running on different z/OS images than IMS Connect.

If IMS Connect and IMS reside on the same LPAR, the transaction is processed without using cascaded transaction support. If IMS Connect and IMS reside on different LPARs, the transaction is processed using cascaded transaction support.

IDLETO

For ports defined for communications with IMS TM, specifies the global timeout interval for open socket connections that are idle and are waiting in a RECV state for the next message. The global timeout value can be overridden for individual ports that are defined by the PORT= parameter by specifying IDLETO= in the port definition.

When the timeout interval expires, IMS Connect closes the socket connection due to inactivity.

The timeout interval is in hundredths of seconds. Timeout values can be 0 - 2 147 483 647 (X'7FFFFFFF'). A value of 0 disables this timeout function, which prevents inactive connections from timing out.

This parameter is optional and applies only to persistent socket connections.

MULTIRTP=

Sets the IMS Connect default specification for OTMA multiple RESUME TPIPE (MULTIRTP) support.

The IMS Connect default specification for MULTIRTP support can be overridden for individual DATASTORE connections by specifying a MULTIRTP value in the DATASTORE definition.

If MULTIRTP is not specified on either the HWS statement or a DATASTORE statement, the OTMA MULTIRTP value defines the default.

0 The numeric character "0" removes any existing MULTIRTP specification from the IMS configuration so that, by default, data store client bid requests do not include any specification for MULTIRTP support. MULTIRTP support is determined by either the DATASTORE definition or the OTMA client descriptor in the DFSYDT*x* member of the PROCLIB data set.

OFF

Specifies that IMS Connect client bid requests indicate that IMS Connect requires an OTMA tpipe that supports only a single active RESUME TPIPE request. Output messages on the tpipe are sent serially.

ON Specifies that IMS Connect client bid requests require an OTMA TPIPE that can support multiple active RESUME TPIPE requests in parallel. When a tpipe has multiple active RESUME TPIPE requests from multiple DATASTORE connections, OTMA sends the callout or CM0 output messages on the first available DATASTORE connection that is in a receive state.

OAUTO

Specifies whether IMS Connect automatically connects to ODBM when an instance of ODBM is activated in the IMSplex.

ON IMS Connect will connect to all the future ODBMs that enter the IMSplex. This option can be specified in the ODBMATOCONN= parameter of the IMS Connect configuration member.

OFF

IMS Connect will not register with any future ODBMs that enter the IMSplex. After turning off the automatic connection of IMS Connect to ODBM, you can connect to ODBM manually by using the UPDATE IMSCON TYPE(ODBM) START(COMM) or equivalent command to open communication with an instance of ODBM.

ODRACFST

Specifies whether RACF statistics are recorded and updated when IMS Connect issues the RACROUTE REQUEST=VERIFY command to authenticate connections from an ODBM client to IMS DB.

This keyword is used only if RACF=Y is specified in the HWS statement of the HWSCFG*xx* configuration member.

ON RACF statistics are recorded and updated when IMS Connect issues the RACROUTE REQUEST=VERIFY command to authenticate connections from an ODBM client to IMS DB. If the logon is successful, a message is also issued. When you specify ODRACFST=Y, the STAT=ASIS parameter is used by IMS Connect on the RACROUTE REQUEST=VERIFY command. With STAT=ASIS, the RACF messages and statistics are controlled by the installation's current options on the RACF command SETROPTS.

After you enable RACF statistics, the statistics are recorded by RACF no more than once per day to a system management facility (SMF) data set or log stream. The SMF data set or log stream that is used to record the RACF statistics is specified in the RACF configuration.

OFF

RACF statistics are not recorded when IMS Connect issues the RACROUTE REQUEST=VERIFY command to authenticate connections from an ODBM client to IMS DB. Also, the statistics are not updated and, if the logon is successful, no message is issued. When you specify ODRACFST=N, the STAT=NO parameter is used by IMS Connect on the RACROUTE REQUEST=VERIFY command. With STAT=NO, the options specified on the RACF command SETROPTS are ignored.

For more information about the STAT= parameter of the RACROUTE REQUEST=VERIFY macro call, see "z/OS Security Server RACROUTE Macro Reference" in the IBM z/OS documentation.

PSWDMC

Specifies whether IMS Connect support for mixed-case passwords is to be turned on or off, or if it depends on the mixed-case password specification in RACF.

ON Enables IMS Connect support for mixed-case passwords.

OFF

Disables IMS Connect support for mixed-case passwords. If mixed-case password support is disabled, IMS Connect converts any lowercase characters in passwords to uppercase characters.

RCF

Depends on the mixed-case password specification in RACF. If it is off, IMS Connect converts any lowercase characters in passwords to uppercase characters.

RACF

Specifies that the RACF flag is to be turned on or off.

ON Enables the RACF user identification and verification.

OFF

Disables the RACF user identification and verification.

RRS

Specifies that z/OS Resource Recovery Services (RRS) is to be turned on or off. RRS is required for two-phase-commit support.

ON Enables communication between IMS Connect and RRS.

OFF

Disables communication between IMS Connect and RRS.

TMRACFST

Specifies whether RACF statistics are recorded and updated when IMS Connect issues the RACROUTE REQUEST=VERIFY command to authenticate connections from an OTMA client to IMS TM.

This keyword is used only if RACF=Y is specified in the HWS statement of the HWSCFGxx configuration member.

ON RACF statistics are recorded and updated when IMS Connect issues the RACROUTE REQUEST=VERIFY command to authenticate connections from an OTMA client to IMS TM. If the logon is successful, a message is also issued. When you specify TMRACFST=Y, the STAT=ASIS parameter is used by IMS Connect on the RACROUTE REQUEST=VERIFY command. With STAT=ASIS, the RACF messages and statistics are controlled by the installation's current options on the RACF command SETROPTS.

After you enable RACF statistics, the statistics are recorded by RACF no more than once per day to a system management facility (SMF) data set or log stream. The SMF data set or log stream that is used to record the RACF statistics is specified in the RACF configuration.

OFF

RACF statistics are not recorded when IMS Connect issues the RACROUTE REQUEST=VERIFY command to authenticate connections from an OTMA client to IMS TM. Also, the statistics are not updated and, if the logon is successful, no message is issued. When you specify TMRACFST=N, the STAT=NO parameter is used by IMS Connect on the RACROUTE REQUEST=VERIFY command. With STAT=NO, the options specified on the RACF command SETROPTS are ignored.

For more information about the STAT= parameter of the RACROUTE REQUEST=VERIFY macro call, see "z/OS Security Server RACROUTE Macro Reference" in the IBM z/OS documentation.

UIDCACHE

Specifies whether RACF user ID caching is used when RACF authentication is enabled.

ON Enables RACF user ID caching when RACF authentication is enabled.

OFF
> Disables RACF user ID caching when RACF authentication is enabled.

START(RECORDER)
> Starts the line trace data set for IMS Connect. This keyword is mutually exclusive with STOP(RECORDER).

STOP(RECORDER)
> Stops the line trace data set for IMS Connect. This keyword is mutually exclusive with START(RECORDER).

Usage notes

You can issue the UPDATE IMSCON TYPE(CONFIG) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(CONFIG) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 419. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(CONFIG) command

UPDATE IMSCON TYPE(CONFIG) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(CONFIG) SHUTDOWN(COMM)	CLOSEHWS	SHUTDOWN MEMBER
UPDATE IMSCON TYPE(CONFIG) SHUTDOWN(COMM) OPTION(FORCE)	CLOSEHWS FORCE	SHUTDOWN MEMBER OPTION(FORCE)
UPDATE IMSCON TYPE(CONFIG) SHUTDOWN(COMM) OPTION(QUIESCE)	CLOSEHWS QUIESCE	SHUTDOWN MEMBER OPTION(QUIESCE)
UPDATE IMSCON TYPE(CONFIG) SET(OAUTO(ON))	SETOAUTO YES	UPDATE MEMBER TYPE(IMSCON) SET(OAUTO(ON))
UPDATE IMSCON TYPE(CONFIG) SET(OAUTO(OFF)	SETOAUTO NO	UPDATE MEMBER TYPE(IMSCON) SET(OAUTO(OFF))
UPDATE IMSCON TYPE(CONFIG) SET(PSWDMC(ON))	SETPWMC ON	UPDATE MEMBER TYPE(IMSCON) SET(PSWDMC(ON))

Table 419. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(CONFIG) command (continued)

UPDATE IMSCON TYPE(CONFIG) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(CONFIG) SET(PSWDMC(OFF))	SETPWMC OFF	UPDATE MEMBER TYPE(IMSCON) SET(PSWDMC(OFF))
UPDATE IMSCON TYPE(CONFIG) SET(PSWDMC(RCF))	SETPWMC RCF	UPDATE MEMBER TYPE(IMSCON) SET(PSWDMC(RCF))
UPDATE IMSCON TYPE(CONFIG) SET(RACF(ON))	SETRACF ON	UPDATE MEMBER TYPE(IMSCON) SET(RACF(ON))
UPDATE IMSCON TYPE(CONFIG) SET(RACF(OFF))	SETRACF OFF	UPDATE MEMBER TYPE(IMSCON) SET(RACF(OFF))
UPDATE IMSCON TYPE(CONFIG) SET(RRS(ON))	SETRRS ON	UPDATE MEMBER TYPE(IMSCON) SET(RRS(ON))
UPDATE IMSCON TYPE(CONFIG) SET(RRS(OFF))	SETRRS OFF	UPDATE MEMBER TYPE(IMSCON) SET(RRS(OFF))
UPDATE IMSCON TYPE(CONFIG) SET(UIDCACHE(ON))	SETUIDC ON	UPDATE MEMBER TYPE(IMSCON) SET(UIDCACHE(ON))
UPDATE IMSCON TYPE(CONFIG) SET(UIDCACHE(OFF))	SETUIDC OFF	UPDATE MEMBER TYPE(IMSCON) SET(UIDCACHE(OFF))
UPDATE IMSCON TYPE(CONFIG) START(RECORDER)	RECORDER OPEN	UPDATE MEMBER TYPE(IMSCON) START(TRACE)
UPDATE IMSCON TYPE(CONFIG) STOP(RECORDER)	RECORDER CLOSE	UPDATE MEMBER TYPE(IMSCON) STOP(TRACE)

Output fields

Short label
>   Contains the short label that is generated in the XML output.

Long label
>   Contains the column heading displayed on the TSO SPOC screen.

Keyword
>   Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
>   Provides a brief description of the output field.

Table 420. Output fields for the UPDATE IMSCON TYPE(CONFIG) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 422.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.

Return and reason codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(CONFIG) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 421. Return and reason codes for the UPDATE IMSCON TYPE(CONFIG) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(CONFIG) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C000008'	X'000020C0'	The command specified an IDLETO value that was outside of the valid range. The command was rejected.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Completion codes

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 422. Completion codes for the UPDATE IMSCON TYPE(CONFIG) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(CONFIG) command completed successfully for the resources.

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for UPDATE IMSCON TYPE(CONFIG) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(CONFIG) SET(RRS(ON))
```

TSO SPOC output:

```
MbrName  CC
HWS1      0
```

OM API input:

```
CMD(UPDATE IMSCON TYPE(CONFIG) SET(RRS(ON)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 02:12:57.587305</statime>
<stotime>2010.298 02:12:57.590267</stotime>
<staseq>C6C77D0A01269015</staseq>
<stoseq>C6C77D0A01DFB355</stoseq>
<rqsttkn1>USRT001 10191257</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(CONFIG) SET(RRS(ON)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: RRS has been enabled for IMS Connect, HWS1.

Example 2 for UPDATE IMSCON TYPE(CONFIG) command

TSO SPOC input:
```
UPDATE IMSCON TYPE(CONFIG) SHUTDOWN(COMM) OPTION(FORCE)
```

TSO SPOC output:
```
MbrName  CC
HWS1      0
```

OM API input:
```
CMD(UPDATE IMSCON TYPE(CONFIG) SHUTDOWN() OPTION(FORCE))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 02:14:58.410022</statime>
<stotime>2010.298 02:15:14.132658</stotime>
<staseq>C6C77D7D3AE26562</staseq>
<stoseq>C6C77D8C396B2020</stoseq>
<rqsttkn1>USRT001 10191458</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON         </kwd>
<input>UPD IMSCON TYPE(CONFIG) SHUTDOWN(COMM) OPTION(FORCE) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS Connect, HWS1, has been shut down.

Example 3 for UPDATE IMSCON TYPE(CONFIG) command

TSO SPOC input:
```
UPDATE IMSCON TYPE(CONFIG) START(RECORDER)
```

TSO SPOC output:
```
MbrName  CC
HWS1      0
```

OM API input:

```
CMD(UPDATE IMSCON TYPE(CONFIG) START(RECORDER))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 02:32:07.499478</statime>
<stotime>2010.298 02:32:07.503090</stotime>
<staseq>C6C78152A56D6F7E</staseq>
<stoseq>C6C78152A64F223E</stoseq>
<rqsttkn1>USRT001 10193207</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(CONFIG) START(RECORDER) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The IMS Connect line trace data set has been opened.

Example 4 for UPDATE IMSCON TYPE(CONFIG) command

In the following example, the SET(ODRACFST(ON)) keyword is used to update
the RACF statistics option so that RACF statistics are enabled when IMS Connect
issues the RACROUTE REQUEST=VERIFY call for ODBM clients.

TSO SPOC input:

```
UPD IMSCON TYPE(CONFIG) SET(ODRACFST(ON))
```

TSO SPOC output:

```
MbrName            CC
HWS1                0
```

OM API input:

```
CMD(UPD IMSCON TYPE(CONFIG) SET(ODRACFST(ON)))
```

OM API output:

```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20   </xmlvsn>
```

```
|                          <statime>2017.111 00:36:43.850434</statime>
|                          <stotime>2017.111 00:36:43.850679</stotime>
|                          <staseq>D26B2BC6296C2F69</staseq>
|                          <stoseq>D26B2BC6297B7EE9</stoseq>
|                          <rqsttkn1>USRT011 10173643</rqsttkn1>
|                          <rc>00000000</rc>
|                          <rsn>00000000</rsn>
|                          </ctl>
|                          <cmd>
|                          <master>HWS1    </master>
|                          <userid>USRT011 </userid>
|                          <verb>UPD </verb>
|                          <kwd>IMSCON         </kwd>
|                          <input>UPD IMSCON TYPE(CONFIG) SET(ODRACFST(ON)) </input>
|                          </cmd>
|                          <cmdrsphdr>
|                          <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
|                           len="8" dtype="CHAR" align="left" skipb="no" />
|                          <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
|                           len="4" dtype="INT" align="right" skipb="no" />
|                          <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
|                           scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
|                          </cmdrsphdr>
|                          <cmdrspdata>
|                          <rsp>MBR(HWS1             ) CC(   0) </rsp>
|                          </cmdrspdata>
|                          </imsout>
```

| *Example 5 for UPDATE IMSCON TYPE(CONFIG) command*

| In the following example, the global-level IDLETO value is updated to 1000 or 10
| seconds.

| TSO SPOC input:

```
| UPDATE IMSCON TYPE(CONFIG) SET(IDLETO(1000))
```

| TSO SPOC output:

```
| MbrName           CC
| HWS1               0
```

| OM API input:

```
| CMD(UPDATE IMSCON TYPE(CONFIG) SET(IDLET0(1000)))
```

| OM API output:

```
| <cmd>
| <master>HWS1    </master>
| <userid>USRT002 </userid>
| <verb>UPD </verb>
| <kwd>IMSCON         </kwd>
| <input>UPDATE IMSCON TYPE(CONFIG) SET(IDLETO(1000)) </input>
| </cmd>
| <cmdrsphdr>
| <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
|  len="8" dtype="CHAR" align="left" skipb="no" />
| <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
|  len="4" dtype="INT" align="right" skipb="no" />
| <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
|  scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
| </cmdrsphdr>
| <cmdrspdata>
| <rsp>MBR(HWS1             ) CC(   0) </rsp>
| </cmdrspdata>
```

UPDATE IMSCON TYPE(CONVERTER) command

Use the UPDATE IMSCON TYPE(CONVERTER) command to refresh one or more IMS Connect XML converters.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 1109
- "Equivalent WTOR and z/OS commands" on page 1109
- "Output fields" on page 1109
- "Return, reason, and completion codes" on page 1110
- "Examples" on page 1111

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax
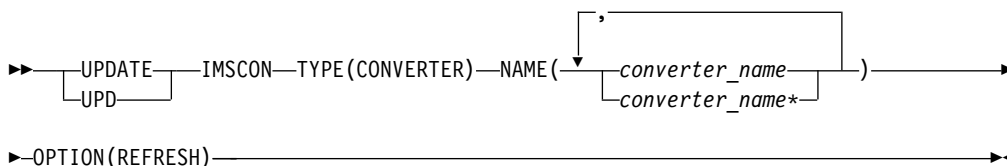
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(CONVERTER) command.

NAME

Specifies one or more XML converters to be refreshed. You can specify a single converter name or a list of converter names separated by commas. Wildcards can be used in the names.

OPTION(REFRESH)

Refreshes the specified XML converters.

Usage notes

You can issue the UPDATE IMSCON TYPE(CONVERTER) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(CONVERTER) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 423. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(CONVERTER) command

UPDATE IMSCON TYPE(CONVERTER) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(CONVERTER) NAME(*converter_name*) OPTION(REFRESH)	REFRESH CONVERTER NAME(*converter_name*)	UPDATE CONVERTER NAME(*converter_name*) OPTION(REFRESH)

Output fields

Short label

Contains the short label that is generated in the XML output.

Long label

Contains the column heading displayed on the TSO SPOC screen.

Keyword

Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning

Provides a brief description of the output field.

Table 424. Output fields for the UPDATE IMSCON TYPE(CONVERTER) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 426.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
CVTR	Converter	N/A	Name of the XML converter. The name is always returned.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(CONVERTER) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 425. Return and reason codes for the UPDATE IMSCON TYPE(CONVERTER) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(CONVERTER) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 426. Completion codes for the UPDATE IMSCON TYPE(CONVERTER) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(CONVERTER) command completed successfully for the resources.

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for UPDATE IMSCON TYPE(CONVERTER) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(CONVERTER) NAME(IMSPHBKD) OPTION(REFRESH)
```

TSO SPOC output:

```
Converter   MbrName CC
IMSPHBKD    HWS1     0
```

OM API input:

```
CMD(UPDATE IMSCON TYPE(CONVERTER) NAME(IMSPHBKD) OPTION(REFRESH))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 16:32:25.126708</statime>
<stotime>2010.298 16:32:25.140495</stotime>
<staseq>C6C83D24A4734B82</staseq>
<stoseq>C6C83D24A7D0F602</stoseq>
<rqsttkn1>USRT001 10093225</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(CONVERTER) NAME(IMSPHBKD) OPTION(REFRESH)
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="CVTR" llbl="Converter" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
```

```
<cmdrspdata>
<rsp>CVTR(IMSPHBKD) MBR(HWS1          ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The XML converter IMSPHBKD was successfully refreshed.

Related reference:

REFRESH CONVERTER command (Commands)

IMS Connect UPDATE CONVERTER command (Commands)

UPDATE IMSCON TYPE(DATASTORE) command

Use the UPDATE IMSCON TYPE(DATASTORE) command to start or stop communication between IMS Connect and an IMS OTMA data store, or to change data store attributes.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1113
- "Usage notes" on page 1116
- "Equivalent WTOR and z/OS commands" on page 1117
- "Output fields" on page 1117
- "Return and reason codes" on page 1118
- "Completion codes" on page 1119
- "Examples" on page 1122

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►──┬─UPDATE─┬──IMSCON──TYPE(DATASTORE)──NAME(──┬──┬─datastore_name──┬──┬──)──────►
    └─UPD────┘                                  └─ │ ┌─,──────────────       │
                                                    ▼─datastore_name*─┘
```

```
►──┬─START(COMM)──────────────────────────────────────────────────────────►◄
   ├─STOP(COMM)───────────────────────────────────────────┐
   │              ┌─────────────────,───────────────────┐  │
   └─SET(─▼─┬─ACKTO(to_value)─────────────────┐─────────┴──┘
           │                                  )
           ├─APPL(─┬─0──────────┬─)
           │       └─appl_value─┘
           ├─CASCADE(─┬─0───┬─)
           │          ├─OFF─┤
           │          └─ON──┘
           ├─CM0ATOQ(─┬─0──────────┬─)
           │          └─queue_name─┘
           ├─DRU(─┬─0────────┬─)
           │      └─dru_name─┘
           ├─GROUP(xcf_group_name)─
           ├─MAXI(max_value)─
           ├─MEMBER(xcf_member_name)─
           ├─MULTIRTP(─┬─0───┬─)
           │           ├─#───┤
           │           ├─OFF─┤
           │           └─ON──┘
           ├─OAAV(aging_value)─
           ├─RRNAME(─┬─0────────────┬─)
           │         └─reroute_name─┘
           ├─SMEMBER(─┬─0─────────────────┬─)
           │          └─super_member_name─┘
           └─TMEMBER(ims_xcf_member_name)─
```

Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(DATASTORE) command.

NAME

Specifies one or more data store names to be updated. You can specify a single data store name or a list of data store names that are separated by commas. Wildcards can be used in the names.

SET

Specifies the attribute values to be changed.

ACKTO

Specifies a numeric value of the timeout interval for acknowledgments to OTMA for CM0 and CM1 output messages, and for IMS-to-IMS transaction messages.

The timeout interval is in seconds. The range is 0 - 255. If the timeout value is 0, the OTMA ACK timeout default value of 120 seconds is set.

APPL

Specifies a 1- to 8-character alphanumeric name of the APPL name that is defined to RACF in the PTKTDATA statement.

If the character string "0" is specified, the APPL name is set to blanks.

CASCADE=

For this data store connection only, enables or disables support for the cascading of global, two-phase commit (synclevel=syncpoint) z/OS Resource Recovery Services transactions from IMS TM Resource Adapter

via a TCP/IP connection to an IMS TM or IMS DB/DC system that is on a different z/OS image (LPAR) than IMS Connect. Cascading transactions that are received via a LOCAL option connection is not supported.

The CASCADE parameter applies only to the support for cascading global, RRS transactions between IMS Connect and IMS TM when they are on different z/OS images. The specification of the IMS Connect CASCADE parameter does not affect support between other components or subsystems, such as between IMS subsystems in a shared queues environment.

The CASCADE specification in the definition of a data store connection overrides the value of CASCADE that is in effect for the IMS Connect system.

Until the next restart of IMS Connect, a CASCADE specification made by the UPDATE IMSCON TYPE(DATASTORE) SET(CASCADE()) command overrides the CASCADE specification on the DATASTORE configuration statement.

Before you can change the CASCADE specification in a data store connection definition, you must stop the data store connection in IMS Connect by issuing an UPDATE IMSCON TYPE(DATASTORE) STOP(COMM) command or a similar WTOR or z/OS MODIFY command.

0   When the numeric character "0" is specified, the CASCADE specification is removed from the definition of the data store connection. The data store connection inherits the CASCADE value that is in effect for the IMS Connect system configuration.

OFF
    Disables support for cascading of global, two-phase commit (synclevel=syncpoint) RRS transactions to an IMS TM or IMS DB/DC system that is running on a different LPAR than IMS Connect. If a synclevel=syncpoint transaction is routed to an IMS system that resides on a different LPAR than IMS Connect, IMS Connect rejects the transaction with return code 4 and reason code NCASCADE.

    When support is disabled, you can send a synclevel=syncpoint transaction to IMS on this data store connection only if IMS Connect and IMS reside on the same LPAR.

ON Enables support for cascading of global, two-phase commit (synclevel=syncpoint) RRS transactions to an IMS TM or IMS DB/DC system that is running on different LPAR than IMS Connect.

    IMS Connect detects whether the IMS system is on the same LPAR. If IMS Connect and IMS are running on different LPARs, two-phase commit processing uses cascaded transactions. If IMS Connect and IMS are running on the same LPAR, two-phase commit processing does not use cascaded transactions.

CM0ATOQ
    Specifies a 1-to 8-character alphanumeric name of the OTMA CM0 ACK timeout queue. This value overrides both the OTMA default value of DFS$$TOQ and any value that is set on the HWS statement of the IMS Connect configuration member.

    If the character string "0" is specified, the queue name is set to blanks.

DRU

Specifies a 1- to 8-character alphanumeric name of the OTMA destination resolution user exit that is to be passed to OTMA.

If the character string "0" is specified, the DRU name is set to blanks.

GROUP

Specifies the 1- to 8-character alphanumeric name of the z/OS cross-system coupling facility (XCF) group for the IMS OTMA. IMS Connect uses this value to join the XCF group.

Because IMS Connect and IMS must be in the same XCF group to communicate, this group name must match the XCF group name that you define to IMS (GRNAME) in the IMS startup JCL (for example, `"OTMA=Y,GRNAME=&GROUP,USERVAR=&MEMBER",...`).

MAXI

Specifies a numeric value of the OTMA input message flood control value.

The range is 0 - 9999. If you specify a value of 0, the OTMA default value of 5000 is used. If you specify a value 1-200, the OTMA minimum value of 200 is used.

MEMBER

Specifies the 1- to 16-character alphanumeric XCF member name that identifies IMS Connect in the XCF group that is specified by the GROUP parameter. This name is the XCF name that IMS uses to communicate with IMS Connect in that XCF group. This XCF member name for IMS Connect must be unique in the data store definitions for all data stores that are members of the same XCF group.

MULTIRTP

Specifies whether IMS Connect requests an OTMA tpipe that supports multiple active resume tpipe requests (MULTIRTP) when IMS Connect establishes DATASTORE connection with the IMS data store.

Any MULTIRTP value specified other than the character string "0" overrides the MULTIRTP value that is in effect for the IMS Connect system configuration. To accept the MULTIRTP value that is in effect for the IMS Connect system configuration, specify MULTIRTP(0).

The following values are valid for the MULTIRTP keyword:

0   When the numeric character "0" is specified, the MULTIRTP specification is removed from the definition of the data store connection. The data store connection inherits the MULTIRTP value that is in effect for the IMS Connect system configuration. If the IMS Connect system does not specify a MULTIRTP value, the OTMA client descriptor determines whether the tpipes for this data store connection support multiple active resume tpipe requests.

Specifies that the OTMA client descriptor determines the MULTIRTP value for this data store connection.

OFF

Disables support for multiple active resume tpipe requests on tpipes for this data store connection.

ON   Enables support for multiple active resume tpipe requests on tpipes for this data store connection.

OAAV

Specifies a numeric value of the OTMA accessor environment element

(ACEE) aging value. When the OTMA ACEE aging value is reached, OTMA refreshes the ACEE before it processes the next input message that is received from IMS Connect.

The OTMA ACEE aging value is in seconds. The range is 0 - 999999. If you specify 0, OTMA uses the default value of 999999. If you specify a value 1-300, OTMA uses a value of 300 seconds.

RRNAME
Specifies a 1-to 8-character alphanumeric name of the alternate destination of a client reroute request.

If the character string "0" is specified, the alternate destination name is set to the default setting of HWS$DEF.

SMEMBER
Specifies the 1-to 4-character name of the OTMA super member to which this IMS data store belongs. If specified, this value overrides the attribute values for the SMEMBER parameter in the HWS statement of the HWSCFGxx member of the IMS PROCLIB data set.

If the character string "0" is specified, the value of SMEMBER specified on this IMS data store is set to blanks, and the super member function is disabled for this data store.

TMEMBER
Specifies the 1- to 16-character alphanumeric XCF member name of the IMS that this IMS Connect communicates with in the XCF group. This target member name must match the member name that IMS uses when it joins the XCF group. The XCF member name for IMS is specified in the IMS startup JCL (for example, `"...,OTMA=Y,GRNAME=&GROUP,OTMANM=&TMEMBER,...").

START(COMM) | STOP(COMM)
Mutually exclusive keywords that you use to start or stop communication with the data stores specified.

START(COMM)
Starts communication with the data stores specified.

STOP(COMM)
Stops communication with the data stores specified.

Usage notes

You can issue the UPDATE IMSCON TYPE(DATASTORE) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Updates that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect, unless the updates are also made to the corresponding configuration statement in the HWSCFGxx member of the IMS PROCLIB data set before the next restart.

All attributes except for APPL and RRNAME require the data store to be stopped (in NOTACTIVE state) before they can be updated with the UPDATE IMSCON TYPE(DATASTORE) command.

| Attributes that are specified on the SET() keyword of the UPDATE IMSCON
| TYPE(DATASTORE) command override any specification for that attribute in the
| DATASTORE statement.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and
IMS Connect z/OS commands that perform similar functions as the UPDATE
IMSCON TYPE(DATASTORE) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect
  reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by
  using the IMS Connect *jobname*.

*Table 427. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON
TYPE(DATASTORE) command.*

UPDATE IMSCON TYPE(DATASTORE) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(DATASTORE) NAME(*datastore_name*) START(COMM)	OPENDS *datastore_id* STARTDS *datastore_id*	UPDATE DATASTORE NAME(*datastoreName*) START(COMM)
UPDATE IMSCON TYPE(DATASTORE) NAME(*datastore_name*) STOP(COMM)	STOPDS *datastore_id*	UPDATE DATASTORE NAME(*datastoreName*) STOP(COMM)
UPDATE IMSCON TYPE(DATASTORE) NAME(*datastore_name*) SET()	None	None

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be
generated. N/A (not applicable) appears for output fields that are always
returned. *error* appears for output fields that are returned only in the case
of an error.

Meaning
Provides a brief description of the output field.

Table 428. Output fields for the UPDATE IMSCON TYPE(DATASTORE) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See "Completion codes" on page 1119.

Table 428. Output fields for the UPDATE IMSCON TYPE(DATASTORE) command (continued)

Short label	Long label	Keyword	Meaning
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
DS	DataStore	N/A	The data store name. The data store name is always returned.
ERRT	ErrorText	*error*	Error text that provides diagnostic information. Error text can be returned for a nonzero completion code and further explains the completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.

Return and reason codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(DATASTORE) command are standard for all commands that are entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 429. Return and reason codes for the UPDATE IMSCON TYPE(DATASTORE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(DATASTORE) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'00000008'	X'00002044'	The command was rejected because it contained both START() and STOP() keywords. Only one of these keywords is allowed.
X'00000008'	X'00002094'	The RRNAME parameter value contains an invalid character.
X'00000008'	X'0000209C'	The SET() keyword cannot be used with the keywords START() or STOP().
X'00000008'	X'000020A0'	The MEMBER parameter value contains an invalid character.
X'00000008'	X'000020A4'	The GROUP parameter value contains an invalid character.
X'00000008'	X'000020A8'	The SMEMBER parameter value contains an invalid character.
X'00000008'	X'000020AC'	The TMEMBER parameter value contains an invalid character.
X'00000008'	X'000020B0'	The APPL parameter value contains an invalid character.
X'00000008'	X'000020B4'	The CM0ATOQ parameter value contains an invalid character.

Table 429. Return and reason codes for the UPDATE IMSCON TYPE(DATASTORE) command (continued)

Return code	Reason code	Meaning
X'00000008'	X'000020B8'	The DRU parameter value contains an invalid character.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, which is accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, which is accompanied by its completion code.

Completion codes

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 430. Completion codes for the UPDATE IMSCON TYPE(DATASTORE) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(DATASTORE) command completed successfully for the resources.
3	Some attributes not updated	The command was successful for some attributes. There is a separate line of output with its own completion code for each attribute that IMS Connect failed to set.
9	Data store table create failed	During START(COMM) processing, IMS Connect could not create a data store table (DST) control block. For more information, see the HWSD0212E message that is included in the command reply message.
A	TRWB creation failed	The command attempted to create a TWU control block for the scheduler controller thread, the transmit thread, or the receive thread for the data store, but the BPECBGET request failed. The command output might also contain one of the following messages: • HWSD0212E • HWSD0222W
B	Data store table enqueue failed	During START(COMM) processing, IMS Connect attempted to enqueue a DST control block to the active queue for the data store, but the request failed. For more information, see the HWSD0212E message that is included in the command reply message.

Table 430. Completion codes for the UPDATE IMSCON TYPE(DATASTORE) command (continued)

Completion code	Completion code text	Meaning
C	Create scheduler thread failed	During START(COMM) processing, IMS Connect attempted to create the scheduler controller thread for the data store, but the BPETHDCR request failed.
		For more information, see the HWSD0212E message that is included in the command reply message.
D	Data store is in shutdown	IMS Connect could not start the data store because the data store was shutting down.
		For more information, see the HWSD0222W message that is included in the command reply message.
E	Data store block create failed	During START(COMM) processing, IMS Connect could not start the data store because an attempt to create a data store block (DSB) for either the transmit or receive thread for the data store failed.
		For more information, see the HWSD0222W message that is included in the command reply message.
F	Create xmit/recv thread failed	During START(COMM) processing, IMS Connect attempted to create either the transmit or receive thread for the data store, but the BPETHDCR request failed.
		For more information, see the HWSD0222W message that is included in the command reply message.
10	No resources found	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active now. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
11	Resource is stopped	During STOP(COMM) processing, IMS Connect could not stop a data store because the data store is already stopped.
12	Resource is not stopped	During START(COMM) processing, IMS Connect could not start a data store because the data store is not stopped.
1A	BPEGETM error	During START(COMM) processing, IMS Connect failed to obtain storage from the BPEGETM service.
		The command output might also contain the following message: HWSO1205W.

Table 430. Completion codes for the UPDATE IMSCON TYPE(DATASTORE) command (continued)

Completion code	Completion code text	Meaning
25	IXCQUERY call failed	During START(COMM) processing, the call to IXCQUERY to query the XCF group member failed. IMS Connect will still try to join the XCF group. If IMS Connect is able to join the XCF group, it waits for a notification that indicates that the member it is waiting for has also joined the XCF group. When the XCF group member also joins, IMS Connect proceeds with connection to the data store.
		The command output might also contain one of the following messages:
		• HWSO1210W
		• HWSD0292I
26	IXCJOIN call failed	During START(COMM) processing, IMS Connect encountered an error when it attempted to join the XCF group through a call to IXCJOIN.
		The command output might also contain the following message: HWSO1220W.
27	Client BID process failed	During START(COMM) processing, IMS Connect encountered an error when it performed the client BID process with OTMA.
		The command output might also contain one of the following messages:
		• HWSO1310W
		• HWSO1320W
28	IXCLEAVE call failed	During START(COMM) processing, IMS Connect encountered an error when it attempted to leave the XCF group through a call to IXCLEAVE.
		The command output might also contain the following message: HWSO1315W.
29	XCF group member not active	During START(COMM) processing, IMS Connect detected that the XCF group member is not active. IMS Connect will still try to join the XCF group. If IMS Connect is able to join the XCF group, it waits for a notification that indicates that the member that it is waiting for has also joined the XCF group. When the XCF group member also joins, IMS Connect proceeds with connection to the data store.
		The command output might also contain the following message: HWSO1215W.
2A	C512 block creation failed	During START(COMM) processing, IMS Connect encountered an error while it attempted to obtain a block of storage of type C512 from the BPECBGET service.
		The command output might also contain the following message: HWSO1305W.

Table 430. Completion codes for the UPDATE IMSCON TYPE(DATASTORE) command (continued)

Completion code	Completion code text	Meaning
2B	RACF UTOKEN extraction failed	During START(COMM) processing, IMS Connect encountered an error while it extracted the UTOKEN for this instance of IMS Connect.
		The command output might also contain the following message: HWSO1325W.
4E	MEMBER/group combo is duplicate	The MEMBER parameter value specified is a duplicate of an existing data store MEMBER value.
4F	Member/GROUP combo is duplicate	The GROUP parameter value specified and the current MEMBER value of this data store is a duplicate of an existing data store definition for the specified XCF group.

Examples

Example 1 for UPDATE IMSCON TYPE(DATASTORE) command

TSO SPOC input:
```
UPDATE IMSCON TYPE(DATASTORE) NAME(IMS1) START(COMM)
```

TSO SPOC output:
```
DataStore  MbrName  CC
IMS1       HWS1      0
```

OM API input:
```
CMD(UPDATE IMSCON TYPE(DATASTORE) NAME(IMS1) START(COMM))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 02:34:53.534886</statime>
<stotime>2010.298 02:34:53.536353</stotime>
<staseq>C6C781F0FD6A67A5</staseq>
<stoseq>C6C781F0FDC61CA5</stoseq>
<rqsttkn1>USRT001 10193453</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1     </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(DATASTORE) NAME(IMS1) START(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
```

```
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DS(IMS1    ) MBR(HWS1             ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Communication between IMS Connect, HWS1, and the data store
IMS1 has been started.

Example 2 for UPDATE IMSCON TYPE(DATASTORE) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(DATASTORE) NAME(IMS1)
SET(ACKTO(100),CM0ATOQ(TIMEOUT1))
```

TSO SPOC output:

```
DataStore MbrName         CC
IMS1      HWS1             0
```

OM API input:

```
CMD(UPD IMSCON TYPE(DATASTORE) NAME(IMS1)
SET(ACKTO(100),CM0ATOQ(TIMEOUT1)))
```

OM API output:

```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2014.232 22:21:09.203929</statime>
<stotime>2014.232 22:21:09.204176</stotime>
<staseq>CDA274E167BD9188</staseq>
<stoseq>CDA274E167CD0188</stoseq>
<rqsttkn1>USRT002 10152109</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT002 </userid>
<verb>UPD </verb>
<kwd>IMSCON        </kwd>
<input>UPDATE IMSCON TYPE(DATASTORE)
 NAME(IMS1)SET(ACKTO(100),CM0ATOQ(TIMEOUT1)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes" len="4"
 dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0" scroll="yes"
 len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="a" key="3" scroll="yes"
 len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
```

```
<cmdrspdata>
<rsp>DS(IMS1     ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: For communications to the data store IMS1, the ACKTO attribute is set to 100 and the CM0ATOQ attribute is set to TIMEOUT1.

Example 3 for UPDATE IMSCON TYPE(DATASTORE) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(DATASTORE) NAME(IMS*)
SET(ACKTO(120),MAXI(6000))
```

TSO SPOC output:

DataStore	MbrName	CC	CCText	ErrorText
IMS1	HWS1	0		
IMS2	HWS1	14	Resource is not stopped	ACKTO
IMS2	HWS1	14	Resource is not stopped	MAXI

OM API input:

```
CMD(UPD IMSCON TYPE(DATASTORE) NAME(IMS*)
SET(ACKTO(120),MAXI(6000)))
```

OM API output:

```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2014.232 22:16:11.548664</statime>
<stotime>2014.232 22:16:11.548920</stotime>
<staseq>CDA273C589FF8082</staseq>
<stoseq>CDA273C58A0F8882</stoseq>
<rqsttkn1>USRT002 10151611</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="HWS1    ">
<typ>IMSCON  </typ>
<rc>0C00000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful   </rsntxt>
</mbr>
</cmderr>
<cmd>
<master>HWS1    </master>
<userid>USRT002 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPDATE IMSCON TYPE(DATASTORE) NAME(IMS*)
 SET(ACKTO(120),MAXI(6000)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DS" llbl="DataStore" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
```

```
          <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes" len="4"
           dtype="INT" align="right" skipb="no" />
          <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0" scroll="yes"
           len="32" dtype="CHAR" align="left" skipb="yes" />
          <hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="a" key="3" scroll="yes"
           len="16" dtype="CHAR" align="left" skipb="yes" />
        </cmdrsphdr>
        <cmdrspdata>
        <rsp>DS(IMS1    ) MBR(HWS1            ) CC(   0) </rsp>
        <rsp>DS(IMS2    ) MBR(HWS1            ) CC(  14) CCTXT(Resource is not stopped)
         ERRT(ACKTO          ) </rsp>
        <rsp>DS(IMS2    ) MBR(HWS1            ) CC(  14) CCTXT(Resource is not stopped)
         ERRT(MAXI           ) </rsp>
        </cmdrspdata>
        </imsout>
```

Explanation: The command is attempting to set the ACKTO and MAXI attributes
for all data stores that start with the characters "IMS". For data store IMS1, the
ACKTO attribute is set to 120 and the MAXI attribute is set to 6000. For data store
IMS2, the updates to the attributes failed because the resource data store IMS2 is
not stopped. There is a line of output for each attribute that IMS Connect failed to
set for each IMS data store.

Example 4 for UPDATE IMSCON TYPE(DATASTORE) command

TSO SPOC input:
```
UPDATE IMSCON TYPE(DATASTORE) NAME(IMS1)
SET(MEMBER(HWS2),SMEMBER(0))
```

TSO SPOC output:
```
DataStore MbrName            CC CCText                         ErrorText
IMS1      HWS1                3 Some attributes not updated
IMS1      HWS1               4E Duplicate XCF member name      HWS2
```

OM API input:
```
CMD(UPDATE IMSCON TYPE(DATASTORE) NAME(IMS1)
SET(MEMBER(HWS2),SMEMBER(0)))
```

OM API output:
```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2014.232 22:38:47.819937</statime>
<stotime>2014.232 22:38:47.820244</stotime>
<staseq>CDA278D2FAEA1192</staseq>
<stoseq>CDA278D2FAFD4792</stoseq>
<rqsttkn1>USRT002 10153847</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="HWS1    ">
<typ>IMSCON </typ>
<rc>0C00000C</rc>
<rsn>00003004</rsn>
<rsntxt>No requests were successful   </rsntxt>
```

```
|         </mbr>
|         </cmderr>
|         <cmd>
|         <master>HWS1    </master>
|         <userid>USRT002 </userid>
|         <verb>UPD </verb>
|         <kwd>IMSCON         </kwd>
|         <input>UPDATE IMSCON TYPE(DATASTORE) NAME(IMS1)
|           SET(MEMBER(HWS2),SMEMBER(0)) </input>
|         </cmd>
|         <cmdrsphdr>
|         <hdr slbl="DS" llbl="DataStore" scope="LCL" sort="a" key="1" scroll="no"
|          len="8" dtype="CHAR" align="left" skipb="no" />
|         <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
|          len="8" dtype="CHAR" align="left" skipb="no" />
|         <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
|          len="4" dtype="INT" align="right" skipb="no" />
|         <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0" scroll="yes"
|          len="32" dtype="CHAR" align="left" skipb="yes" />
|         <hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="a" key="3" scroll="yes"
|          len="16" dtype="CHAR" align="left" skipb="yes" />
|         </cmdrsphdr>
|         <cmdrspdata>
|         <rsp>DS(IMS1    ) MBR(HWS1            ) CC(  4E) CCTXT(Duplicate XCF member name)
|          ERRT(HWS2            ) </rsp>
|         <rsp>DS(IMS1    ) MBR(HWS1            ) CC(   3) CCTXT(Some attributes not updated)
|          </rsp>
|         </cmdrspdata>
|         </imsout>
```

| **Explanation:** The command is attempting to set the MEMBER and SMEMBER
| attributes for data store IMS1. Some updates were successful. A separate CC is
| displayed for each attribute update that was not successful. IMS Connect detected
| that another data store in the same XCF group as IMS1 has the same member
| name, HWS2, that was specified in MEMBER(), so MEMBER was not set. The
| SMEMBER attribute was set successfully because there was no CC for it.

Related reference:

➡️ OPENDS command (Commands)

➡️ STARTDS command (Commands)

➡️ STOPDS command (Commands)

➡️ IMS Connect UPDATE DATASTORE command (Commands)

UPDATE IMSCON TYPE(IMSPLEX) command

Use the UPDATE IMSCON TYPE(IMSPLEX) command to start or stop
communication between IMS Connect and the specified IMSplex.

Subsections:

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►──┬─UPDATE─┬──IMSCON─TYPE(IMSPLEX)─NAME(imsplex_name)──┬─START(COMM)─┬──►◄
    └─UPD────┘                                           └─STOP(COMM)──┘
```

Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(IMSPLEX) command.

NAME
> Specifies the name of the IMSplex. This name must be defined to the IMS Connect through the configuration member HWSCFGxx. In addition, this name must match the TMEMBER that is defined in the IMSplex configuration statement.

START(COMM) | STOP(COMM)
> Mutually exclusive keywords that you use to start or stop communication with the IMSplex.

> **START(COMM)**
>> Starts communication with the IMSplex. Use this command to reestablish communication with the IMSplex that is communicating with OM or ODBM if communication between IMS Connect and the IMSplex fails. For example, use this command to restart communication when all activities for the IMSplex in the IMS Connect are terminated, or after an UPDATE IMSCON TYPE(IMSPLEX) STOP(COMM) command terminated communication with the IMSplex.

>> Type-2 commands require that communication between IMS Connect and the IMSplex of which OM is a member is already established. Therefore, the UPDATE IMSCON TYPE(IMSPLEX) START(COMM) command can be used only to start communication between IMS Connect and a different IMSplex. To start communication between IMS Connect and the IMSplex that is used for type-2 command support, use either the WTOR command STARTIP or OPENIP, or the z/OS command UPDATE IMSPLEX START(COMM).

> **STOP(COMM)**
>> Stops communication with the IMSplex. Work currently in progress for the IMSplex is ended and communication with the IMSplex and its threads are terminated. This option can also be used for any error situation that requires immediate termination of communication with the IMSplex.

>> **Note:** If you stop communication with the IMSplex that is used for type-2 commands, you will not be able to issue further commands to IMS Connect through OM because the connection to SCI is terminated. To

restart communication with the IMSplex, use either the WTOR command STARTIP or OPENIP, or the z/OS command UPDATE IMSPLEX START(COMM).

Usage notes

You can issue the UPDATE IMSCON TYPE(IMSPLEX) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

When the UPDATE IMSCON TYPE(IMSPLEX) NAME(*imsplexName*) STOP(COMM) is issued, work currently in progress for the IMSplex ends and communication with the IMSplex and its threads are terminated. Any messages in progress are rejected and an error message is returned to the requester. STOPIP can also be used for any error situation that requires immediate termination of communication with the IMSplex.

If IMS Connect supports ISC TCP/IP links and the STOP(COMM) keyword is used, IMS Connect sends a notification to IMS for each ISC link to inform IMS that all of the ISC parallel sessions are terminated on the ISC link.

If IMS Connect supports MSC IMS-to-IMS TCP/IP connections and the STOP(COMM) keyword is used, IMS Connect sends a notification to IMS for each MSC physical link to inform IMS that all of the MSC logical links are terminated on the physical link.

Use the START(COMM) keyword to start communication with the IMSplex at a later time.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 431. Output fields for the UPDATE IMSCON TYPE(IMSPLEX) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 433 on page 1129.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.

Table 431. Output fields for the UPDATE IMSCON TYPE(IMSPLEX) command (continued)

Short label	Long label	Keyword	Meaning
IMSPLX	IMSplex	N/A	The IMSplex name. The IMSplex name is always returned.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(IMSPLEX) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 432. Return and reason codes for the UPDATE IMSCON TYPE(IMSPLEX) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(IMSPLEX) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 433. Completion codes for the UPDATE IMSCON TYPE(IMSPLEX) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(IMSPLEX) command completed successfully for the resources.
9	Data store table create failed	During START(COMM) processing, IMS Connect could not create a data store table (DST) control block for communication with the IMSplex. For more information, see the HWSD0212E message that is included in the command reply message.

*Table 433. Completion codes for the UPDATE IMSCON TYPE(IMSPLEX)
command (continued)*

Completion code	Completion code text	Meaning
A	TRWB creation failed	The command attempted to create a TWU control block for the scheduler controller thread, the transmit thread, or the receive thread for communication with the IMSplex, but the BPECBGET request failed. The command output might also contain one of the following messages: • HWSD0212E • HWSM0522W
B	Data store table enqueue failed	During START(COMM) processing, IMS Connect attempted to enqueue a DST control block to the active queue for communication with the IMSplex, but the request failed. For more information, see the HWSD0212E message that is included in the command reply message.
C	Create scheduler thread failed	During START(COMM) processing, IMS Connect attempted to create the scheduler controller thread for communication with the IMSplex, but the BPETHDCR request failed. For more information, see the HWSD0212E message that is included in the command reply message.
D	Data store is in shutdown	IMS Connect could not start communication with the IMSplex because communication with the IMSplex was shutting down. For more information, see the HWSM0522W message that is included in the command reply message.
E	Data store block create failed	During START(COMM) processing, IMS Connect could not start communication with the IMSplex because an attempt to create a data store block (DSB) for either the transmit or receive thread for communication with the IMSplex failed. For more information, see the HWSM0522W message that is included in the command reply message.
F	Create xmit/recv thread failed	During START(COMM) processing, IMS Connect attempted to create either the transmit or receive thread for communication with the IMSplex, but the BPETHDCR request failed. For more information, see the HWSM0522W message that is included in the command reply message.
10	No resources found	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Table 433. Completion codes for the UPDATE IMSCON TYPE(IMSPLEX) command (continued)

Completion code	Completion code text	Meaning
11	Resource is stopped	During STOP(COMM) processing,IMS Connect could not stop communication with the IMSplex because communication with the IMSplex is already stopped.
12	Resource is not stopped	During START(COMM) processing, IMS Connect could not start communication with the IMSplex because communication with the IMSplex is not stopped.
1A	BPEGETM error	During START(COMM) processing, IMS Connect failed to obtain storage from the BPEGETM service. The command output might also contain the following message: HWSI1705W.
2C	IMS Connect is waiting for SCI	During START(COMM) processing, IMS Connect has detected that SCI is not available for communication with the IMSplex. IMS Connect is waiting for SCI to become available because ODBMAUTOCONN=Y was specified in the ODACCESS statement of the HWSCFGxx member of the IMS PROCLIB data set. The command output might also contain the following message: HWSX0940A.
2D	Deregistration from SCI failed	During START(COMM) processing, IMS Connect encountered an error when it attempted to deregister from SCI for communication with the IMSplex. The command output might also contain the following message: HWSI1815W.
2E	Registration to SCI failed	During START(COMM) processing, IMS Connect encountered an error when it attempted to register with SCI for communication with the IMSplex. The command output might also contain the following message: HWSI1720W.
2F	SCI is not available	During START(COMM) processing, IMS Connect has detected that SCI is not available for communication with the IMSplex. The command output might also contain the following message: HWSI1816W.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(IMSPLEX) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 434. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(IMSPLEX) command.

UPDATE IMSCON TYPE(IMSPLEX) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(IMSPLEX) NAME(*imsplex_name*) START(COMM)	OPENIP *imsplex_id*	UPDATE IMSPLEX NAME(*imsplex_name*) START(COMM)
	STARTIP *imsplex_id*	
UPDATE IMSCON TYPE(IMSPLEX) NAME(*imsplex_name*) STOP(COMM)	STOPIP *imsplex_id*	UPDATE IMSPLEX NAME(*imsplex_name*) STOP(COMM)

Examples

Example 1 for UPDATE IMSCON TYPE(IMSPLEX) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(IMSPLEX) NAME(PLEX1) STOP(COMM)
```

TSO SPOC output:

```
MbrName    IMSplex  CC
HWS1       PLEX1     0
```

OM API input:

```
CMD(UPDATE IMSCON TYPE(IMSPLEX) NAME(PLEX1) STOP(COMM))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 02:39:37.778072</statime>
<stotime>2010.298 02:39:37.787961</stotime>
<staseq>C6C7830010B98787</staseq>
<stoseq>C6C7830013239087</stoseq>
<rqsttkn1>USRT001 10193937</rqsttkn1>
<rc>02000004</rc>
<rsn>00001014</rsn>
<rsnmsg>CSLN055I</rsnmsg>
<rsntxt>At least one request completed with warning(s).</rsntxt>
</ctl>
<cmderr>
<mbr name="HWS1     ">
<typ>IMSCON   </typ>
<rc>00000000</rc>
<rsn>00000000</rsn>
</mbr>
</cmderr>
<cmd>
<master>HWS1     </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(IMSPLEX) NAME(PLEX1) STOP(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="IMSPLX" llbl="IMSplex" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
```

```
  len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>IMSPLX(PLEX1   ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Communication between IMS Connect, HWS1, and the IMSplex, PLEX1, has been stopped.

Related reference:

➡ OPENIP command (Commands)

➡ STARTIP command (Commands)

➡ STOPIP command (Commands)

➡ IMS Connect UPDATE IMSPLEX command (Commands)

UPDATE IMSCON TYPE(ISC) command

Use the UPDATE IMSCON TYPE(ISC) command to stop or restart communications on an ISC link.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1134
- "Usage notes" on page 1134
- "Equivalent WTOR and z/OS commands" on page 1134
- "Output fields" on page 1135
- "Return, reason, and completion codes" on page 1135
- "Examples" on page 1138

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►──┬─UPDATE─┬──IMSCON──TYPE(ISC)──NAME(──┬─▼─isc_stmt_id──┬──)──────────►
    └─UPD────┘                            └─isc_stmt_id*───┘
```

```
►►─┬─START(COMM)─┬──────────────────────────────────────────────────────────►◄
   └─STOP(COMM)──┘
```
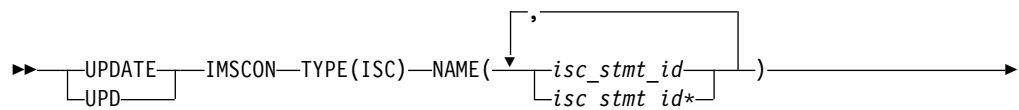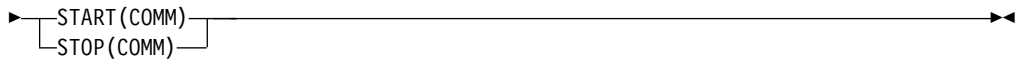
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(ISC) command.

NAME
> Specifies the ID of an ISC statement that defines an ISC link. You can specify a single ISC link ID or a list of ISC link IDs separated by commas. Wildcards can be used in the names.

START(COMM)
> Starts or restarts the ISC communications on the ISC link after it has been terminated. The status of the ISC link is set to "ACTIVE" after the completion of the command. START(COMM) is mutually exclusive with STOP(COMM).

STOP(COMM)
> Terminates all communications on an ISC link. The status of the ISC link is set to "NOTACTIVE" after the completion of the command. STOP(COMM) is mutually exclusive with START(COMM).

Usage notes

Recommendation: Do not issue the UPDATE IMSCON TYPE(ISC) command if any ISC sessions (ISCUSERs) are active on the ISC link. Any active sessions should be shut down from IMS by using the /QUIESCE or /CLSDST command before issuing this command. If the sessions are terminated in IMS, but still appear active in IMS Connect, you can issue the UPDATE IMSCON TYPE(ISC) command to clean them up.

You can issue the UPDATE IMSCON TYPE(ISC) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

When the UPDATE IMSCON TYPE(ISC) STOP(COMM) command is issued, IMS Connect:
- Stops active parallel sessions and prevents further communication on the specified ISC link
- Informs IMS that communication has stopped on the ISC link
- Changes the status of the ISC link to NOT ACTIVE
- Issues message HWSG4005I

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(ISC) command.

Notes:

- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 435. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(ISC) command

UPDATE IMSCON TYPE(ISC) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(ISC) NAME(*isc_stmt_id*) START(COMM)	None	None
UPDATE IMSCON TYPE(ISC) NAME(*isc_stmt_id*) STOP(COMM)	None	None

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
Provides a brief description of the output field.

Table 436. Output fields for the UPDATE IMSCON TYPE(ISC) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 438 on page 1136.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
ISC	ISCName	N/A	The name of the ISC link, as it is defined on the ID keyword of the ISC statement in the IMS Connect configuration member. The ISC name is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(ISC) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 437. Return and reason codes for the UPDATE IMSCON TYPE(ISC) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(ISC) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C000008'	X'00002004'	An invalid keyword or keyword parameter was specified.
X'0C000008'	X'00002014'	An invalid character was specified in the NAME() parameter.
X'0C000008'	X'00002018'	The NAME keyword was not specified, but is a required keyword for the UPDATE IMSCON TYPE(ISC) command.
X'0C000008'	X'00002020'	Neither the STOP keyword nor the START keyword was specified. One of these must be specified.
X'0C000008'	X'00002044'	The STOP and START keywords were both specified in the command. Only one of these may be specified.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005000'	The command processor failed to obtain storage via BPECBGET.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 438. Completion codes for the UPDATE IMSCON TYPE(ISC) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(ISC) command completed successfully for the resource.
8	Command processing error	The command could not start the ISC link because the ISC open/close controller is not active. This error is possible during early IMS Connect initialization or late in IMS Connect termination.
9	Datastore table create failed	During START(COMM) processing, IMS Connect could not create a datastore table (DST) control block. Refer to the HWSD0212E message that is included in the command reply message for more information.
A	TRWB creation failed	The command attempted to create a TWU control block for the scheduler controller thread, the transmit thread, or the receive thread for the ISC link, but the BPECBGET request failed. Refer to the HWSD0212E or HWSD0222W message that is included in the command reply message for more information.

Completion code	Completion code text	Meaning
B	Datastore table enqueue failed	During START(COMM) processing, IMS Connect attempted to enqueue a DST control block to the active queue for the ISC link, but the request failed. Refer to the HWSD0212E message that is included in the command reply message for more information.
C	Create scheduler thread failed	During START(COMM) processing, IMS Connect attempted to create the scheduler controller thread for the ISC link, but the BPETHDCR request failed. Refer to the HWSD0212E message that is included in the command reply message for more information.
D	Datastore is in shutdown	IMS Connect could not start the ISC link because the ISC link was shutting down. Refer to the HWSD0222W message that is included in the command reply message for more information.
E	Datastore block create failed	During START(COMM) processing, IMS Connect could not start the ISC link because an attempt to create a DSB control block for either the transmit or receive thread for the ISC link failed. Refer to the HWSD0222W message that is included in the command reply message for more information.
F	Create xmit/recv thread failed	During START(COMM) processing, IMS Connect attempted to create either the transmit or receive thread for the ISC link, but the BPETHDCR request failed. Refer to the HWSD0222W message that is included in the command reply message for more information.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
11	Resource is stopped	During STOP(COMM) processing, IMS Connect could not stop an ISC link because the link is already stopped.
12	Resource is not stopped	During START(COMM) processing, IMS Connect could not start an ISC link because the link is not stopped.
1A	BPEGETM error	During START(COMM) processing, IMS Connect failed to obtain storage from the BPEGETM service. Refer to the HWSG4230E message that is included in the command reply message for more information.
1B	IMSPLEX resource is stopped	During START(COMM) processing, IMS Connect was unable to start the ISC resource because the associated IMSplex resource is stopped in IMS Connect. The IMSplex can be started with the IMS Connect WTO command STARTIP or the equivalent IMS Connect z/OS MODIFY or IMS type-2 command. Refer to the HWSG4015E message that is included in the command reply message for more information.

Examples

Example 1 for UPDATE IMSCON TYPE(ISC) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(ISC) NAME(LU6TCP1) STOP(COMM)
```

TSO SPOC output:

```
ISCName    MbrName   CC
LU6TCP1    HWS1       0
```

OM API input:

```
CMD(UPDATE IMSCON TYPE(ISC) NAME(LU6TCP1) STOP(COMM))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2012.034 22:40:43.734518</statime>
<stotime>2012.034 22:40:43.736309</stotime>
<staseq>C9127498505F6590</staseq>
<stoseq>C912749850CF5390</stoseq>
<rqsttkn1>USRT001 10144043</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1     </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(ISC) NAME(ISC12XX) STOP(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="a" key="1"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>ISC(LU6TCP1 ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The communication between IMS Connect, HWS1, and ISC LU6TCP1 has been stopped.

Related reference:

➡ ISC statement (System Definition)

➡ RMTCICS statement (System Definition)

UPDATE IMSCON TYPE(ISCUSER) command

Use the UPDATE IMSCON TYPE(ISCUSER) command to stop an ISC user session (ISCUSER) that is assigned to an ISC link in IMS Connect.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 1140
- "Equivalent WTOR and z/OS commands" on page 1140
- "Output fields" on page 1141
- "Return, reason, and completion codes" on page 1141
- "Example" on page 1142

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►─┬─UPDATE─┬─IMSCON─TYPE(ISCUSER)─NAME(─┬─◄─┬─isc_user_name──┬─)─────────►
   └─UPD────┘                             │      └─isc_user_name*─┘
                                          └──────,──────┘

►─STOP(COMM)──────────────────────────────────────────────────────────►◄
          └─ISC(─┬─◄─┬─isc_link_id──┬─)─┘
                 │      └─isc_link_id*─┘
                 └──────,──────┘
```
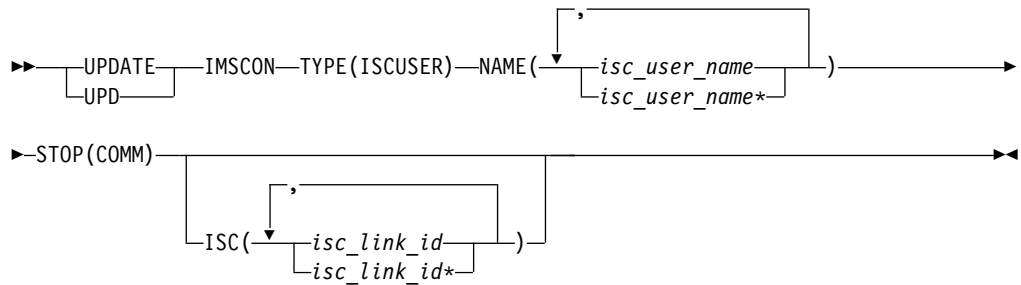
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(ISCUSER) command.

ISC

> Specifies the ISC link ID defined in the ID parameter of the ISC statement. You can specify a single ISC link ID or a list of ISC link IDs separated by commas. Wildcards can be used in the names.
>
> ISC is an optional keyword. If specified, the command applies only to those user sessions that are associated with the specified link ID. If omitted, then the command applies to user sessions that are associated with any link ID.

NAME
> Specifies the ISC user session name. You can specify a single ISC user name or a list of ISC user names that are separated by commas. Wildcards can be used in the names.

STOP(COMM)
> Stops communication for the ISC user.

Usage notes

You can issue the UPDATE IMSCON TYPE(ISCUSER) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

Use the UPDATE IMSCON TYPE(ISCUSER) STOP(COMM) command to clean up IMS Connect resources that are associated with an ISC user session when the user session was already terminated, but the IMS Connect resources that are associated with the user session were not cleaned up correctly.

Recommendation: Use the IMS command /QUIESCE NODE to terminate ISC user sessions. Use the UPDATE IMSCON TYPE(ISCUSER) command only when the IMS Connect resources that are associated with an ISC user session that was terminated failed to clean up correctly.

When the UPDATE IMSCON TYPE(ISCUSER) STOP(COMM) command is issued, IMS Connect:
- Stops communication on the ISC user session
- Informs IMS that communication is stopped on the user session
- Deletes the control blocks associated with the user session and frees the associated storage
- Issues message HWSG4010I

To avoid accidentally stopping a user session that uses the same user session name on another ISC link, limit the command processing to a specific ISC link by specifying the ID of the target ISC link on the **ISC** keyword.

If the UPDATE IMSCON TYPE(ISCUSER) command is issued against two or more ISC links, or the *isc_link_id* value is omitted, IMS Connect stops all of the ISC user sessions that match the specified ISC user names on each ISC link.

To display information about the ISC user sessions that are assigned to the ISC links that are defined to an IMS Connect instance, use either of the IMS type-2 commands QUERY IMSCON TYPE(ISCUSER) or QUERY IMSCON TYPE(ISC).

Equivalent WTOR and z/OS commands

The UPDATE IMSCON TYPE(ISCUSER) command does not have an equivalent IMS Connect WTOR (Write to Operator with Reply) or IMS Connect z/OS command.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading that is displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) is displayed for output fields that are always returned. *error* is displayed for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 439. Output fields for the UPDATE IMSCON TYPE(ISCUSER) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 441 on page 1142.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. The CCText field is returned only for an error completion code.
IUSER	ISCUser	N/A	Specifies the ISC user session that was terminated. The identifier is always returned.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
ISC	ISCName	N/A	Identifier of the ISC link that is associated with the user session that was terminated. The identifier, which is defined on the ID parameter of the ISC statement, is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(ISCUSER) command are standard for all commands that are entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 440. Return and reason codes for the UPDATE IMSCON TYPE(ISCUSER) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(ISCUSER) command completed successfully. The command output contains a line for each resource, which is accompanied by its completion code.
X'0C000008'	X'00002004'	An invalid keyword or keyword parameter was specified.
X'0C000008'	X'00002014'	An invalid character was specified in the NAME() parameter.

Table 440. Return and reason codes for the UPDATE IMSCON TYPE(ISCUSER) command (continued)

Return code	Reason code	Meaning
X'0C000008'	X'00002018'	The NAME keyword was not specified, but is a required keyword for the UPDATE IMSCON TYPE(ISCUSER) command.
X'0C000008'	X'00002020'	The STOP keyword must be specified.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, which is accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, which is accompanied by its completion code.
X'0C000014'	X'00005000'	The command processor failed to obtain storage via BPECBGET.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 441. Completion codes for the UPDATE IMSCON TYPE(ISCUSER) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(ISCUSER) command completed successfully for the resources.
8	Command processing error	The ISCUSER resource could not be processed because of an internal processing error. Refer to the command return and reason codes for more details.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active currently. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Example

In the following example, the command stops the ISC user session, ISCUSER1, which is associated with the ISC link ISCLINK1.

TSO SPOC input:
```
UPDATE IMSCON TYPE(ISCUSER) NAME(ISCUSER1) STOP(COMM)
```

TSO SPOC output:
```
ISCUser  ISCName  MbrName           CC
ISCUSER1 ISCLINK1 HWS1               0
```

OM API input:
```
CMD(UPDATE IMSCON TYPE(ISCUSER) NAME(ISCUSER1) STOP(COMM))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2012.335 17:33:09.520770</statime>
<stotime>2012.335 17:33:09.535112</stotime>
<staseq>CA8CA13CBF18231A</staseq>
<stoseq>CA8CA13CC2988D9A</stoseq>
<rqsttkn1>USRT001 10093309</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(ISCUSER) NAME(ISCUSER1) STOP(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="IUSER" llbl="ISCUser" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="ISC" llbl="ISCName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="3" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>IUSER(ISCUSER1) ISC(ISCLINK1) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Related reference:

➡ ISC statement (System Definition)

➡ RMTCICS statement (System Definition)

UPDATE IMSCON TYPE(LINK) command

Use the UPDATE IMSCON TYPE(LINK) command to stop communications on an MSC logical link that is assigned to an MSC physical link in IMS Connect.

Subsections:

- "Examples" on page 1147

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►──┬─UPDATE─┬──IMSCON──TYPE(LINK)──NAME(──┬─◄──logical_link_name──┬──)───────────────►
    └─UPD────┘                            └─logical_link_name*───┘

►──STOP(COMM)──────────────────────────────────────────────────────────────────►◄
            └─MSC(──┬─◄──lclplk_id──┬──)─┘
                    └─lclplk_id*────┘
```
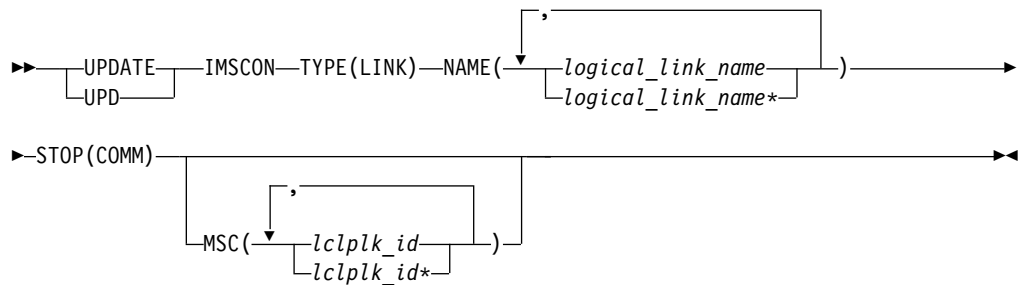
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(LINK) command.

MSC
    Specifies the MSC physical link ID defined in the LCLPLKID parameter of the MSC statement. You can specify a single MSC physical link name or a list of MSC physical link names separated by commas. Wildcards can be used in the names.

    This is an optional keyword. If specified, the command applies only to those logical links that are associated with the specified physical link ID. If omitted, then the command applies to logical links that are associated with any physical link ID.

NAME
    Specifies the MSC logical link name. You can specify a single MSC logical link name or a list of MSC logical link name separated by commas. Wildcards can be used in the names.

STOP(COMM)
    Stops logical link communication for the MSC that is associated with the link.

Usage notes

You can issue the UPDATE IMSCON TYPE(LINK) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

Use the UPDATE IMSCON TYPE(LINK) command to clean up the resources associated with an MSC logical link when the link has already been terminated, but the IMS Connect resources associated with the link were not cleaned up correctly.

Recommendation: Use the IMS command /PSTOP to terminate MSC logical links. Use the UPDATE IMSCON TYPE(LINK) command only when the IMS Connect resources associated with an MSC logical link that has already been terminated have failed to clean up correctly.

When the UPDATE IMSCON TYPE(LINK) command is issued, IMS Connect:
- Stops communication on the MSC logical link
- Informs IMS that communication has stopped on the logical link
- Deletes the control blocks associated with the logical link and frees the associated storage
- Issues message HWSF3310I

To avoid accidentally stopping a logical link that uses the same logical link name on another physical link, limit the command processing to a specific physical link by specifying the name of the target physical link as the *lclplk_id* value.

If the UPDATE IMSCON TYPE(LINK) command is issued against two or more physical links, or the *lclplk_id* value is omitted, IMS Connect stops communication on all MSC logical links that match the *logical_link_name* specified on the UPDATE IMSCON TYPE(LINK) command.

To display information about the MSC logical links that are assigned to the MSC physical links that are defined to an IMS Connect instance, use any of the following commands:
- In the IMS type-2 command format, either QUERY IMSCON TYPE(LINK) or QUERY IMSCON TYPE(MSC)
- In the WTOR command format, VIEWMSC
- In the z/OS MODIFY command format, QUERY MSC

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(LINK) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 442. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(LINK) command

UPDATE IMSCON TYPE(LINK) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(LINK) NAME(*logical_link_name*) STOP(COMM)	STOPLINK *logical_link_name*	DELETE LINK NAME(*linkName*)
UPDATE IMSCON TYPE(LINK) NAME(*logical_link_name*) MSC(*lclplk_id*) STOP(COMM)	STOPLINK *logical_link_name lclplk_id*	DELETE LINK NAME(*linkname*) LCLPLKID(*lclPlkid*)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 443. Output fields for the UPDATE IMSCON TYPE(LINK) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 445 on page 1147.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
LINK	Link	N/A	Specifies the MSC logical link that was terminated. The identifier is always returned.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
MSC	MscName	N/A	Identifier of the MSC physical link associated with the logical link that was terminated. The identifier, which is defined on the LCLPLKID parameter, is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(LINK) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 444. Return and reason codes for the UPDATE IMSCON TYPE(LINK) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(LINK) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 445. Completion codes for the UPDATE IMSCON TYPE(LINK) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(LINK) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for UPDATE IMSCON TYPE(LINK) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(LINK) NAME(MSCLINK1) STOP(COMM)
```

TSO SPOC output:

```
Link      MscName  MbrName  CC
MSCLINK1  MSC12    HWS1      0
```

OM API input:

```
CMD(UPDATE IMSCON TYPE(LINK) NAME(MSCLINK1) STOP(COMM))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 02:42:55.240634</statime>
<stotime>2010.298 02:42:55.245578</stotime>
<staseq>C6C783BC615BAEFC</staseq>
<stoseq>C6C783BC6290A27C</stoseq>
<rqsttkn1>USRT001 10194255</rqsttkn1>
```

```
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON        </kwd>
<input>UPD IMSCON TYPE(LINK) NAME(MSCLINK1) STOP(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="LINK" llbl="Link" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MSC" llbl="MscName" scope="LCL" sort="a" key="1"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>LINK(MSCLINK1) MSC(MSC12   ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The communication for the MSC, MSC12, associated with the logical link MSCLINK1 has been stopped.

Related reference:

➡ STOPLINK command (Commands)

➡ IMS Connect DELETE LINK command (Commands)

➡ MSC statement (System Definition)

UPDATE IMSCON TYPE(MSC) command

Use the UPDATE IMSCON TYPE(MSC) command to restart the MSC communications after it has been terminated and to terminate all communications to an MSC physical link, including all associated logical links.

Subsections:
- "Environment"
- "Syntax" on page 1149
- "Keywords" on page 1149
- "Usage notes" on page 1149
- "Equivalent WTOR and z/OS commands" on page 1150
- "Output fields" on page 1150
- "Return, reason, and completion codes" on page 1151
- "Examples" on page 1153

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).

- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
>>--+--UPDATE--+--IMSCON--TYPE(MSC)--NAME(--+-+--lclPlkid---+--+--)--+--START(COMM)--+-->><
    +--UPD-----+                            | |                |     +--STOP(COMM)---+
                                            | +--lclPlkid*-----+
                                            +--------,---------+
```
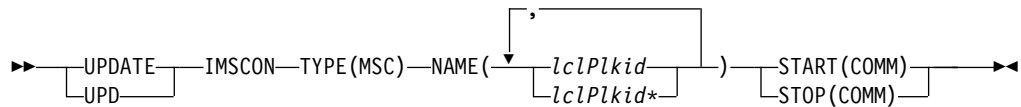
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(MSC) command.

NAME
Specifies the MSC physical link ID defined in the LCLPLKID parameter of the MSC statement. You can specify a single MSC physical link name or a list of MSC physical link name separated by commas. Wildcards can be used in the names.

START(COMM)
Starts or restarts the MSC communications of the MSC physical link after it has been terminated by an UPDATE IMSCON TYPE(MSC) STOP(COMM) command. The status of the MSC communication for the MSC physical link is set to "ACTIVE" after the completion of the command. START(COMM) is mutually exclusive with STOP(COMM).

STOP(COMM)
Terminates all communications to an MSC physical link, including all its logical links. The status of the MSC communication for the MSC physical link is set to "NOTACTIVE" after the completion of the command. STOP(COMM) is mutually exclusive with START(COMM).

Usage notes

You can issue the UPDATE IMSCON TYPE(MSC) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

When the UPDATE IMSCON TYPE(MSC) STOP(COMM) command is issued, IMS Connect:
- Stops communication on the specified MSC physical link, including communications on all the MSC logical links that are assigned to the physical link
- Informs IMS that communication has stopped on the physical link so that IMS can also terminate the physical link and any logical links that are assigned to the physical link
- Changes the status of the MSC physical link and its assigned logical links to NOT ACTIVE
- For TCP/IP generic resources, clears affinity of a physical link to the IMS system
- Issues message HWSF3305I

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(MSC) command.

Notes:

- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 446. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(MSC) command

UPDATE IMSCON TYPE(MSC) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(MSC) NAME(*lclPlkid*) START(COMM)	STARTMSC *lclPlkid*	UPDATE MSC NAME(*lclPlkid*) START(COMM)
UPDATE IMSCON TYPE(MSC) NAME(*lclPlkid*) STOP(COMM)	STOPMSC *lclPlkid*	UPDATE MSC NAME(*lclPlkid*) STOP(COMM)

Output fields

Short label
: Contains the short label that is generated in the XML output.

Long label
: Contains the column heading displayed on the TSO SPOC screen.

Keyword
: Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
: Provides a brief description of the output field.

Table 447. Output fields for the UPDATE IMSCON TYPE(MSC) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 449 on page 1151.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
MSC	MscName	N/A	The MSC physical link name. The MSC physical link name is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(MSC) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 448. Return and reason codes for the UPDATE IMSCON TYPE(MSC) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(MSC) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 449. Completion codes for the UPDATE IMSCON TYPE(MSC) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(MSC) command completed successfully for the resources.
9	Data store table create failed	During START(COMM) processing, IMS Connect could not create a data store table (DST) control block. For more information, see the HWSD0212E message that is included in the command reply message.
A	TRWB creation failed	The command attempted to create a TWU control block for the scheduler controller thread, the transmit thread, or the receive thread for the MSC physical link, but the BPECBGET request failed. The command output might also contain one of the following messages: • HWSD0212E • HWSD0222W
B	Data store table enqueue failed	During START(COMM) processing, IMS Connect attempted to enqueue a DST control block to the active queue for the MSC physical link, but the request failed. For more information, see the HWSD0212E message that is included in the command reply message.

Table 449. Completion codes for the UPDATE IMSCON TYPE(MSC) command (continued)

Completion code	Completion code text	Meaning
C	Create scheduler thread failed	During START(COMM) processing, IMS Connect attempted to create the scheduler controller thread for the MSC physical link, but the BPETHDCR request failed.
		For more information, see the HWSD0212E message that is included in the command reply message.
D	Data store is in shutdown	IMS Connect could not start the MSC physical link because the MSC physical link was shutting down.
		For more information, see the HWSD0222W message that is included in the command reply message.
E	Data store block create failed	During START(COMM) processing, IMS Connect could not start the MSC physical link because an attempt to create a data store block (DSB) for either the transmit or receive thread for the MSC physical link failed.
		For more information, see the HWSD0222W message that is included in the command reply message.
F	Create xmit/recv thread failed	During START(COMM) processing, IMS Connect attempted to create either the transmit or receive thread for the MSC physical link, but the BPETHDCR request failed.
		For more information, see the HWSD0222W message that is included in the command reply message.
10	No resources found	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
11	Resource is stopped	During STOP(COMM) processing, IMS Connect could not stop a MSC physical link because the MSC physical link is already stopped.
12	Resource is not stopped	During START(COMM) processing, IMS Connect could not start a MSC physical link because the MSC physical link is not stopped.
1A	BPEGETM error	During START(COMM) processing, IMS Connect failed to obtain storage from the BPEGETM service.
		The command output might also contain the following message: HWSE3130W.
2F	SCI is not available	During START(COMM) processing, IMS Connect detected that SCI is not available to communicate with MSC.
		The command output might also contain the following message: HWSF3315E.

Examples

Example 1 for UPDATE IMSCON TYPE(MSC) command

TSO SPOC input:
```
UPDATE IMSCON TYPE(MSC) NAME(MSC12) STOP(COMM)
```

TSO SPOC output:
```
MscName    MbrName   CC
MSC12      HWS1       0
```

OM API input:
```
CMD(UPDATE IMSCON TYPE(MSC) NAME(MSC12) STOP(COMM))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 02:45:18.138130</statime>
<stotime>2010.298 02:45:18.155690</stotime>
<staseq>C6C78444A871292C</staseq>
<stoseq>C6C78444ACBAA9AC</stoseq>
<rqsttkn1>USRT001 10194518</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(MSC) NAME(MSC12) STOP(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="MSC" llbl="MscName" scope="LCL" sort="a" key="1"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MSC(MSC12   ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The communication between IMS Connect, HWS1, and MSC MSC12 has been stopped.

Related reference:

➡ STARTMSC command (Commands)

➡ STOPMSC command (Commands)

➡ IMS Connect UPDATE MSC command (Commands)

UPDATE IMSCON TYPE(ODBM) command

Use the UPDATE IMSCON TYPE(ODBM) command to start and stop communication between the IMS Connect and an IMS Open Database Manager (ODBM).

Use the UPDATE IMSCON TYPE(ODBM) command to reestablish communication with an ODBM after communication fails between the IMS Connect and the ODBM. For example, use this command to restart communication when all activities for the ODBM in the IMS Connect is terminated.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 1155
- "Equivalent WTOR and z/OS commands" on page 1155
- "Output fields" on page 1155
- "Return, reason, and completion codes" on page 1156
- "Examples" on page 1158

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
>>──┬─UPDATE─┬──IMSCON──TYPE(ODBM)──NAME(──┬──┬─odbm_name──┬──┬─)──────────>
    └─UPD────┘                        ┌─,◄──┘  └─odbm_name*─┘
                                                                            

>──┬─START(COMM)─┬──────────────────────────────────────────────────────><
   └─STOP(COMM)──┘
```
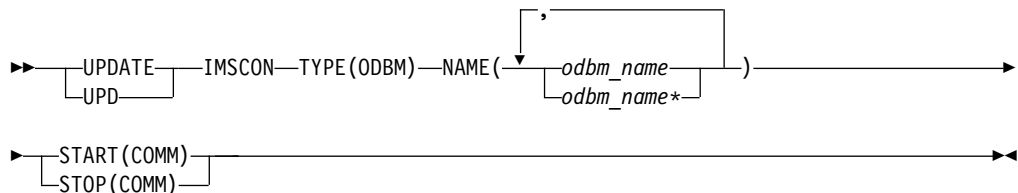
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(ODBM) command.

NAME
   Specifies the names of one or more IMS ODBMs to be updated. You can specify a single ODBM name or a list of ODBM names separated by commas. Wildcards can be used in the names.

START(COMM) | STOP(COMM)
   Specifies to start or stop communication with IMS ODBM.

START(COMM)
Starts communication with IMS ODBM. START(COMM) is mutually exclusive with STOP(COMM).

STOP(COMM)
Stops communication with IMS ODBM. STOP(COMM) is mutually exclusive with START(COMM).

Usage notes

You can issue the UPDATE IMSCON TYPE(ODBM) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Use the UPDATE IMSCON TYPE(ODBM) command for any type of error situation that requires immediate termination of communication with an ODBM. Work currently in progress for an ODBM is ended and communications with that ODBM and its threads are terminated.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(ODBM) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 450. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(ODBM) command

UPDATE IMSCON TYPE(ODBM) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UDPATE IMSCON TYPE(ODBM) NAME(*odbm_name*) START(COMM)	STARTOD *odbm_name*	UPDATE ODBM NAME(*odbmName*) START(COMM)
UDPATE IMSCON TYPE(ODBM) NAME(*odbm_name*) STOP(COMM)	STOPOD *odbm_name*	UPDATE ODBM NAME(*odbmName*) STOP(COMM)

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be

generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
Provides a brief description of the output field.

Table 451. Output fields for the UPDATE IMSCON TYPE(ODBM) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 453.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
ODBM	ODBMName	N/A	Name of the ODBM. The ODBM name is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(ODBM) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 452. Return and reason codes for the UPDATE IMSCON TYPE(ODBM) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(ODBM) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 453. Completion codes for the UPDATE IMSCON TYPE(ODBM) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(ODBM) command completed successfully for the resources.

Table 453. Completion codes for the UPDATE IMSCON TYPE(ODBM) command (continued)

Completion code	Completion code text	Meaning
9	Data store table create failed	During START(COMM) processing, IMS Connect could not create a data store table (DST) control block.
		For more information, see the HWSD0212E message that is included in the command reply message.
A	TRWB creation failed	The command attempted to create a TWU control block for the scheduler controller thread, the transmit thread, or the receive thread for the ODBM data store, but the BPECBGET request failed.
		The command output might also contain one of the following messages:
		• HWSD0212E
		• HWSM0522W
B	Data store table enqueue failed	During START(COMM) processing, IMS Connect attempted to enqueue a DST control block to the active queue for the ODBM data store, but the request failed.
		For more information, see the HWSD0212E message that is included in the command reply message.
C	Create scheduler thread failed	During START(COMM) processing, IMS Connect attempted to create the scheduler controller thread for the ODBM data store, but the BPETHDCR request failed.
		For more information, see the HWSD0212E message that is included in the command reply message.
D	Data store is in shutdown	IMS Connect could not start the ODBM data store because the data store was shutting down.
		For more information, see the HWSM0522W message that is included in the command reply message.
E	Data store block create failed	During START(COMM) processing, IMS Connect could not start the ODBM data store because an attempt to create a data store block (DSB) for either the transmit or receive thread for the data store failed.
		For more information, see the HWSM0522W message that is included in the command reply message.
F	Create xmit/recv thread failed	During START(COMM) processing, IMS Connect attempted to create either the transmit or receive thread for the ODBM data store, but the BPETHDCR request failed.
		For more information, see the HWSM0522W message that is included in the command reply message.
10	No resources found	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active now. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
11	Resource is stopped	During STOP(COMM) processing, IMS Connect could not stop an ODBM because the ODBM is already stopped.

Completion code	Completion code text	Meaning
12	Resource is not stopped	During START(COMM) processing, IMS Connect could not start a ODBM because the ODBM is not stopped.
1A	BPEGETM error	During START(COMM) processing, IMS Connect failed to obtain storage from the BPEGETM service.
		The command output might also contain the following message: HWSQ2230W.
2F	SCI is not available	During START(COMM) processing, IMS Connect detected that SCI is not available to communicate with ODBM.
		The command output might also contain one of the following messages: • HWSN1915E • HWSQ2250W
30	IMSA creation failed	During START(COMM) processing, IMS Connect encountered an error while attempting to obtain a block of storage of type IMSA from the BPECBGET service.
		The command output might also contain the following message: HWSQ2230W.
31	Registration to ODBM failed	During START(COMM) processing, IMS Connect encountered an error when registering with ODBM.
		The command output might also contain the following message: HWSQ2240W.
32	Deregistration to ODBM failed	During START(COMM) processing, IMS Connect encountered an error when deregistering with ODBM.
		The command output might also contain the following message: HWSQ2245W.

Examples

Example 1 for UPDATE IMSCON TYPE(ODBM) command

TSO SPOC input:

```
UDPATE IMSCON TYPE(ODBM) NAME(ODBMA) START(COMM)
```

TSO SPOC output:

```
ODBMName  MbrName  CC
ODBMA     HWS1      0
```

OM API input:

```
CMD(UDPATE IMSCON TYPE(ODBM) NAME(ODBMA) START(COMM))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 15:54:38.327140</statime>
<stotime>2010.298 15:54:38.328400</stotime>
```

```
<staseq>C6C834B2DA964FEC</staseq>
<stoseq>C6C834B2DAE507AC</stoseq>
<rqsttkn1>USRT001 10085438</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON        </kwd>
<input>UPD IMSCON TYPE(ODBM) NAME(ODBMA) START(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="ODBM" llbl="ODBMName" scope="LCL" sort="a" key="1"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>ODBM(ODBMA   ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The communication between IMS Connect, HWS1, and the IMS ODBM, ODBMA, has been restarted.

Related reference:

➡ STARTOD command (Commands)

➡ STOPOD command (Commands)

UPDATE IMSCON TYPE(PORT) command

Use the UPDATE IMSCON TYPE(PORT) command to terminate listening on a TCP/IP port, to reestablish a TCP/IP connection to enable listening on a TCP/IP port, or to change port attributes.

Subsections:
- "Environment"
- "Syntax" on page 1160
- "Keywords" on page 1160
- "Usage notes" on page 1162
- "Equivalent WTOR and z/OS commands" on page 1162
- "Output fields" on page 1163
- "Return and reason codes" on page 1164
- "Completion codes" on page 1164
- "Examples" on page 1166

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).

- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax
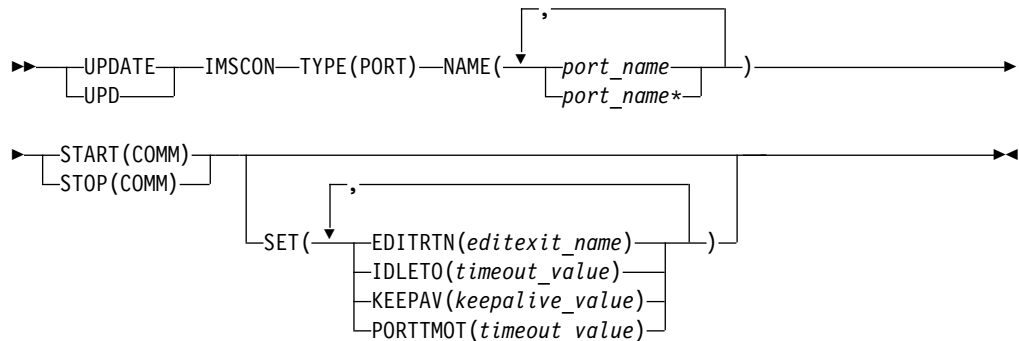
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(PORT) command.

NAME
> Specifies the names of one or more ports. You can specify a single port name or a list of port names separated by commas. Wildcards can be used in the names. The port name must match a port defined to IMS Connect on one of the configuration statements in the IMS Connect PROCLIB member or by using the CREATE IMSCON TYPE(PORT) command..
>
> To update the local port used by the IMS TM Resource Adapter, specify NAME(LOCAL).
>
> An SSL port is displayed with the character "S" appended to the end of the port number. To update the SSL port, specify the port number either with or without the character "S" appended to the end of the port number.
>
> A DRDA port for ODBM use is displayed with the character "D" appended to the end of the port number. To update the DRDA port, specify the port number either with or without the character "D" appended to the end of the port number.
>
> A port defined as a CICSPORT for receiving messages on an ISC link from a CICS subsystem is displayed with the character "C" appended to the end of the port number. To update a CICSPORT port, specify the port number either with or without the character "C" appended to the end of the port number.

SET()
> Specifies the attribute values to be changed. The SET() keyword cannot be used in conjunction with the keywords START() or STOP().
>
> **EDITRTN(*editexit_name*)**
>> Specifies a 1- to 8-character alphanumeric name of the Port Message Edit exit routine that can modify messages that do not conform to standard message formats for IMS Connect. The exit routine must be accessible to IMS Connect by either JOBLIB, STEPLIB, or LinkList.

The Port Message Edit exit routine and the EDITRTN attribute are only valid for regular ports. Invalid port types are: LOCAL, ODBM, SSL, and CICS.

To set the EDITRTN attribute, the port must be in NOTACTIVE state by issuing the UPD IMSCON TYPE(PORT) STOP(COMM) command. If needed, run the command with the STOP(COMM) attribute prior to running the command with the EDITRTN attribute.

When IMS Connect is unable to load the specified Port Message Edit exit routine, it will use the previously defined, if any, Port Message Edit exit routine when the port is started.

To disable the use of the Port Message Edit exit routine, use a zero as the value for the EDITRTN parameter, `EDITRTN(0)`. The name of the previously set Port Message Edit exit routine still shows in the output of the BPE DISPLAY USEREXIT command, but IMS Connect does not call the Port Message Edit exit routine.

The use of the BPE REFRESH USEREXIT command to refresh the existing Port Message Edit exit routine might produce unexpected results when the IMS Connect port is active, because the new copy of the exit becomes available immediately after command completion. This might affect both input and output messages for currently connected TCP/IP clients. The BPE REFRESH USEREXIT command is processed by BPE, so IMS Connect does not issue the TERM call to the old version of the exit or the INIT call to the new version of the exit. Instead of using the BPE REFRESH USEREXIT command to update the port's edit exit routine, try to use the UPD IMSCON TYPE(PORT) SET(EDITRTN) command with the current name of the Port Message Edit exit routine.

IDLETO

For open socket connections on this port, specifies the timeout interval for connections that are idle and waiting in a RECV state for the next message. When the timeout interval expires, IMS Connect closes the socket connection due to inactivity.

The timeout interval is in hundredths of seconds. Timeout values can be from -1 to 2 147 483 647 (X'7FFFFFFF'). A value of 0 disables this timeout function which prevents inactive connections from timing out.

A value of -1 specifies that connections on the port use the global-level idle timeout value.

Unless -1 is specified, the port-level idle timeout value overrides the global-level idle timeout value.

This parameter is optional and applies only to persistent socket connections.

The IDLETO parameter is valid only for ports that are used for communication with IMS TM, which are defined by the PORT= and PORTID= parameters in the TCPIP statement of the IMS Connect HWSCFGxx configuration member or by the CREATE IMSCON TYPE(PORT) command.

KEEPAV(*keepalive_value*)

Specifies a numeric value for the TCP/IP KeepAlive interval for sockets on this port. The KeepAlive interval is in seconds. The range is from 0 to 2,147,460 seconds. If zero is specified, the TCP/IP stack interval value is used.

The KEEPAV attribute is only valid for regular, DRDA, and CICS ports. Invalid port types are: LOCAL and SSL.

PORTTMOT(*timeout_value*)

Specifies a numeric value for the amount of time that IMS Connect waits for the next input message from a client application that is connected on a DRDA port before IMS Connect disconnects the client. The timeout interval is in hundredths of a second. The range is from 0 to 2,147,483,647 hundredths of a second. A value of zero disables the timeout function.

The PORTTMOT attribute is only valid for DRDA ports. Invalid port types are: LOCAL, regular, SSL, and CICS.

START(COMM) | STOP(COMM)

Specifies to start or stop communication on the port.

START(COMM)

Starts communication on the port. Use the UPDATE IMSCON TYPE(PORT) START(COMM) command when communication stops between the IMS Connect and a TCP/IP port, but the IMS Connect has not terminated.

STOP(COMM)

Stops communication on the port. Work currently in progress can continue for existing clients. Only the listening for new request messages on the port is terminated immediately. When existing work has completed, the port is no longer active.

Usage notes

You can issue the UPDATE IMSCON TYPE(PORT) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Updates that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect, unless the updates are also made to the corresponding configuration statement in the HWSCFGxx member of the IMS PROCLIB data set before the next restart.

A port must be stopped (in NOTACTIVE state) before its EDITRTN attribute can be updated with the UPDATE IMSCON TYPE(PORT) SET(EDITRTN()) command.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(PORT) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 454. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(PORT) command.

UPDATE IMSCON TYPE(PORT) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UDPATE IMSCON TYPE(PORT) NAME(*port_name*) START(COMM)	OPENPORT *port_id*	UPDATE PORT NAME(*port_name*) START(COMM)
	STARTPT *port_id*	
UDPATE IMSCON TYPE(PORT) NAME(*port_name*) STOP(COMM)	STOPPORT *port_id*	UPDATE PORT NAME(*port_name*) STOP(COMM)

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
Provides a brief description of the output field.

Table 455. Output fields for the UPDATE IMSCON TYPE(PORT) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
ERRT	ErrorText	*error*	Error text that provides diagnostic information. Error text can be returned for a non-zero completion code and further explains the completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.

Table 455. Output fields for the UPDATE IMSCON TYPE(PORT) command (continued)

Short label	Long label	Keyword	Meaning
PORT	Port	N/A	The port number. The port number is always returned. If one of the following characters is appended to the end of the port number, it indicates that the port is dedicated to a particular purpose: **C** Identifies a CICS port. **D** Identifies an ODBM port. **S** Identifies an SSL port. If "LOCAL" is displayed instead of a port number, the port is a local port that is used by the IMS TM Resource Adapter.

Return and reason codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(PORT) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 456. Return and reason codes for the UPDATE IMSCON TYPE(PORT) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(PORT) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'02000008'	X'0000203C'	The command entered contains an invalid keyword parameter value. The command was rejected.
X'0C000008'	X'0000209C'	The SET() keyword cannot be used in conjunction with the keywords START() or STOP().
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.

Completion codes

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 457. Completion codes for the UPDATE IMSCON TYPE(PORT) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(PORT) command completed successfully for the resources.
3	Some attributes not updated	The command was successful for some attributes. There is a separate line of output with its own completion code for each attribute that IMS Connect failed to set.
10	No resources found	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
11	Resource is stopped	During STOP(COMM) processing, IMS Connect could not stop a port because it was already stopped.
12	Resource is not stopped	During START(COMM) processing, IMS Connect could not start a port because it was already started.
14	Resource is not stopped	The resource is not stopped. The resource must be stopped first to update the attribute that is displayed in the ErrorText column.
17	Attribute is invalid for rsc	The attribute that is displayed in the ErrorText column is invalid for this particular port. For example, the attribute might not be supported on a CICS or DRDA port. The attribute cannot be set.
12	Resource is not stopped	The resource is not stopped. To update attributes of this resource, it must be stopped first.
37	Continuing to retry command	IMS Connect encountered an error while processing the command. It continues to retry processing of the command in the background. The command output might also contain one of the following messages, to indicate why the command failed: • HWSP1400W • HWSP1402W • HWSP1405W • HWSP1415I • HWSP1415E • HWSP1420E • HWSP1485E • HWSS0771W • HWSS0777W
38	Unable to start port scheduler	IMS Connect was unable to start the port scheduler. The command output also contains the following message: HWSS0712W.
3D	Start process was interrupted	START(COMM) processing was interrupted by another command or because IMS Connect is shutting down. The command output contains the following message: HWSS0770I.

Table 457. Completion codes for the UPDATE IMSCON TYPE(PORT) command (continued)

Completion code	Completion code text	Meaning
4B	Unable to load port edit routine	IMS Connect could not load the Port Message Edit exit routine. The name of the Port Message Edit exit routine is displayed in the ErrorText column.
4C	Unable to update attribute	IMS Connect could not update the attribute that is displayed in the ErrorText column.

Examples

Example 1 for UPDATE IMSCON TYPE(PORT) command

TSO SPOC input:
```
UDPATE IMSCON TYPE(PORT) NAME(9999) START(COMM)
```

TSO SPOC output:
```
Port   MbrName  CC
9999   HWS1      0
```

OM API input:
```
CMD(UDPATE IMSCON TYPE(PORT) NAME(9999) START(COMM))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 02:49:44.222509</statime>
<stotime>2010.298 02:49:44.225852</stotime>
<staseq>C6C785426A72DABA</staseq>
<stoseq>C6C785426B43C87A</stoseq>
<rqsttkn1>USRT001 10194944</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(PORT) NAME(9999) START(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left"skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: IMS Connect, HWS1, is now listening on the TCP/IP port 9999.

Example 2 for UPDATE IMSCON TYPE(PORT) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(PORT) NAME(9999) START(COMM)
```

TSO SPOC output:

```
IMSplex . . . . . . : PLEX1
Routing . . . . . . : HWS1
Start time. . . . : 2013.253 16:52:42.35
Stop time . . . . : 2013.253 16:52:42.35
Return code . . . : 0200000C
Reason code . . . : 00003008
Reason text . . . : None of the clients were successful.
Command master. . : HWS1


            Return    Reason
MbrName     Code      Code      Reason text
--------    --------  --------  --------------
HWS1        0C00000C  00003004  No requests were successful

MbrName  Messages
-------- ------------------------
HWS1     HWSP1415E TCP/IP SOCKET FUNCTION CALL FAILED; F=GETSOCK , R=-1, E=112,
M=SDOT, ID=9999
```

Explanation: IMS Connect, HWS1, could not start communication on the TCP/IP port 9999, because a TCP/IP socket function call failed.

Example 3 for UPDATE IMSCON TYPE(PORT) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(PORT) NAME(9999)
SET(KEEPAV(120000))
```

TSO SPOC output:

```
Port    MbrName          CC
9999    HWS1              0
```

OM API input:

```
CMD(UPD IMSCON TYPE(PORT) NAME(9999)
SET(KEEPAV(120000)))
```

OM API output:

```
<?xml version="1.0"?>
<!DOCTYPE imsout SYSTEM "imsout.dtd">
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2014.232 16:09:50.846238</statime>
<stotime>2014.232 16:09:50.846556</stotime>
<staseq>CDA221E31B91EFD4</staseq>
<stoseq>CDA221E31BA5CD54</stoseq>
<rqsttkn1>USRT011 10090950</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
```

```
|                          <master>HWS1    </master>
|                          <userid>USRT011 </userid>
|                          <verb>UPD </verb>
|                          <kwd>IMSCON          </kwd>
|                          <input>UPDATE IMSCON TYPE(PORT) NAME(9999) SET(KEEPAV(120000)) </input>
|                          </cmd>
|                          <cmdrsphdr>
|                          <hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
|                           len="8" dtype="CHAR" align="left" skipb="no" />
|                          <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
|                           len="8" dtype="CHAR" align="left" skipb="no" />
|                          <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes" len="4"
|                           dtype="INT" align="right" skipb="no" />
|                          <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0" scroll="yes"
|                           len="32" dtype="CHAR" align="left" skipb="yes" />
|                          </cmdrsphdr>
|                          <cmdrspdata>
|                          <rsp>PORT(9999    ) MBR(HWS1            ) CC(   0) </rsp>
|                          </cmdrspdata>
|                          </imsout>
```

| **Explanation:** The KeepAlive interval (KEEPAV) attribute is set to 120,000 seconds
| for port 9999.

| *Example 4 for UPDATE IMSCON TYPE(PORT) command*

| TSO SPOC input:
```
| UPDATE IMSCON TYPE(PORT) NAME(9999,7777)
| SET(KEEPAV(120000),EDITRTN(HWSEDIT0),PORTTMOT(6000))
```

| TSO SPOC output:
```
| Port    MbrName             CC CCText                       ErrorText
| 7777D   HWS1                 3 Command complete for some
| 7777D   HWS1                17 Attribute is invalid for rsc  EDITRTN
| 9999    HWS1                 3 Command complete for some
| 9999    HWS1                14 Resource is not stopped       EDITRTN
| 9999    HWS1                17 Attribute is invalid for rsc  PORTTMOT
```

| OM API input:
```
| CMD(UPDATE IMSCON TYPE(PORT) NAME(9999,7777)
| SET(KEEPAV(120000),EDITRTN(HWSEDIT0),PORTTMOT(6000)))
```

| OM API output:
```
| <?xml version="1.0"?>
| <!DOCTYPE imsout SYSTEM "imsout.dtd">
| <imsout>
| <ctl>
| <omname>OM1OM   </omname>
| <omvsn>1.7.0</omvsn>
| <xmlvsn>20  </xmlvsn>
| <statime>2014.232 16:26:34.861722</statime>
| <stotime>2014.232 16:26:34.862037</stotime>
| <staseq>CDA225A09C89AE52</staseq>
| <stoseq>CDA225A09C9D50D2</stoseq>
| <rqsttkn1>USRT011 10092634</rqsttkn1>
| <rc>0200000C</rc>
| <rsn>00003008</rsn>
| <rsnmsg>CSLN054I</rsnmsg>
| <rsntxt>None of the clients were successful.</rsntxt>
| </ctl>
| <cmderr>
| <mbr name="HWS1    ">
| <typ>IMSCON  </typ>
```

```
<rc>0C00000C</rc>
<rsn>00003004</rsn>
<rsntxt>No requests were successful   </rsntxt>
</mbr>
</cmderr>
<cmd>
<master>HWS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPDATE IMSCON TYPE(PORT) NAME(9999,7777)
 SET(KEEPAV(120000),EDITRTN(HWSEDIT0),PORTTMOT(6000)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes" len="4"
 dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0" scroll="yes"
 len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="a" key="3" scroll="no"
 len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(  14) CCTXT(Resource is not
 stopped) ERRT(EDITRTN         ) </rsp>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(  17) CCTXT(Attribute is invalid
 for rsc) ERRT(PORTTMOT        ) </rsp>
<rsp>PORT(9999    ) MBR(HWS1            ) CC(   3) CCTXT(Command complete for
 some) </rsp>
<rsp>PORT(7777D   ) MBR(HWS1            ) CC(  17) CCTXT(Attribute is invalid
 for rsc) ERRT(EDITRTN         ) </rsp>
<rsp>PORT(7777D   ) MBR(HWS1            ) CC(   3) CCTXT(Command complete for
 some) </rsp>
</cmdrspdata>
</imsout>
```

The command is attempting to set the KEEPAV, EDITRTN, and PORTTMOT
attributes for regular port 9999 and DRDA port 7777.

For DRDA port 7777, some updates are successful. A separate completion code is
displayed for each attribute update that is not successful. The Port Message Edit
exit routine (EDITRTN) attribute is not valid for a DRDA port, so it is not set. The
KeepAlive interval (KEEPAV) and the DRDA port timeout (PORTTMOT) attributes
are set successfully since there no completion codes are returned for them.

For DRDA port 7777, the Port Message Edit exit routine (EDITRTN) attribute is not
valid for a DRDA port, so none of the attributes for port 7777 are set.

Example 5 for UPDATE IMSCON TYPE(PORT) command

In the following example, the idle timeout value for connections on port 9999 is
updated to 3000, or 30 seconds.

TSO SPOC input:
```
UPDATE IMSCON TYPE(PORT) NAME(9999) SET(IDLETO(3000))
```

TSO SPOC output:
```
Port    MbrName           CC
9999    HWS1               0
```

OM API input:

CMD(UPDATE IMSCON TYPE(PORT) NAME(9999) SET(IDLETO(3000)))

OM API output:

```
<cmd>
<master>HWS1     </master>
<userid>USRT002 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPDATE IMSCON TYPE(PORT) NAME(9999) SET(IDLETO(3000)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PORT" llbl="Port" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="a" key="3"
 scroll="no" len="16" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PORT(9999    ) MBR(HWS1             ) CC(   0) </rsp>
</cmdrspdata>
```

Related reference:

[➡] OPENPORT command (Commands)

[➡] STARTPT command (Commands)

[➡] STOPPORT command (Commands)

[➡] IMS Connect UPDATE PORT command (Commands)

UPDATE IMSCON TYPE(RACFUID) command

Use the UPDATE IMSCON TYPE(RACFUID) command to refresh one or more RACF user IDs in the user ID cache.

Subsections:
- "Environment"
- "Syntax" on page 1171
- "Keywords" on page 1171
- "Usage notes" on page 1171
- "Equivalent WTOR and z/OS commands" on page 1171
- "Output fields" on page 1171
- "Return, reason, and completion codes" on page 1172
- "Examples" on page 1173

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
>>--+-UPDATE-+--IMSCON--TYPE(RACFUID)--NAME(--+--userid--+--)--OPTION(REFRESH)--><
     +-UPD----+                           ^-----,------+
```

Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(RACFUID) command.

NAME
> Specifies one or more RACF user IDs in the user ID cache to be refreshed. You can specify a single RACF user ID, or a list of RACF user IDs separated by commas. Wildcards cannot be used in the names.

OPTION(REFRESH)
> Refreshes the specified RACF user IDs.

Usage notes

You can issue the UPDATE IMSCON TYPE(RACFUID) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of `ACTIVE`.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(RACFUID) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 458. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(RACFUID) command

UPDATE IMSCON TYPE(RACFUID) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(RACFUID) NAME(*userid*) OPTION(REFRESH)	REFRESH RACFUID NAME(*userid*)	UPDATE RACFUID NAME(*userid*) OPTION(REFRESH)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 459. Output fields for the UPDATE IMSCON TYPE(RACFUID) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 461 on page 1173.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
UID	UserID	N/A	Name of the RACF user ID. The name is always returned.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(RACFUID) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 460. Return and reason codes for the UPDATE IMSCON TYPE(RACFUID) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(RACFUID) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 461. Completion codes for the UPDATE IMSCON TYPE(RACFUID) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(RACFUID) command completed successfully for the resources.
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

Example 1 for UPDATE IMSCON TYPE(RACFUID) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(RACFUID) NAME(USRT001) OPTION(REFRESH)
```

TSO SPOC output:

```
UserID   MbrName  CC
USRT001  HWS1      0
```

OM API input:

```
CMD ( UPDATE IMSCON TYPE(RACFUID) NAME(USRT001) OPTION(REFRESH) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 16:40:40.833976</statime>
<stotime>2010.298 16:40:40.835055</stotime>
<staseq>C6C83EFD62BB82A8</staseq>
<stoseq>C6C83EFD62FEFD28</stoseq>
<rqsttkn1>USRT001 10094040</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(RACFUID) NAME(USRT001) OPTION(REFRESH) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="UID" llbl="UserID" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
```

```
<cmdrspdata>
<rsp>UID(USRT001 ) MBR(HWS1           ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The RACF user ID USRT001 was successfully refreshed.

Related reference:

↪ REFRESH RACFUID command (Commands)

↪ IMS Connect UPDATE RACFUID command (Commands)

UPDATE IMSCON TYPE(RMTCICS) command

Use the UPDATE IMSCON TYPE(RMTCICS) command to stop or restart communication between an IMS Connect instance and a remote CICS subsystem.

Recommendation: Before issuing the UPDATE IMSCON TYPE(RMTCICS) command, confirm that there are no active ISC sessions (ISCUSERs) on the connection. Any active sessions should be shut down from IMS by using the /QUIESCE or /CLSDST command. If the sessions are terminated in IMS but still appear active in IMS Connect, the UPDATE IMSCON TYPE(RMTCICS) command cleans them up.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1175
- "Usage notes" on page 1175
- "Equivalent WTOR and z/OS commands" on page 1176
- "Output fields" on page 1176
- "Return, reason, and completion codes" on page 1177
- "Example of the UPDATE IMSCON TYPE(RMTCICS) command" on page 1179

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax
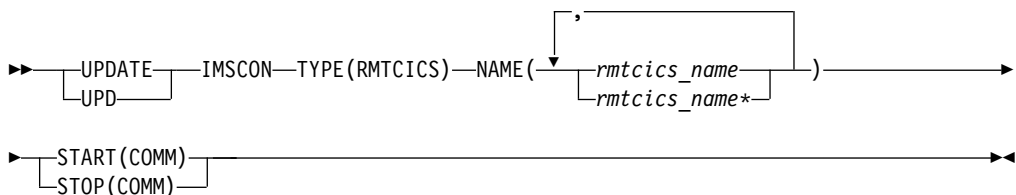
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(RMTCICS) command.

NAME
> Specifies one or more remote CICS connection names to be updated. You can specify a single remote CICS connection name or a list of remote CICS connection names separated by commas. You can use wildcard characters in the names.

START(COMM) | STOP(COMM)
> Mutually exclusive keywords that you use to start or stop communications for the remote CICS connection.

> **START(COMM)**
> > Starts the named remote CICS connection in IMS Connect. After issuing this command, the TCP/IP connection between IMS Connect and the remote CICS is enabled. However, if an ISC link was also stopped by the UPDATE IMSCON TYPE(ISC) STOP(COMM) command, to fully enable communications for ISC, you also need to issue the UPDATE IMSCON TYPE(ISC) START(COMM) command.

> **STOP(COMM)**
> > Stops communications on the remote CICS connection.

> > When the UPDATE IMSCON TYPE(RMTCICS) STOP(COMM) command is issued, IMS Connect:
> > - Stops all communication with a remote CICS instance on the connection specified in the command.
> > - Informs IMS that the connection to the remote CICS instance has been stopped so that IMS can shut down any affected ISC parallel sessions.
> > - Terminates existing socket connections and prevents new socket connections until the UPDATE RMTCICS START(COMM) is issued.
> > - Changes the status of the connection to STOPPED.
> > - Issues message HWSV4405I.

> > If IMS Connect receives a message for a connection in the STOPPED state, IMS Connect returns a NAK response to IMS and issues message HWSG4040W. IMS retains the message at the front of the queue.

Usage notes

You can issue the UPDATE IMSCON TYPE(RMTCICS) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

When the UPDATE IMSCON TYPE(RMTCICS) START(COMM) command is issued, IMS Connect resumes communications on the specified connection to a remote CICS instance. IMS Connect changes the status of the connection to NOT ACTIVE until new sockets are opened for communications with the remote instance of CICS.

The UPDATE IMSCON TYPE(RMTCICS) START(COMM) command is required only when communication on a connection with a remote CICS instance have been previously stopped by an UPDATE RMTCICS STOP(COMM) command.

For ISC communications, the UPDATE IMSCON TYPE(RMTCICS) START(COMM) command resumes only TCP/IP communications in IMS Connect, and does not change the stopped status of ISC links. If the ISC links have been stopped by an UPDATE IMSCON TYPE(ISC) STOP(COMM) command, after you issue the UPDATE IMSCON TYPE(RMTCICS) START(COMM) command, you also need to restart the ISC links in IMS Connect by issuing the UPDATE IMSCON TYPE(ISC) START(COMM) command.

Recommendation: When restarting ISC communications, to prevent IMS from sending ISC messages to IMS Connect before TCP/IP communications have been restarted, always issue the UPDATE IMSCON TYPE(RMTCICS) START(COMM) command before issuing the UPDATE IMSCON TYPE(ISC) START(COMM).

Use the QUERY IMSCON TYPE(RMTCICS) command to display information about the current connections to a remote CICS instance that are defined to the local IMS Connect instance.

Equivalent WTOR and z/OS commands

The UPDATE IMSCON TYPE(RMTCICS) command does not have an equivalent WTOR or z/OS MODIFY command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 462. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(RMTCICS) command

UPDATE IMSCON TYPE(RMTCICS) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(RMTCICS) NAME(*) START(COMM)	None	None
UPDATE IMSCON TYPE(RMTCICS) NAME(*rmtcics_name*) STOP(COMM)	None	None

Output fields

Short label
Contains the short label that is generated in the XML output.

Long label
Contains the column heading displayed on the TSO SPOC screen.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 463. Output fields for the UPDATE IMSCON TYPE(RMTCICS) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 465 on page 1178.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
RCICS	RmtCICS	N/A	Identifier of the remote CICS connection to be acted upon. The identifier is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(RMTCICS) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 464. Return and reason codes for the UPDATE IMSCON TYPE(RMTCICS) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(RMTCICS) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C000008'	X'00002004'	An invalid keyword or keyword parameter was specified.
X'0C000008'	X'00002014'	An invalid character was specified in the NAME() parameter.
X'0C000008'	X'00002018'	The NAME keyword was not specified, but is a required keyword for the UPDATE IMSCON TYPE(RMTCICS) command.
X'0C000008'	X'00002020'	Neither the STOP keyword nor the START keyword was specified. One of these must be specified.
X'0C000008'	X'00002044'	The STOP and START keywords were both specified in the command. Only one of these may be specified.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.

Table 464. Return and reason codes for the UPDATE IMSCON TYPE(RMTCICS) command (continued)

Return code	Reason code	Meaning
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005000'	The command processor failed to obtain storage via BPECBGET.
X'0C000014'	X'00005050'	The command processor was unable to stop the RMTCICS connection.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 465. Completion codes for the UPDATE IMSCON TYPE(RMTCICS) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(RMTCICS) command completed successfully for the resources.
A	TRWB creation failed	The command attempted to create a TWU block for the RMTCICS connection, but the attempt failed. Refer to message HWSV4415W which is included in the command output.
C	Create scheduler thread failed	The command attempted to create a BPE thread for the RMTCICS scheduler, but the attempt failed. Refer to message HWSV4415W which is included in the command output.
10	No Resources Found	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active at this time. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
11	Resource is stopped	The command could not stop a RMTCICS connection because it was already stopped.
12	Resource is not stopped	The command could not start a RMTCICS connection because it was not stopped.
1A	BPEGETM error	The command attempted to obtain storage needed for the RMTCICS connection, but the BPEGETM attempt failed. Refer to message HWSV4630W, which is included in the command output.
1E	BPEATTCH failed	The command attempted to attach a TCB for the RMTCICS connection, but the BPEATTCH attempt failed. Refer to message HWSV4415W, which is included in the command output.
1F	BPX1GAI get ipaddr failed	The command attempted to obtain the IP address of the CICS system, but the BPX1GAI call to TCP/IP failed. Refer to message HWSV4600E, which is included in the command output.

Table 465. Completion codes for the UPDATE IMSCON TYPE(RMTCICS) command (continued)

Completion code	Completion code text	Meaning
23	BPX1GAI invalid IPV4/6 struct	The command attempted to obtain the IP address of the CICS system, but the BPX1GAI call to TCP/IP returned an invalid IP address structure that was neither IPV4 or IPV6. Refer to message HWSV4600E, which is included in the command output.
24	BPX1FAI free structure failed	The command attempted to return storage that was returned by TCP/IP in the BPX1GAI call, but the BPX1FAI call to free the storage failed. Refer to message HWSV4600E, which is included in the command output.

Example of the UPDATE IMSCON TYPE(RMTCICS) command

In the following example, the UPDATE IMSCON TYPE(RMTCICS) command is used to start the remote CICS connection CONNECT2.

TSO SPOC input:

```
UPD IMSCON TYPE(RMTCICS) NAME(CICS4) START(COMM)
```

TSO SPOC output:

```
RmtCICS  MbrName          CC
CICS4    HWS1              0
```

OM API input:

```
CMD ( UPD IMSCON TYPE(RMTCICS) NAME(CICS4) START(COMM) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2012.230 23:13:59.998357</statime>
<stotime>2012.230 23:14:00.007729</stotime>
<staseq>CA08E99671795E1A</staseq>
<stoseq>CA08E99673C31A9A</stoseq>
<rqsttkn1>USRT001 10161359</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(RMTCICS) NAME(CICS4) START(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="RCICS" llbl="RmtCICS" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
```

```
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RCICS(CICS4   ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Related reference:

"QUERY IMSCON TYPE(RMTCICS) command" on page 244

"QUERY IMSCON TYPE(ISCUSER) command" on page 190

"UPDATE IMSCON TYPE(ISC) command" on page 1133

RMTCICS statement (System Definition)

UPDATE IMSCON TYPE(RMTIMSCON) command

Use the UPDATE IMSCON TYPE(RMTIMSCON) command to enable communications for the remote IMS Connect connection after it has been stopped, or to stop the communications of the remote IMS Connect connection.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 1181
- "Equivalent WTOR and z/OS commands" on page 1182
- "Output fields" on page 1183
- "Return, reason, and completion codes" on page 1183
- "Examples" on page 1186

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:
- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►──┬─UPDATE─┬──IMSCON──TYPE(RMTIMSCON)──NAME(──┬─┬─rmtimscon_name──┬─┬──)───►
    └─UPD────┘                                  │ └─────────,◄──────┘ │
                                                └─rmtimscon_name*──────┘

►──┬─START(COMM)─┬──────────────────────────────────────────────────────►◄
   └─STOP(COMM)──┘
```
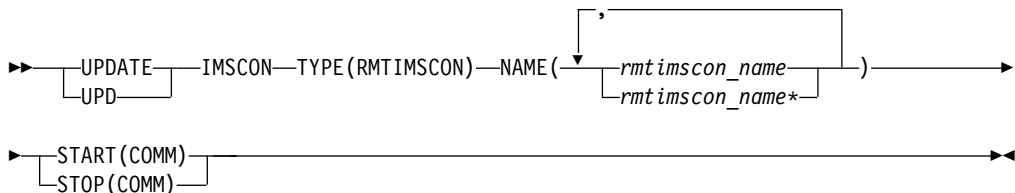
Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(RMTIMSCON) command.

NAME

> Specifies one or more remote IMS Connect connection names to be updated. You can specify a single remote IMS Connect connection name or a list of remote IMS Connect connection names separated by commas. Wildcards can be used in the names.

START(COMM) | STOP(COMM)

> Mutually exclusive keywords that you use to start or stop communications for the remote IMS Connect connection.

> **START(COMM)**
>
> > Starts communications for the remote IMS Connect connection. After issuing this command, OTMA communications with the remote IMS Connect are enabled. To enable communications for MSC, you might need to additionally issue the UPDATE IMSCON TYPE(MSC) START(COMM) command in order to complete the MSC physical link to the remote IMS Connect if MSC communications were disabled with the UPDATE IMSCON TYPE(MSC) STOP(COMM) command.

> **STOP(COMM)**
>
> > Stops communications for the remote IMS Connect connection.
> >
> > When the UPDATE IMSCON TYPE(RMTIMSCON) STOP(COMM) command is issued, IMS Connect:
> >
> > - Stops all communication with a remote IMS Connect instance on the connection specified in the command.
> > - Informs IMS that the connection to the remote IMS Connect instance has been stopped so that IMS can shut down any affected MSC logical links.
> > - Terminates existing socket connections and prevents new socket connections until the UPDATE RMTIMSCON START(COMM) or an equivalent command is issued.
> > - Changes the status of the connection to STOPPED.
> > - Issues message HWST3505I.
> >
> > After the UPDATE IMSCON TYPE(RMTIMSCON) STOP(COMM) command is issued, if IMS Connect receives a message for the connection in the STOPPED state, IMS Connect returns a NAK response to OTMA and issues message HWST3575W. OTMA retains the message at the front of the queue.

Usage notes

You can issue the UPDATE IMSCON TYPE(RMTIMSCON) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

When the UPDATE IMSCON TYPE(RMTIMSCON) START(COMM) command is issued, IMS Connect resumes communications on the specified connection to a remote IMS Connect instance. IMS Connect changes the status of the connection to NOT ACTIVE until new sockets are opened for communications with the remote instance of IMS Connect.

The UPDATE IMSCON TYPE(RMTIMSCON) START(COMM) command is required only when communications on a connection with a remote IMS Connect

instance have been previously stopped by either a STOPRMT command or an UPDATE RMTIMSCON STOP(COMM) command.

If the specified RMTIMSCON connection is defined with AUTOCONN=Y, when the UPDATE IMSCON TYPE(RMTIMSCON) START(COMM) is issued, IMS Connect automatically creates socket connections to the remote IMS Connect. The number of socket connections IMS Connect opens is determined by the RESVSOC parameter.

OTMA communications can resume as soon as IMS Connect executes the UPDATE IMSCON TYPE(RMTIMSCON) START(COMM) command.

For MSC communications, the UPDATE IMSCON TYPE(RMTIMSCON) START(COMM) command resumes only TCP/IP communications in IMS Connect, and does not change the stopped status of MSC links. If the MSC links have been stopped by an UPDATE IMSCON TYPE(MSC) STOP(COMM) command, a WTOR command STOPMSC, or an IMS Connect z/OS command UPDATE MSC STOP(COMM), after you issue the UPDATE IMSCON TYPE(RMTIMSCON) START(COMM) command, you also need to restart the MSC links in IMS Connect by issuing the UPDATE IMSCON TYPE(MSC) START(COMM) command, the STARTMSC command, or the UPDATE MSC START(COMM) command.

Recommendation: When restarting MSC communications, to prevent IMS from sending MSC messages to IMS Connect before TCP/IP communications have been restarted, always issue the UPDATE IMSCON TYPE(RMTIMSCON) START(COMM) command before issuing the UPDATE IMSCON TYPE(MSC) START(COMM) or an equivalent command.

Use the QUERY IMSCON TYPE(RMTIMSCON) command, the VIEWRMT command, or the QUERY RMTIMSCON command to display information about the current connections to a remote IMS Connect instance that are defined to the local IMS Connect instance.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(RMTIMSCON) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 466. WTOR and IMS Connect z/OS equivalents for the UPDATE IMSCON TYPE(RMTIMSCON) command

UPDATE IMSCON TYPE(RMTIMSCON) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(RMTIMSCON) NAME(*rmtimscon_name*) START(COMM)	STARTRMT *rmtimsconName*	UPDATE RMTIMSCON NAME(*rmtimsconName*) START(COMM)

UPDATE IMSCON TYPE(RMTIMSCON) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(RMTIMSCON) NAME(*rmtimscon_name*) STOP(COMM)	STOPRMT *rmtimsconName*	UPDATE RMTIMSCON NAME(*rmtimsconName*) STOP(COMM)

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) appears for output fields that are always returned. *error* appears for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 467. Output fields for the UPDATE IMSCON TYPE(RMTIMSCON) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 469 on page 1184.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. This field is returned only for an error completion code.
MBR	MbrName	N/A	Identifier of the IMS Connect that built the output line. The identifier is always returned.
RIC	RmtImsCon	N/A	Identifier of the remote IMS Connect connection to be acted upon. The identifier is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(RMTIMSCON) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 468. Return and reason codes for the UPDATE IMSCON TYPE(RMTIMSCON) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(RMTIMSCON) command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code.
X'0C000014'	X'00005008'	The command processor failed to obtain storage via BPEGETM.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 469. Completion codes for the UPDATE IMSCON TYPE(RMTIMSCON) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(RMTIMSCON) command completed successfully for the resources.
A	TRWB creation failed	The command attempted to create a TWU control block for the scheduler controller thread, the transmit thread, or the receive thread for the communication for the remote IMS Connect connection, but the BPECBGET request failed. The command output might also contain the following message: HWST3515W.
C	Create scheduler thread failed	During START(COMM) processing, IMS Connect attempted to create the scheduler controller thread for the communication for the remote IMS Connect connection, but the BPETHDCR request failed. For more information, see the HWST3515W message that is included in the command reply message.
10	No resources found	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active now. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.
11	Resource is stopped	During STOP(COMM) processing, IMS Connect could not stop the communication for the remote IMS Connect connection because the communication for the remote IMS Connect connection is already stopped.

Table 469. Completion codes for the UPDATE IMSCON TYPE(RMTIMSCON) command (continued)

Completion code	Completion code text	Meaning
12	Resource is not stopped	During START(COMM) processing, IMS Connect could not start the communication for the remote IMS Connect connection because the communication for the remote IMS Connect connection is not stopped.
3D	Start process was interrupted	START(COMM) processing was interrupted by another command or because IMS Connect is shutting down. The command output might also contain one or more of the following messages: • HWST3510E • HWST3530W • HWST3540E • HWST3560W • HWST3700E • HWST3730W
42	Started, connection(s) failed	The communication for the remote IMS Connect connection started, but there was a problem with starting at least one of the requested socket connections, which are specified in the RESVSOC parameter of the RMTIMSCON statement in the HWSCFGxx member of the IMS PROCLIB data set. IMS Connect is not trying to establish the failed socket connection again.
43	Started, retrying connection(s)	The communication for the remote IMS Connect connection started, but there was a problem with starting at least one of the requested socket connections, which are specified in the RESVSOC parameter of the RMTIMSCON statement in the HWSCFGxx member of the IMS PROCLIB data set. IMS Connect is trying to establish the failed socket connection again. The command output might also contain one or more of the following messages: • HWST3530W • HWST3540E • HWST3560W • HWST3700E • HWST3730W

Table 469. Completion codes for the UPDATE IMSCON TYPE(RMTIMSCON) command (continued)

Completion code	Completion code text	Meaning
44	Started, retry failed	The communication for the remote IMS Connect connection started, but there was a problem with starting at least one of the requested socket connections, which are specified in the RESVSOC parameter of the RMTIMSCON statement in the HWSCFGxx member of the IMS PROCLIB data set. In addition, the attempt to start a thread to reestablish the failed socket connections also failed.
		The command output might also contain one or more of the following messages: • HWST3530W • HWST3540E • HWST3560W • HWST3700E • HWST3730W
49	Unable to start RMTIMSCON	IMS Connect was not able to start the communication for the remote IMS Connect connection.
		The command output might also contain one or more of the following messages: • HWST3505I • HWST3510E • HWST3700E • HWST3730W
4A	Started, unable to notify OTMA	The communication for the remote IMS Connect connection started, but IMS Connect encountered an error when it attempted to notify OTMA that the communication for the remote IMS Connect connection started.
		The command output might also contain one or more of the following messages: • HWST3530W • HWST3540E • HWST3560W • HWST3700E • HWST3730W

Examples

Example 1 for UPDATE IMSCON TYPE(RMTIMSCON) command

TSO SPOC input:

```
UPDATE IMSCON TYPE(RMTIMSCON) NAME(CONNECT2) START(COMM)
```

TSO SPOC output:

```
RmtImsCon    MbrName   CC
CONNECT2     HWS1       0
```

OM API input:

```
CMD(UPDATE IMSCON TYPE(RMTIMSCON) NAME(CONNECT2) START(COMM))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2010.298 02:55:46.185267</statime>
<stotime>2010.298 02:55:46.189428</stotime>
<staseq>C6C7869B9C4335C2</staseq>
<stoseq>C6C7869B9D474302</stoseq>
<rqsttkn1>USRT001 10195546</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1     </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPD IMSCON TYPE(RMTIMSCON) NAME(CONNECT2) START(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="RIC" llbl="RmtImsCon" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RIC(CONNECT2) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The remote IMS Connect connection for CONNECT2 has been started.

Related reference:

➡️ STARTRMT command (Commands)

➡️ STOPRMT command (Commands)

➡️ IMS Connect UPDATE RMTIMSCON command (Commands)

UPDATE IMSCON TYPE(SENDCLNT) command

Use the UPDATE IMSCON TYPE(SENDCLNT) command to terminate the send socket clients on a specified connection to either a remote IMS Connect instance or a remote CICS subsystem.

Subsections:
- "Environment" on page 1188
- "Syntax" on page 1188
- "Keywords" on page 1188
- "Usage notes" on page 1188
- "Equivalent WTOR and z/OS commands" on page 1189

- "Output fields" on page 1190
- "Return, reason, and completion codes" on page 1190
- "Examples" on page 1192

Environment

The UPDATE IMSCON command is applicable only to IMS Connect. To issue this command, the following conditions must be satisfied:

- IMS Connect must be active and configured to communicate with the Common Service Layer (CSL) Structured Call Interface (SCI).
- A type-2 command environment with Structured Call Interface (SCI) and Operations Manager (OM) must be active.

Syntax

```
►►─┬─UPDATE─┬─IMSCON─TYPE(SENDCLNT)─NAME(─┬─sendclient_name──┬─)──────────────────►
   └─UPD────┘                             └─sendclient_name*─┘

►─┬─RMTCICS(rmtcics_name)──────┬──STOP(COMM)──────────────────────────────────►◄
  └─RMTIMSCON(rmtimscon_name)──┘
```

Keywords

The following keywords are valid for the UPDATE IMSCON TYPE(SENDCLNT) command.

NAME
    Specifies the client ID of the send socket to terminate. You can specify a single send client name or a wildcard name.

RMTCICS
    Specifies the remote CICS connection that the send client is on. Either the RMTCICS keyword or the RMTIMSCON keyword must be specified.

RMTIMSCON
    Specifies the remote IMS Connect connection that the send client is on. Either the RMTCICS keyword or the RMTIMSCON keyword must be specified.

STOP(COMM)
    Terminates the send socket clients for a specified remote IMS Connect connection.

Usage notes

You can issue the UPDATE IMSCON TYPE(SENDCLNT) command only through the Operations Manager (OM) API.

IMS Connect can process IMS Connect type-2 commands only if the IMSplex from which the commands were issued has a status of ACTIVE.

Modifications that are made to IMS Connect resources by use of this command are not saved across restarts of IMS Connect.

Use the UPDATE IMSCON TYPE(SENDCLNT) command to terminate send sockets on a connection to either another IMS Connect instance or a remote CICS subsystem.

To identify the IDs of the send sockets that you need to terminate, use the QUERY IMSCON TYPE(SENDCLNT) type-2 command.

If the connection is to another IMS Connect instance, you can optionally use the following IMS Connect commands to identify send client sockets:
- In the WTOR command format, VIEWRMT
- In the z/OS MODIFY command format, QUERY RMTIMSCON

If the socket connections are used for MSC links, do not use the UPDATE IMSCON TYPE(SENDCLNT) command to terminate the sockets. Issuing the UPDATE IMSCON TYPE(SENDCLNT) command breaks MSC links. Instead, stop MSC links by issuing the IMS command /PSTOP LINK from IMS, which automatically terminates the associated send and receive sockets in IMS Connect.

An alternative method of terminating send sockets that are used for MSC messages is by using the IMS Connect WTOR command STOPLINK or the IMS Connect z/OS modify command DELETE LINK. These commands terminate the send sockets and receive sockets that are used by the MSC link. The IMS type-2 command equivalent is UPDATE IMSCON TYPE(LINK).

If a socket connection is used for OTMA messages and the send socket connection is in RECV state when the UPDATE IMSCON TYPE(SENDCLNT) command is issued, IMS Connect issues a NAK to OTMA and then terminates the send socket connection. OTMA reroutes the message to the dead letter queue, HWS$DLQ.

Do not use the UPDATE IMSCON TYPE(SENDCLNT) command to terminate socket connections that are used for ISC links. The UPDATE IMSCON TYPE(SENDCLNT) command breaks ISC links. Instead, stop ISC links by issuing the IMS command /QUIESCE NODE USER from IMS, which automatically terminates the associated send and receive sockets in IMS Connect.

Equivalent WTOR and z/OS commands

The following table lists IMS Connect WTOR (Write to Operator with Reply) and IMS Connect z/OS commands that perform similar functions as the UPDATE IMSCON TYPE(SENDCLNT) command.

Notes:
- IMS Connect WTOR commands are replies to the outstanding IMS Connect reply message.
- IMS Connect z/OS commands are issued through the z/OS (MVS) interface by using the IMS Connect *jobname*.

Table 470. WTOR and MS Connect z/OS equivalents for the UPDATE IMSCON TYPE(SENDCLNT) command

UPDATE IMSCON TYPE(SENDCLNT) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(SENDCLNT) NAME(*sendclient_name*) RMTIMSCON(*rmtimscon_name*) STOP(COMM)	STOPSCLN *rmtimscon sendclient*	DELETE RMTIMSCON NAME(*rmtimsconname*) SENDCLNT(*clientid*)

Table 470. WTOR and MS Connect z/OS equivalents for the UPDATE IMSCON TYPE(SENDCLNT) command (continued)

UPDATE IMSCON TYPE(SENDCLNT) command	Equivalent IMS Connect WTOR command	Equivalent IMS Connect z/OS command
UPDATE IMSCON TYPE(SENDCLNT) NAME(*sendclient_name*) RMTCICS(*rmtcics_name*) STOP(COMM)	STOPSCLN *rmtcics sendclient*	No equivalent z/OS command

Output fields

Short label
> Contains the short label that is generated in the XML output.

Long label
> Contains the column heading that is displayed on the TSO SPOC screen.

Keyword
> Identifies the keyword on the command that caused the field to be generated. N/A (not applicable) is displayed for output fields that are always returned. *error* is displayed for output fields that are returned only in the case of an error.

Meaning
> Provides a brief description of the output field.

Table 471. Output fields for the UPDATE IMSCON TYPE(SENDCLNT) command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code that indicates whether IMS Connect was able to process the command for the specified resource. The completion code is always returned. See Table 473 on page 1191.
CCTXT	CCText	*error*	Completion code text that briefly explains the meaning of the nonzero completion code. The CCTXT field is returned only for an error completion code.
RIC	RmtImsCon	RMTIMSCON	Identifier of the remote IMS Connect connection to be acted upon.
RCICS	RmtCICS	RMTCICS	Identifier of the remote CICS connection to be acted upon.
SCL	SendClnt	N/A	Identifier of the client ID of the send socket to be terminated. The SCL field is always returned.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE IMSCON TYPE(SENDCLNT) command are standard for all commands that are entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 472. Return and reason codes for the UPDATE IMSCON TYPE(SENDCLNT) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE IMSCON TYPE(SENDCLNT) command completed successfully. The command output contains a line for each resource, which is accompanied by its completion code.
X'0C000008'	X'00002004'	An invalid keyword or keyword parameter was specified.
X'0C000008'	X'00002010'	A generic name was specified in the RMTIMSCON or RMTCICS keyword. Only specific names can be specified.
X'0C000008'	X'00002014'	An invalid character was specified in the NAME() parameter.
X'0C000008'	X'00002018'	The NAME keyword was not specified, but is a required keyword for the UPDATE IMSCON TYPE(SENDCLNT) command.
X'0C000008'	X'0000201C'	Neither the RMTIMSCON keyword nor the RMTCICS keyword was specified. One of these keywords is required.
X'0C000008'	X'00002020'	The STOP keyword must be specified.
X'0C000008'	X'0000202C'	Both the RMTIMSCON and RMTCICS keywords were specified. Only one of these keywords can be specified.
X'0C000008'	X'00002133'	Multiple names were specified in the RMTIMSCON or RMTCICS keyword. Only one name can be specified.
X'0C00000C'	X'00003000'	The command was successful for some resources but failed for others. The command output contains a line for each resource, which is accompanied by its completion code.
X'0C00000C'	X'00003004'	The command was not successful for any resource. The command output contains a line for each resource, which is accompanied by its completion code.
X'0C000014'	X'00005000'	The command processor failed to obtain storage via BPECBGET.
X'0C000014'	X'00005050'	The command processor failed because of an internal processing error. Refer to any error messages returned with the command or displayed on the system console for more details.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 473. Completion codes for the UPDATE IMSCON TYPE(SENDCLNT) command

Completion code	Completion code text	Meaning
0		The UPDATE IMSCON TYPE(SENDCLNT) command completed successfully for the resources.

*Table 473. Completion codes for the UPDATE IMSCON TYPE(SENDCLNT)
command (continued)*

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	The resource name is unknown to the client that is processing the request. The resource name might have been typed in error or the resource might not be active now. If a wildcard was specified in the command, there were no matches for the name. Confirm that the correct spelling of the resource name is specified on the command.

Examples

*Example of UPDATE IMSCON TYPE(SENDCLNT) RMTIMSCON(rmtimscon)
command*

In the following example, a send client socket, OTM12345, is stopped. The stopped
send socket is on the remote IMS Connect connection, CONNECT2.

TSO SPOC input:
```
UPDATE IMSCON TYPE(SENDCLNT) NAME(OTM12345) RMTIMSCON(CONNECT2) STOP(COMM)
```

TSO SPOC output:
```
SendClient  RmtImsCon    MbrName  CC
OTM12345    CONNECT2     HWS1      0
```

OM API input:
```
CMD(UPDATE IMSCON TYPE(SENDCLNT) NAME(OTM12345) RMTIMSCON(CONNECT2) STOP(COMM))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2010.298 03:01:07.108399</statime>
<stotime>2010.298 03:01:07.112686</stotime>
<staseq>C6C787CDAAA2F767</staseq>
<stoseq>C6C787CDABAEE8C7</stoseq>
<rqsttkn1>USRT001 10200107</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON          </kwd>
<input>UPDATE IMSCON TYPE(SENDCLNT) NAME(OTM12345) RMTIMSCON(CONNECT2)
 STOP(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="n" key="1"
 scroll="no" len="8" dtype="CHAR" align="lfet" skipb="no" />
<hdr slbl="RIC" llbl="RmtImsCon" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
```

```
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>SCL(OTM12345) RIC(CONNECT2) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Example of UPDATE IMSCON TYPE(SENDCLNT) RMTCICS(rmtcics) command

In the following example, a send client socket, ISC98765, is stopped. The stopped
send socket is on the remote CICS connection, CICSA1.

TSO SPOC input:
```
UPDATE IMSCON TYPE(SENDCLNT) NAME(ISC98765) RMTCICS(CICSA1) STOP(COMM)
```

TSO SPOC output:
```
SendClnt RmtCICS  MbrName        CC
ISC98765 CICSA1   HWS1            0
```

OM API input:
```
CMD(UPDATE IMSCON TYPE(SENDCLNT) NAME(ISC98765) RMTCICS(CICSA1) STOP(COMM) )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.6.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2012.339 23:27:56.704048</statime>
<stotime>2012.339 23:27:56.707079</stotime>
<staseq>CA91F7FFA1530C9C</staseq>
<stoseq>CA91F7FFA210779C</stoseq>
<rqsttkn1>USRT001 10152756</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>HWS1    </master>
<userid>USRT001 </userid>
<verb>UPD </verb>
<kwd>IMSCON       </kwd>
<input>UPDATE IMSCON TYPE(SENDCLNT) NAME(ISC98765) RMTCICS(CICSA1)
 STOP(COMM) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="SCL" llbl="SendClnt" scope="LCL" sort="n" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="RCICS" llbl="RmtCICS" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>SCL(ISC98765) RCICS(CICSA1  ) MBR(HWS1            ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Related reference:

- STOPSCLN command (Commands)
- IMS Connect DELETE SENDCLNT command (Commands)

UPDATE LE command

Use the UPDATE LE command to define Language Environment (LE) runtime parameter overrides or to change the system option to enable or disable LE override processing.

The parameters can be filtered by a transaction code, LTERM name, user ID, or program name for MPP and JMP regions. The parameters may be filtered by a program name for IFB, BMP, and JBP regions. Message driven BMP regions can also filter on a transaction code. Any combination of parameters may be used to qualify the application instance to which the runtime parameters are applied. The first available entry in the table is used. The new entry may be added before or after existing entries, depending on where free space exists in the table.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1195
- "Usage notes" on page 1196
- "Output fields" on page 1196
- "Return, reason, and completion codes" on page 1196
- "Examples" on page 1197

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE LE command and keywords can be issued.

Table 474. Valid environments for the UPDATE LE command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE LE	X	X	X
LTERM	X	X	X
PGM	X	X	X
SET	X	X	X
TRAN	X	X	X
USERID	X	X	X

Syntax

```
►►──┬─UPDATE─┬─LE──────────────────────────────────────────────►◄
    └─UPD────┘
```

```
►►─SET(LEOPT(─┬─NO──┬─))─────────────────────────────────────────────►◄
              └─YES─┘
         ┌─────────────,─────────────┐
         │                            │
         ▼─┬─LTERM(ltermname)──┬──┬─SET(LERUNOPTS(runtimeparameters))─┘
           ├─PGM(programname)──┤
           ├─TRAN(tranname)────┤
           └─USERID(userid)────┘
```
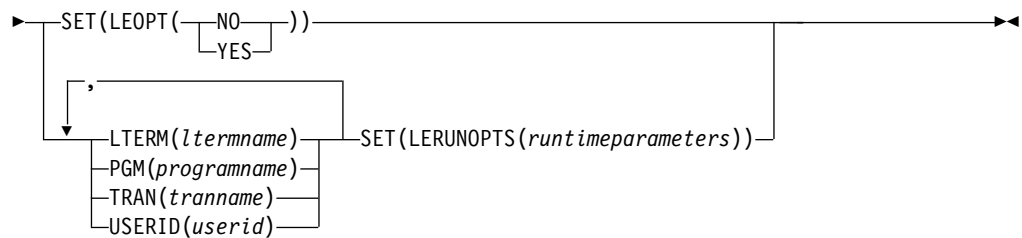
Keywords

The following keywords are valid for the UPDATE LE command:

LTERM()
> Specifies the 1-8 character name of the lterm to which the parameters are to be defined.

PGM()
> Specifies the 1-8 character name of the program to which the parameters are to be defined.

SET() Specifies the updates to attributes or parameters.

> **LEOPT()**
>> Specifies whether LE runtime parameters can be overridden dynamically for all active IMS systems in the IMSplex. YES indicates that the IMS systems enable overrides which enables the DL/I INQY call to retrieve runtime parameters. NO indicates that the IMS systems do not enable dynamic overrides to the parameters.
>>
>> This option allows the user to override the LEOPT=Y|N option that is specified in the DFSCGxxx PROCLIB member without having to bring down the IMS system.
>>
>> When runtime overrides are disabled (IMS is started with LEOPT=N or the UPD LE SET(LEOPT(NO)) command is issued) the runtime parameter table continues to be updated as UPD LE SET(LERUNOPTS()) or DEL LE commands are issued. If the UPD LE SET(LEOPT(YES)) command is then specified, all changes that were made during the time overrides were disabled are available.
>>
>> This command is automatically routed to all IMS systems that are active in the IMSplex. The user can not override the OM routing to route to a single IMS.
>
> **LERUNOPTS()**
>> Specifies the LE dynamic runtime parameters. If an existing set of parameters is found for the specified TRAN, LTERM, USERID, or PGM, the new parameter string completely replaces the existing parameter string. The parameters are not appended to the existing string. The string is only replaced when the specified filters are an exact match for the existing entry.
>>
>> If there is no existing entry, then the first unused entry that is large enough to contain the parameters is used for the update. An unused entry is one that was previously deleted by the DEL LE command. If there are no unused entries or none that are large enough, storage is allocated for a new entry and the entry is added to the top of the table.

The UPDATE LE SET(LERUNOPTS()) command is processed regardless of the LEOPT system option. This means that IMS continues to build the runtime parameter table entries even though they will not be retrieved through the DL/I INQY call.

This command is automatically routed to all IMS systems that are active in the IMSplex. The user cannot override the parameters on a single IMS.

TRAN()
Specifies the 1-8 character name of the transaction to which the parameters are to be defined.

USERID()
Specifies the 1-8 character name of the user ID to which the parameters are to be defined.

Usage notes

This command may be specified only through the Operations Manager API. The command syntax for this command is defined in XML and is available to automation programs that communicate with OM.

OM overrides the routing on the command and routes the command to all IMS systems in the IMSplex. The user specified route list is ignored.

At least one of the resource filters (TRAN, LTERM, USERID, or PGM) must be specified.

Output fields

The following table shows the UPDATE LE output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. ERR appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 475. Output fields for the UPDATE LE command

Short label	Keyword	Meaning
CC	N/A	Completion code for the line of output. Completion code is always returned.
MBR	N/A	IMSplex member (IMS identifier) that built the output line. Member name is always returned.

Return, reason, and completion codes

The OM return and reason codes that may be returned as a result of this command are standard for all commands entered through the OM API. An IMS return and reason code is returned to OM by the UPDATE LE command.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE LE command completed successfully.
X'00000008'	X'00002008'	No keywords were specified on the command. At least one keyword is required. When SET(LERUNOPTS()) is specified, at least one resource name must be specified.
X'00000008'	X'00002010'	An asterisk or percentage sign was specified in the filter name. Wildcards are not supported on the command.
X'00000008'	X'00002014'	An invalid character was specified in the filter name.
X'00000008'	X'00002040'	An invalid filter was specified on the command. When SET(LEOPT()) is specified, no other resource names can be specified.
X'00000010'	X'00004040'	The parameter override header has not been initialized. Retry the command after restart is complete.
X'00000014'	X'00005000'	Unable to get storage from IMODULE GETSTOR.
X'00000014'	X'00005010'	Unable to obtain latch.
X'00000014'	X'00005FFF'	Internal IMS error - should not occur.

The following table contains the completion code that can be returned on a UPDATE LE command.

Table 476. Completion code for the UPDATE LE command

Completion code	Meaning
0	The UPDATE LE command completed successfully for the specified resource.

Examples

The following are examples of the UPDATE LE command.

Example 1 for UPDATE LE command

Assume the following filters and parameters are specified on UPD LE SET(LERUNOPTS()) commands that are processed in the order listed.
1. TRAN(PART) LTERM(TERM2) SET(LERUNOPTS(hhhh))
2. TRAN(PART) LTERM(TERM2) SET(LERUNOPTS(iiii))
3. LTERM(TERM2) USERID(BETTY) SET(LERUNOPTS(gggg))
4. TRAN(PART) LTERM(TERM1) USERID(BOB) SET(LERUNOPTS(ffff))
5. TRAN(PART) LTERM(TERM1) USERID(BARBARA) SET(LERUNOPTS(eeee))
6. PGM(DFSSAM02) SET(LERUNOPTS(dddd))
7. TRAN(PART) LTERM(TERM1) SET(LERUNOPTS(cccc))
8. TRAN(PART) USERID(BETTY) SET(LERUNOPTS(bbbb))
9. TRAN(PART) PGM(DFSSAM02) SET(LERUNOPTS(aaaa))

Rules for matching an entry which results in an update of an existing entry:
- The number of filters defined on the UPDATE LE must match the number of filters defined in the entry.
- The filter values defined on the UPDATE LE must be an exact match for those defined in the entry.

The following table is a logical representation of the parameter override table entries at the end of the command processing. The table includes the transaction name, LTERM, USERID, Program, and LERUNOPTS for each entry.

Table 477. Parameter override table entries for UPDATE LE example 1

Entry#	TRAN	LTERM	USERID	PROGRAM	LERUNOPTS
1	PART			DFSSAM02	aaaa
2	PART		BETTY		bbbb
3	PART	TERM1			cccc
4				DFSSAM02	dddd
5	PART	TERM1	BARBARA		eeee
6	PART	TERM1	BOB		ffff
7		TERM2	BETTY		gggg
8	PART	TERM2			iiii

Example 2 for UPDATE LE command

TSO SPOC input:

```
UPD LE TRAN(IAPMDI26) USERID(USRT001)
SET(LERUNOPTS(RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR)))
```

TSO SPOC output:

```
MbrName    CC
SYS3        0
```

OM API input:

```
CMD(UPD LE TRAN(IAPMDI26) USERID(USRT001)
SET(LERUNOPTS(RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR))))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.1.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2002.163 17:56:10.220516</statime>
<stotime>2002.163 17:56:10.221547</stotime>
<staseq>B7C4CA4EDBFE420E</staseq>
<stoseq>B7C4CA4EDC3EB382</stoseq>
<rqsttkn1>USRT002 10105610</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>SYS3    </master>
<userid>USRT002 </userid>
<verb>UPD </verb>
<kwd>LE             </kwd>
<input>UPD LE TRAN(IAPMDI26) USERID(USRT001)
```

```
 SET(LERUNOPTS(RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR)))
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="A" key="1" scroll="NO"  len="8"
dtype="CHAR" align="left" />
<hdr slbl="CC"  llbl="CC"      scope="LCL" sort="N" key="0" scroll="YES" len="4"
dtype="INT" align="right" />
</cmdrsphdr>
<cmdrspdata>
<rsp>MBR(SYS3    ) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE LE command adds an entry to the LE runtime options table. The entry added by this command defines two filters: user ID and transaction. The transaction is set to IAPMDI26 and the user ID is set to USRT001. The runtime options string for this table entry is RPTOPTS=((ON),NOOVR),RPTSTG=((OFF),NOOVR). The output shows that IMS member SYS3 processed the command with a return code of 0.

Related concepts:

▸ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

▸ Command keywords and their synonyms (Commands)

UPDATE MSLINK command

Use the UPDATE MSLINK command to set or change the MSC logical link attributes or to change the status of specified logical links.

The UPDATE MSLINK command can be either a type-1 or type-2 command.

Subsections:
-
-
-
-
-
-
-

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE MSLINK command and keywords can be issued.

Table 478. Valid environments for the UPDATE MSLINK command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
UPDATE MSLINK	X		X
NAME	X		X

Table 478. Valid environments for the UPDATE MSLINK command and keywords (continued)

Command / keywords	DB/DC	DBCTL	DCCTL
SET	X		X
START	X		X
STOP	X		X
OPTION	X		X

Syntax

Type-1 command syntax

Type-2 command syntax

A:

B:

C:

```
          ┌──────,──────┐
├──STOP(──▼──COMM────┬──)──┬──────┬──────────────────────────────────┤
             ├──TRACE──┤      └──│ D │──┘
             └──TKOTRC──┘
```

D:

```
             ┌──────,──────┐
├──OPTION(──▼──CHKPT────┬──)─────────────────────────────────────────┤
                ├──NORESET─┤
                ├──RESET───┤
                └──FORCE───┘
```
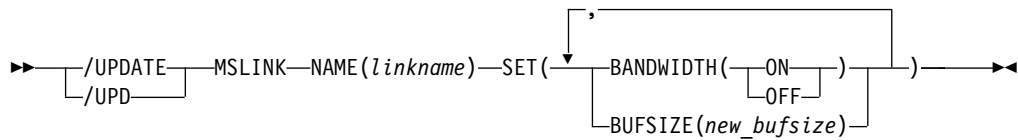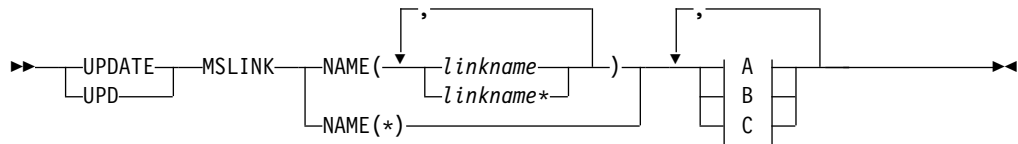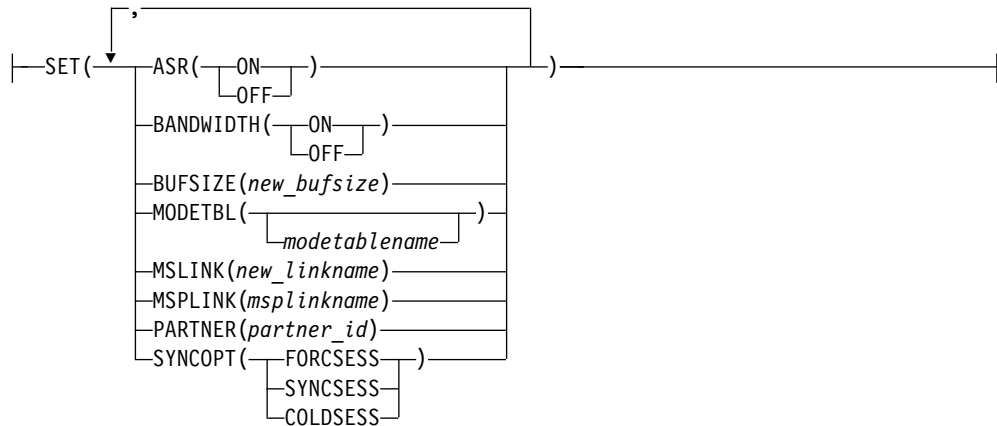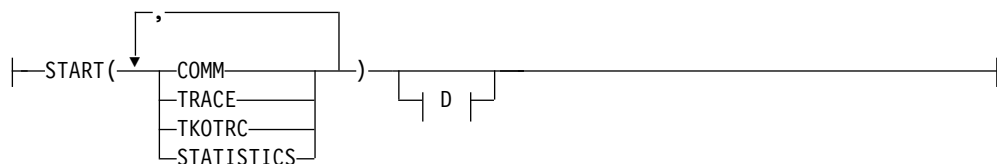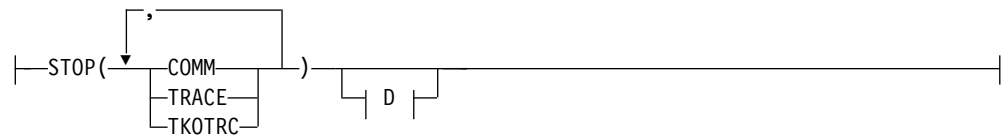
Keywords

The following keywords are valid for the UPDATE MSLINK command:

NAME()
>   Specifies the 1- to 8-character name of the MSC logical links to be processed. You can specify wildcard parameters (*) for the NAME keyword. The logical link names are processed if they match the wildcard name. NAME(*) and NAME(abc*) are examples of valid names.

SET()
>   Specifies the updates to attributes or parameters. You cannot specify both the SET() parameter and the STOP(COMM) parameter.

>   **ASR()**
>>   The Automatic Session Restart (ASR) designation of a VTAM link changes when the Session Outage Notification facility of VTAM is used. The ASR parameter allows you to override the system definition option that is defined for the automatic session restart designation of a link.

>>   You cannot change the designation of ASR for non-VTAM MSC links.

>>   Stop the logical link before you change the ASR designation.

>>   **ON** Specifies that the logical link uses automatic session restart. Automatic session restart is not necessarily activated for a link if a status of ASR is displayed for that link. You must also specify SONSCIP=YES on the APPL definition statement for VTAM when defining the network to VTAM. ASR works only if both sides of the link are using the ASR option.

>>   **OFF**
>>>   Specifies that the logical link does not use automatic session restart.

>   **BANDWIDTH()**
>>   Specifies whether the logical link uses bandwidth mode.

>>   The BANDWIDTH keyword does not apply to logical links assigned to a TCP/IP-type physical link. TCP/IP physical links always operate in bandwidth mode. If SET(BANDWIDTH(OFF)) is specified for a TCP/IP physical link, the command is rejected.

There is no system definition option to enable bandwidth mode. By default, logical links are initialized with bandwidth mode off.

To change the bandwidth mode, first stop the logical link. The link must not display a status ERE.

ON Specifies that the logical link uses bandwidth mode.

OFF
Specifies that the logical link does not use bandwidth mode.

BUFSIZE()
Changes the input and output buffer sizes for the logical link. The minimum buffer size is 1024, and the maximum buffer size is 65536.

The logical link must be stopped before changing the BUFSIZE value. Buffer sizes for logical links follow the same specification requirements as buffer sizes specified on the MSPLINK definition by using the BUFSIZE= keyword.

MODETBL()
The default mode table name of a link changes if you specify the MODETBL() parameter without the START(COMM) parameter. This default is usually established by system definition. To reset this field to its null state (as if a mode table name was not specified at system definition), specify MODETBL().

If you specify both the START(COMM) parameter with the MODETBL() parameter, then the specified mode table name is used only for the START(COMM) session. When the session is terminated, the link mode table name reverts to the default mode table name.

The mode table name determines the mode table entries to be used when activating a VTAM MSC session. You cannot use the MODETBL keyword with non-VTAM sessions.

The logical link must be stopped before changing the MODETBL value.

MSLINK()
Specifies the 1- to 8-alphanumeric new link name for the specified logical link. If you specify the MSLINK parameter, you can specify only one NAME parameter. The new link name cannot already exist as a logical link name. If the command is successful, the link name specified on the NAME keyword is no longer valid.

The logical link must be stopped before changing the MSLINK name.

If the command is successful for a logical link in an environment where the IMSRSC repository is enabled, the MSC logical link paths that reference the old logical link name are updated with the new logical link name, marked as updated, and marked as needing to be exported. The new logical link and the updated logical link paths are automatically exported at the next IMS checkpoint. If the old MSC logical link name definition exists in the IMSRSC repository, delete the definition from the repository before the logical link with the updated name is exported to the repository at the next IMS checkpoint. Delete the old MSC logical link name definition from the IMSRSC repository by using the DELETE DEFN command.

MSPLINK()
Enter the 1- to 8-character name of the MSC physical link to which you

want to assign this logical link. You assign the logical link to the physical link for input and output purposes. The physical link name specified must already exist.

The logical link must be stopped before changing the MSPLINK. If the logical link is being assigned to either a TCP/IP physical link or a VTAM physical link, then the physical link must also be stopped. TCP/IP and VTAM physical links can be stopped by issuing the UPD MSPLINK NAME(*plnkname*) STOP(LOGON). If the TCP/IP physical link is used for TCP/IP generic resources, the physical link can be stopped with the UPD MSPLINK NAME(*plnkname*) STOP(GENLOGON) command.

If the logical link is being assigned to a TCP/IP physical link, bandwidth mode is turned on for the logical link. Bandwidth mode is always on for TCP/IP physical links and cannot be turned off.

If the logical link is being assigned to a CTC or MTM link, then the logical link that is currently assigned to the target physical link must also be stopped.

PARTNER()
Specifies the 2-character alphabetic partner identification. This partner identification ensures that the two related logical links in two systems are always logically and physically connected. Both systems must have logical links with the same partner ID. The new partner ID specified must not already exist.

Only one logical link can be specified in the NAME keyword to change the PARTNER keyword.

The logical link must be stopped before changing the PARTNER keyword.

SYNCOPT()
Specifies the system definition option to be overridden for the named logical link. This keyword allows the user to override the system definition option defined for forcing resynchronization until the next UPDATE MSLINK command is issued or IMS is cold started.

SYNCOPT() is valid only for TCP/IP and VTAM links.

The logical link must be stopped before changing SYNCOPT.

FORCSESS
Specifies that the session initiation is to be completed regardless of the agreement between session restart modes and the message sequence numbers. Use of the FORCSESS parameter could cause messages to be lost or duplicated.

SYNCSESS
Specifies that the session initiation is to be completed only when the session restart modes and message sequence numbers agree.

COLDSESS
Resets the session restart mode to COLD by resetting all the MSLINK dynamic flags and fields to zero, which allows cold session initiation of the link. Use the COLDSESS parameter when attempts to warm start the session fail.

Use of the COLDSESS parameter could cause messages to be lost or duplicated.

Before issuing the command with the COLDSESS option, stop the link first. When COLDSESS is specified, it should be used on both sides of the link.

The COLDSESS parameter does not override the current setting of SYNCSESS or FORCSESS.

START()
Starts the specified MSC logical links, or starts tracing of the specified logical links.

COMM
Starts the specified MSC logical links and allows IMS to start sending and receiving messages.

Communication between IMS systems does not begin until an UPDATE MSLINK command is entered in both systems for CTC or MTM links, or in one of the systems for either a TCP/IP or a VTAM link.

The COMM parameter cannot be specified in both START() and STOP(). The logical link must be stopped and idle, and the assigned physical link must be open, as shown in the QUERY command output.

TRACE
Turns on the tracing of internal IMS events that are related to the specified logical links. Unlike the equivalent type-1 /TRACE command, the trace level cannot be specified on the UPDATE command. Instead, IMS uses the default level of 4. The status and options of the current IMS traces can be displayed with the QUERY MSLINK command. TRACE cannot be specified in both START() and STOP().

Note: The type-2 command UPDATE MSLINK NAME(*linkname*) START(TRACE) uses the same level and module settings that were used the last time the /TRACE SET (ON) LINK command was issued. If a /TRACE SET (ON) LINK command has not been issued since the last cold start, this command defaults to MODULE=ALL and LEVEL=4.

TKOTRC
Enables tracing during takeover. This tracing is separate from regular tracing. This keyword applies only in an XRF environment. TRACE and TKOTRC cannot both be specified under START(). TKOTRC cannot be specified in both START() and STOP().

STATISTICS
Provides statistics support to MSC logical links. These statistics can be used to determine the link performance, adjust the link and message queue buffer sizes to optimum sizes, and increase the link performance. Each logical link (MSLINK) has a work area where statistics are kept.

STOP()
Stops the specified MSC logical links or stops tracing of the specified logical links.

COMM
Stops the specified MSC logical links, and allows IMS to stop sending and receiving messages. The partner link in another IMS system stops itself. COMM cannot be specified in both START() and STOP(). You cannot specify both STOP(COMM) and SET() together.

TRACE
Turns off the tracing of internal IMS events that are related to the specified

logical links. The status and options of the current IMS traces can be displayed with the QUERY MSLINK command. TRACE cannot be specified in both START() and STOP().

TKOTRC()
Disables tracing during takeover. This tracing is separate from regular tracing. This keyword applies only in an XRF environment. TKOTRC cannot be specified in both START() and STOP().

OPTION()
Specifies options for the UPDATE MSLINK command. OPTION is valid only with START(STATISTICS) or STOP(COMM).

FORCE
Can be specified only with STOP(COMM) for CTC, TCP/IP, and VTAM links. Use the FORCE parameter when an MSC link does not clean up and assume a PSTOP IDLE status in IMS after normal UPDATE STOP(COMM) processing, even though the session has been terminated.

After using the FORCE parameter, issue the UPDATE MSLINK command with the NAME(linkname | *) START(COMM) options to restart the link.

For TCP/IP links, the FORCE option is useful for shutting down an MSC TCP/IP link that did not shut down normally after the link was shut down in the partner IMS system. You are not required to shut down a link normally before using the FORCE option.

FORCE also notifies the local IMS Connect instance to clean up sessions for TCP/IP links and VTAM to clean up sessions for VTAM links.

IMS performs the following actions when processing the FORCE option for a TCP/IP link:
- Shuts down the link in the IMS where the command is issued
- Notifies the local IMS Connect instance to clean up the send socket
- Issues an error message
- Places the link in ERE IDLE status

For VTAM links, you can use the FORCE parameter with some network commands to clean up the VTAM link within IMS.

RESET | NORESET
Issuing the UPDATE MSLINK() START(STATISTICS) OPTION(RESET) command will reset the link statistics and set the start time to the current time. The logical link does not need to be stopped and idle to reset the statistics values, or the reset mode, as is the case for changing many of the link characteristics, such as buffer size and bandwidth mode.

OPTION(RESET), OPTION(CHKPT,RESET), and OPTION(CHKPT, NORESET) can be specified with START(STATISTICS).

RESET
When specified without CHKPT, this indicates that statistics for the specified logical link should be reset immediately. When specified with CHKPT, this indicates that statistics for the specified logical link should be reset at every system checkpoint. This only applies to statistics reported on the QUERY MSLINK command. Statistics reported in the log records during system checkpoints are cumulative over the life of IMS, and are not reset by the UPDATE MSLINK command.

NORESET
When specified with CHKPT, this indicates that statistics for the

specified logical link should not be reset at any system checkpoint. This only applies to statistics reported on the QUERY MSLINK command. Statistics reported in the log records during system checkpoints are cumulative over the life of IMS, and are not reset by the UPDATE MSLINK command.

CHKPT

When specified with RESET, this indicates that statistics for the specified logical link should be reset at every system checkpoint. When specified with NORESET, this indicates that statistics for the specified logical link should not be reset at any system checkpoint. This only applies to statistics reported on the QUERY MSLINK command. Statistics reported in the log records during system checkpoints are cumulative over the life of IMS, and are not reset by the UPDATE MSLINK command.

OPTION(CHKPT,RESET) and OPTION(CHKPT, NORESET) can be specified with START(STATISTICS).

Usage notes

You can issue the type-2 UPDATE MSLINK command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for the UPDATE MSLINK command is defined in XML and the syntax is available to automation programs that communicate with OM.

The CREATE MSLINK NAME(*mslinkname*) command specified with SET(MSPLINK(*msplinkname*)) is rejected if at least one logical link attribute specified in the command (ASR, BACKUP, MODETBL, or SYNCOPT) conflicts with the physical link type. However, after the logical link is created, it can be reassigned to any type of physical link with an UPDATE MSLINK command or an /MSASSIGN command. If a logical link defined with attributes applicable to specific physical link types is reassigned to a different physical link type that does not support those attributes, they are ignored when the link is started.

If you use the UPDATE MSLINK command to update the following attributes of a logical link, the link is marked as needing to be exported: ASR, BANDWIDTH, BUFSIZE, MODETBL, MSLINK, MSPLINK, PARTNER and SYNCOPT. To display the logical links that need to be exported, issue the QUERY MSLINK SHOW(EXPORTNEEDED) command. If the IMSRSC repository is enabled, the MSC logical link is automatically exported at the next IMS checkpoint.

The UPDATE MSLINK command compared to other commands

The following table shows variations of the UPDATE MSLINK command and the type-1 IMS commands that perform similar functions.

Table 479. Type-1 equivalents for the UPDATE MSLINK command

UPDATE MSLINK command	Similar IMS command
UPDATE MSLINK NAME(linkname | linkname* | *) SET(ASR(ON|OFF))	/CHANGE LINK link # | ALL ASR ON | OFF
UPDATE MSLINK NAME(linkname | linkname* | *) SET(MODETBL(modetablename))	/CHANGE LINK link # | ALL MODE(modename | NONE)
UPDATE MSLINK NAME(linkname) SET(MSLINK(linkname))	No similar type-1 IMS command exists.

Table 479. Type-1 equivalents for the UPDATE MSLINK command (continued)

UPDATE MSLINK command	Similar IMS command
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) SET(MSPLINK(msplinkname))	/MSASSIGN LINK link # MSPLINK msplinkname
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) SET(PARTNER(partner_id))	No similar type-1 IMS command exists.
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) SET(SYNCOPT(FORCSESS))	/CHANGE LINK link#ǀALL FORCSESS
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) SET(SYNCOPT(SYNCSESS))	/CHANGE LINK link# ǀ ALL SYNCSESS
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) SET(SYNCOPT(COLDSESS))	/CHANGE LINK link# ǀ ALL COLDSESS
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) START(COMM)	/RSTART LINK link # ǀ ALL
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) SET(MODETBL(modetablename)) START(COMM)	/RSTART LINK link# ǀ ALL MODE modename
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) START(TRACE)	/TRACE SET ON ǀ OFF LINK link # ǀ ALL
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) START(TKOTRC)	/TRACE SET ON LINK link # ǀ ALL TAKEOVER
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) STOP(COMM)	/PSTOP LINK link # ǀ ALL
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) STOP(COMM) OPTION(FORCE)	/PSTOP LINK link # PURGE or /PSTOP LINK link # FORCE
UPDATE MSLINK NAME(linkname ǀ linkname* ǀ *) STOP(COMM,TRACE) OPTION(FORCE)	/PSTOP LINK link # FORCE /TRA SET OFF LINK link#

Output fields

Short label
> Contains the short label that is generated in the XML output.

Keyword
> Identifies the keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 480. Output field descriptions for the UPDATE MSLINK command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
LINKN	N/A	Logical link number.
MBR	N/A	IMSplex member that built the output line.

Table 480. Output field descriptions for the UPDATE MSLINK command (continued)

Short label	Keyword	Meaning
MSL	N/A	Logical link name.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE MSLINK command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 481. Return and reason codes for the UPDATE MSLINK command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE MSLINK command completed successfully.
X'00000008'	X'0000200C'	No resources to be updated were found. The resource names specified might be invalid or no resources match the specified filter.
X'00000008'	X'00002040'	More than one filter value is specified on the UPDATE MSLINK command.
X'00000008'	X'00002044'	The UPDATE MSLINK command is not processed because the same attribute value was specified for the START and STOP parameters.
X'00000008'	X'00002048'	Invalid SET attribute is specified.
X'00000008'	X'00002133'	The UPDATE MSLINK command is not processed because multiple logical link names were specified, which is invalid for the SET filters specified in the command.
X'00000008'	X'00002134'	TheUPDATE MSLINK command specified SET(MSPLINK(*msplinkname*)), but *msplinkname* does not exist.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'00004000'	Command is not valid during IMS restart.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000402C'	Command is not valid on the non-MSC-capable system.
X'00000014'	X'00005004'	The UPDATE MSLINK command processing terminated as a DFSOCMD response buffer could not be obtained.

Return code	Reason code	Meaning
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005010'	A latch could not be obtained.

Errors that are unique to the processing of the UPDATE MSLINK command are returned as completion codes. A completion code is returned for each action against an individual resource.

If the UPDATE MSLINK command is entered as a type-2 command, a message that contains the completion codes will be issued. The codes listed in the following table are for a type-1 UDPATE MSLINK command.

Table 482. Completion codes for the UPDATE MSLINK command

Completion code	Completion code text	Meaning
0		The UPDATE MSLINK command completed successfully for the resource.
10	NO RESOURCES FOUND	MSLINK name is invalid, or the specified wildcard parameter does not match any resource names.
11	RESOURCE ALREADY EXISTS	The linkname specified in SET(MSLINK(linkname)) already exists as a logical link.
62	HIOP STORAGE ERROR	IMS was unable to obtain storage from the HIOP pool.
8D	RESOURCE IS NOT STOPPED	The logical link is not stopped. The link must be stopped for the updates specified.
100	INV SET KEYWORD FOR LINK TYPE	A keyword specified in SET() is invalid for the corresponding physical link type.
101	TARGET MSPLINK NOT STOPPED	The target physical link specified in SET(MSPLINK()) is not stopped.
102	TARGET MSLINK NOT STOPPED	The logical link that is currently assigned to the target physical link specified in SET(MSPLINK()) is not stopped.
103	NEW PARTNER ID ALREADY EXISTS	The new partner ID specified in SET(PARTNER()) already exists.
104	MSLINK NOT ASSIGNED TO MSPLINK	The update cannot be performed because the logical link is not assigned to a physical link.
105	MSPLINK NOT OPEN	The update cannot be performed because the assigned physical link is not open.
106	INVALID OPTION FOR LINK TYPE	A keyword specified in OPTION() is invalid for the corresponding physical link type.
107	OPTION=FORCE ALREADY IN EFFECT	OPTION(FORCE) has already been invoked for the logical link.

Examples

The following are examples of the UPDATE MSLINK command:

Example 1 for UPDATE MSLINK command

TSO SPOC input:

```
UPD MSLINK NAME(STAR1L, STAR2L, STAR3L*) SET(SYNCOPT(FORCSESS),
ASR(OFF),MODETBL(),MSPLINK(STAR1)) START(COMM,TRACE)
```

TSO SPOC output:

```
MSLink    MbrName LinkNum CC CCText
STAR1L    IMSA    12      0
STAR2L    IMSA            10 NOT FOUND
STAR3L    IMSA    5       0
STAR3L1   IMSA    6       0
STAR3L2   IMSA    7       0
```

Explanation: This UPDATE MSLINK command is issued to do the following:

1. Override the system definition option for the named logical links: STAR1L and logical links matching STAR3L* (STAR3L, STAR3L1, and STAR3L2). STAR2L is unknown.
2. Change the automatic session restart designation to OFF for the specified logical links.
3. Reset the mode table name of the specified logical links to NONE.
4. Assign the specified logical links to the physical link: STAR1.
5. Allow IMS to start sending and receiving messages on the specified links.
6. Enable tracing of control block trace information.

Example 2 for UPDATE MSLINK command

TSO SPOC input:

```
UPDATE MSLINK NAME(LNK12V02) SET(BANDWIDTH(ON), BUFSIZE(4096))
```

TSO SPOC output:

```
MSLink         MSLink#   MbrName   CC
LNK12V02          10     IMS1       0
```

Explanation: This command changes the buffer size to 4096 and turns on the bandwidth.

Related concepts:

☞ How to interpret CSL request return and reason codes (System Programming APIs)

Related tasks:

☞ Diagnosing link problems by using MSC link statistics (Diagnosis)

Related reference:

☞ Command keywords and their synonyms (Commands)

☞ List of commands with similar functions for multiple resources (Operations and Automation)

UPDATE MSNAME command

Use the UPDATE MSNAME command to set or change the MSC logical link path attributes or to change the status of specified logical link paths.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1212
- "Usage notes" on page 1213
- "Equivalent IMS type-1 commands" on page 1214
- "Output fields" on page 1214
- "Return, reason, and completion codes" on page 1215
- "Examples" on page 1217

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE MSNAME command and keywords can be issued.
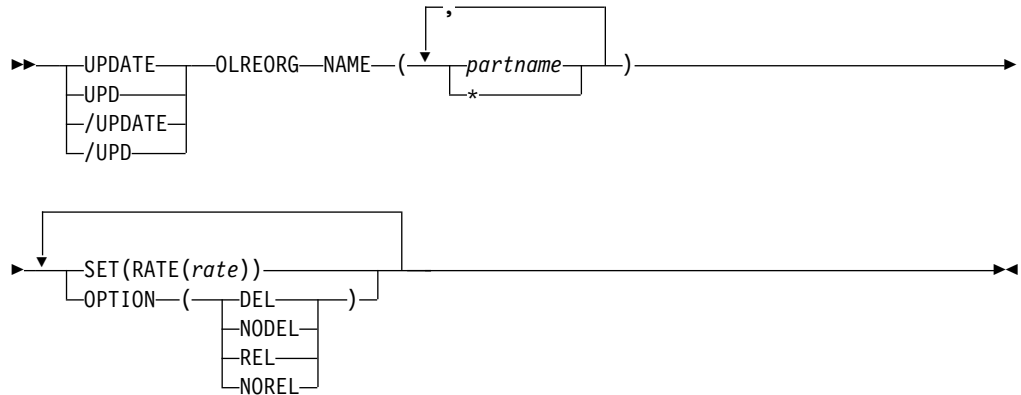
Table 483. Valid environments for the UPDATE MSNAME command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
UPDATE MSNAME	X		X
NAME	X		X
SET	X		X
START	X		X
STOP	X		X

Syntax

A:

B:

```
          ┌─,──────┐
├─START─(──▼──Q────┴──)──────────────────────────────────────────────┤
             └─SEND─┘
```

C:

```
          ┌─,──────┐
├─STOP─(───▼──Q────┴──)──────────────────────────────────────────────┤
             └─SEND─┘
```
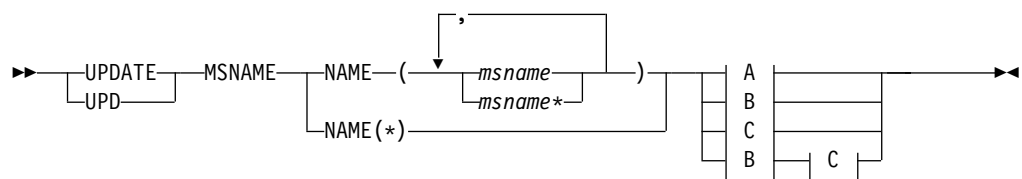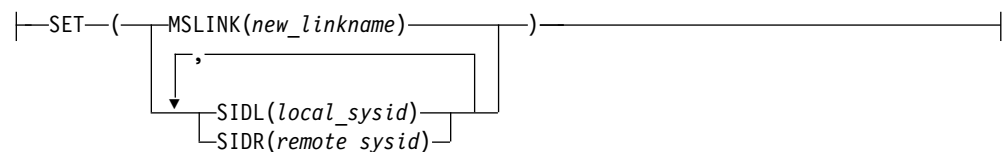
Keywords

The following keywords are valid for the UPDATE MSNAME command:

NAME()
Specifies the 1- to 8-character name of the MSC logical link path to be processed. You can specify wildcards (*) for the NAME keyword. NAME(*) and NAME(abc*) are examples of valid names.

SET()
Specifies the updates to attributes or parameters. You can only change attributes for statically-defined MSNAMEs. Dynamic MSNAMEs cannot be changed.

MSLINK()
Assigns the specified logical link path to the logical link for input and output purposes. Enter the 1- to 8-character name of the MSC logical link to which you want to assign this logical link path. The logical link name specified must already exist.

Both the source logical link (where the MSNAME is currently assigned) and the target logical link (where the MSNAME is being assigned to) must be stopped.

SIDL()
Specifies the system identification of a local system in a multiple system configuration. The specified value must be a number from 1 to 2036. The new specified value must be currently undefined or already defined as a local SID on this system. The SID cannot already be defined as remote.

You can change the SIDL value for only one logical link path on a command. Therefore, if you specify the SIDL parameter, you can specify only one NAME parameter.

The logical link and logical link path must be stopped to change the SIDL.

To change the SIDR or SIDL value of a logical link path:
1. Verify the existing SID values of the logical link path of either /DIS ASMT MSNAME or QUERY MSNAME SHOW(SYSID).

    For SIDL verification only, the operator must verify that the current SIDL value is not the lowest SIDL value in the system. If it is the lowest SIDL value in the system, and there are no other logical link paths in the system with the same SIDL, then the SIDL cannot be changed because the lowest SIDL in the system cannot change.
2. Verify there are no transactions with the existing SID values of either /DIS SYSID TRAN or QUERY TRAN SHOW(MSNAME).

3. If any transactions were found in step 2, assign the transactions to a new valid system of either /MSASSIGN TRAN TO MSNAME, UPDATE TRAN SET(MSNAME()) or UPDATE TRAN SET(SIDR(),SIDL()).

4. Change the SID values of the logical link path using UPDATE MSNAME SET(SIDR(),SIDL()). However, this is not allowed if the new SIDL value is lower than the current lowest SIDL in the system. The lowest SIDL in the system cannot change, and must remain the lowest SIDL in the system.

SIDR()

Specifies the system identification of a remote system in a multiple system configuration. The value specified must be a number from 1 to 2036. The new value specified must be currently undefined on this system (it cannot be already defined as remote or local).

You can change the SIDR value for only one logical link path on a command. Therefore, if you specify the SIDR parameter, you can specify only one NAME parameter.

The logical link and logical link path must be stopped to change the SIDR.

To change the SIDR or SIDL value of a logical link path:

1. Verify the existing SID values of the logical link path of either /DIS ASMT MSNAME or QUERY MSNAME SHOW(SYSID).

2. Verify there are no transactions with the existing SID values of either /DIS SYSID TRAN or QUERY TRAN SHOW(MSNAME).

3. If any transactions were found in step 2, assign the transactions to a new valid system of either /MSASSIGN TRAN TO MSNAME, UPDATE TRAN SET(MSNAME()) or UPDATE TRAN SET(SIDR(),SIDL()).

4. Change the SID values of the logical link path using UPDATE MSNAME SET(SIDR(),SIDL()).

START()

Specifies the attributes to start. You cannot specify the START and STOP parameters with the same attributes.

Q Starts the queuing to the logical link path and starts input for the specified logical link path.

SEND

Starts the sending of messages to the specified logical link path.

STOP()

Specifies the attributes to stop. You cannot specify the START and STOP parameters with the same attributes.

Q Stops the queuing to the logical link path and stops input for the specified logical link path.

SEND

Stops the sending of messages to the specified logical link path.

Usage notes

You can issue this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

If you use the UPDATE MSNAME command to update the following attributes of an MSC logical link path, the logical link path is marked as needing to be exported: MSLINK, SIDL and SIDR. To display the logical link paths that need to be exported, issue the QUERY MSNAME SHOW(EXPORTNEEDED) command. If the IMSRSC repository is enabled, the MSC logical link path is automatically exported at the next IMS checkpoint.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE MSNAME command and the type-1 IMS commands that perform similar functions.

Table 484. Type-1 equivalents for the UPDATE MSNAME command

UPDATE MSNAME command	Similar IMS type-1 command
UPDATE MSNAME NAME(msname) SET(MSLINK(linkname))	/MSASSIGN MSNAME msname TO LINK link #
UPDATE MSNAME NAME(msname) SET(SIDR(remote_sysid))	No similar type-1 IMS command exists.
UPDATE MSNAME NAME(msname) SET(SIDL(local_sysid))	No similar type-1 IMS command exists.
UPDATE MSNAME NAME(msname | msname* | *) START(Q,SEND)	/START MSNAME msname | msname* | ALL
UPDATE MSNAME NAME(msname | msname* | *) STOP(Q,SEND)	/STOP MSNAME msname | msname* | ALL
UPDATE MSNAME NAME(msname | msname* | *) STOP(Q),START(SEND)	/PURGE MSNAME msname | msname* | ALL

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 485. Output field descriptions for the UPDATE MSNAME command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
MBR	N/A	IMSplex member that built the output line.
MSN	N/A	Logical link path name.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE MSNAME command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 486. Return and reason codes for the UPDATE MSNAME command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE MSNAME command completed successfully.
X'00000008'	X'0000200C'	No resources to be updated were found. The resource names specified might be invalid or no resources match the specified filter.
X'00000008'	X'00002040'	More than one filter value is specified on the UPDATE MSNAME command.
X'00000008'	X'00002044'	The UPDATE MSNAME command is not processed because the same attribute value was specified for the START and STOP parameters.
X'00000008'	X'00002126'	The value of SIDL() is invalid. Valid values are 1-2036.
X'00000008'	X'00002128'	The value of SIDR() is invalid. Valid values are 1-2036.
X'00000008'	X'00002136'	The name specified in MSLINK() is not a valid logical link name.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'00004000'	Command is not valid during IMS restart.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'0000402C'	Command is not valid on the non-MSC-capable system.
X'00000014'	X'00005004'	The UPDATE MSNAME command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'00005010'	A latch could not be obtained.

Errors that are unique to the processing of the UPDATE MSNAME command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 487. Completion codes for the UPDATE MSNAME command

Completion code	Completion code text	Meaning
0		The UPDATE MSNAME command completed successfully for the resource.
10	NO RESOURCES FOUND	MSNAME name is invalid, or the specified wildcard parameter does not match any resource names.
45	INVALID SIDR VALUE	The value specified in SIDR() is invalid. The value must be currently undefined in this system.
83	INVALID SIDL VALUE	The value specified in SIDL() is invalid. The value must be currently undefined or local in this system.
B7	USE MANAGER ERROR	IMS encountered an internal error using the USE manager.
102	TARGET MSLINK NOT STOPPED	The update cannot complete because the link specified in MSLINK() is not stopped.
109	INVALID FOR DYNAMIC MSNAME	The update cannot complete because the logical link path specified in NAME() is dynamic. The update is not valid for dynamic logical link paths that are built in a shared queues system, which represent logical link paths defined only on other IMS systems in the shared queues group.
10A	SOURCE MSLINK NOT STOPPED	The update cannot complete because the logical link to which the specified logical link path is assigned is not stopped.
10B	MSC RESOURCE IS IN USE	The update cannot complete because there is in-doubt message activity for a remote transaction or lterm related to the logical link path.
10D	INVALID SIDL, LESS THAN LOW SIDL	The value specified in SIDL() is invalid. The value cannot be lower than the current lowest local SID in this system.
10E	LAST MSNAME FOR LOW SIDL	The current local SIDL value cannot be changed for this logical link path because the current SIDL value is the lowest local SIDL value, and this is the only logical link path that has this local SIDL value.
10F	SIDL STILL LOCAL IN REMOTE TRAN	The current local SIDL value cannot be changed for this logical link path because there are remote transactions defined with this local SIDL value, and this the only logical link path that has this local SIDL value.
110	SIDR STILL LOCAL IN REMOTE TRAN	The current remote SIDR value cannot be changed for this logical link path because there are remote transactions defined with this remote SIDR value.

Examples

The following are examples of the UPDATE MSNAME command:

Example 1 for UPDATE MSNAME command

TSO SPOC input: UPD MSNAME NAME(STAR1N1,STAR2N1) STOP(Q,SEND)

TSO SPOC output:

```
MSName  MbrName CC
STAR1N1 IMSA    0
STAR2N1 IMSA    0
```

Explanation: This UPDATE MSNAME command is issued to stop queuing and the sending of messages to the named logical link paths. This allows the attributes to be changed by another UPD MSNAME command.

Example 2 for UPDATE MSNAME command

TSO SPOC input:
```
UPD MSNAME NAME(STAR1N1) SET(SIDR(10),SIDL(20))
```

TSO SPOC output:

```
MSName  MbrName CC
STAR1N1 IMSA    0
```

Explanation: This UPDATE MSNAME command is issued to do the following:
- Identify the remote system that is represented by SIDR(10).
- Identify the local system as SIDL(20) for routing messages back to this system.

Example 3 for UPDATE MSNAME command

TSO SPOC input:
```
UPD MSNAME NAME(STAR1N1) SET(MSLINK(STAR3L))
```

TSO SPOC output:

```
MSName  MbrName CC
STAR1N1 IMSA    0
```

Explanation: This UPDATE MSNAME command is issued to assign the specified logical link path STAR1N1 to STAR3L.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

➡ /MSASSIGN command (Commands)

➡ List of commands with similar functions for multiple resources (Operations and Automation)

"/START MSNAME command" on page 853

"/STOP MSNAME command" on page 901

Chapter 4, "/PURGE command," on page 33

UPDATE MSPLINK command

Use the UPDATE MSPLINK command to set or change the MSC physical link attributes or to change the status of the specified physical links.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1219
- "Usage notes" on page 1223
- "Equivalent IMS type-1 commands" on page 1224
- "Output fields" on page 1224
- "Return, reason, and completion codes" on page 1225
- "Examples" on page 1226

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE MSPLINK command and keywords can be issued.

Table 488. Valid environments for the UPDATE MSPLINK command and keywords

Command / keywords	DB/DC	DBCTL	DCCTL
UPDATE MSPLINK	X		X
NAME	X		X
SET	X		X
START	X		X
STOP	X		X

Syntax

```
          ►──START(──┬─LOGON──────────┬──)────────────────────────────────────►◄
          │          ├─GENLOGON────────┤
          │          └─LOGON,GENLOGON──┘
          ├──STOP(──┬─LOGON──────────┬──)──────────────────
          │         ├─GENLOGON────────┤
          │         └─LOGON,GENLOGON──┘
          │         ┌──,─────────────────────────────────┐
          └──SET(───▼──┬─ADDR(addr)──────────────────────────┬──)──
                       ├─ASR(──┬─ON──┬──)────────────────────┤
                       │       └─OFF─┘                        │
                       ├─BACKUP(──┬─n──┬──)──────────────────┤
                       │          └─NO─┘                      │
                       ├─BUFSIZE(──size──)───────────────────┤
                       ├─ICONPLKID(TCPIP_IMS_Connect_LCLPLKID)┤
                       ├─IMSCON(TCPIP_local_IMS_Connect)──────┤
                       ├─MODETBL(──┬────────────────┬──)──────┤
                       │           └─modetablename──┘         │
                       ├─MSPLINK(new_msplinkname)─────────────┤
                       ├─NODE(VTAM_node_name)─────────────────┤
                       ├─RMTIMS(TCPIP_remote_IMS_ID)──────────┤
                       └─SESSION(n)───────────────────────────┘
```
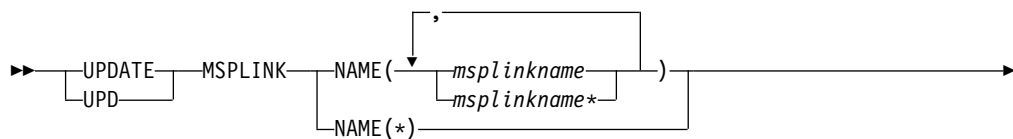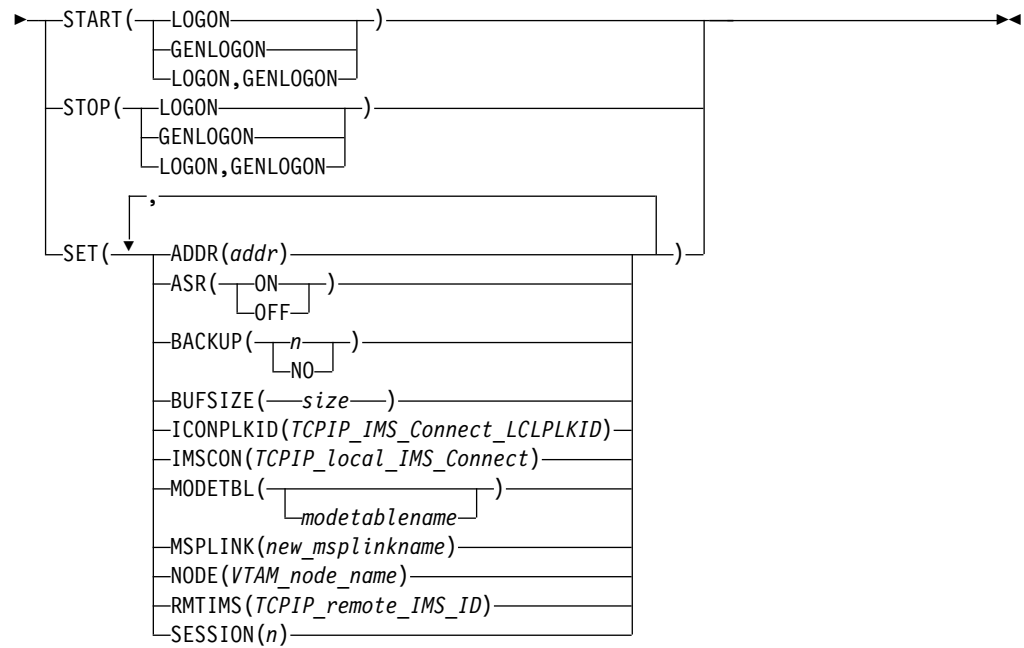
Keywords

The following keywords are valid for the UPDATE MSPLINK command:

NAME()

Specifies the 1- to 8-character name of the MSC physical link that is processed. You can specify wildcards (*) in the name. NAME(*) and NAME(abc*) are examples of valid names.

SET()

Sets the attributes of the physical link. The required and optional parameters differ depending on the type of physical link that you are defining.

ASR(), MODETBL() and NODE() are VTAM link specific parameters.

ADDR()

Specifies the address of the channel-to-channel adapter. The address value can be 3 or 4 hexadecimal digits. This operand is valid only for CTC link types.

The ADDR value must be unique among all CTC physical links that are defined in the IMS system.

Stop the assigned logical link before you change the ADDR value.

All devices that are attached to the same channel as the CTC adapter must be accessible through an alternative channel. In addition, do not attach system resources, for example, paging devices, to the same channel as the CTC adapter.

ASR()

The Automatic Session Restart (ASR) designation for each logical link assigned to the specified VTAM physical link changes when the Session Outage Notification facility of VTAM is used. The ASR parameter allows you to override the system definition option that is defined for the automatic session restart designation of a link.

You cannot change the designation of ASR for non-VTAM MSC links.

Stop the physical link and each assigned logical link before you change the ASR designation.

ON Specifies that the logical link uses automatic session restart. Automatic session restart is not necessarily activated for a link if a status of ASR is displayed for that link. SONSCIP=YES must also be specified on the APPL definition statement for VTAM when defining the network to VTAM. ASR works only if both sides of the link use the ASR option.

OFF
Specifies that the logical link does not use automatic session restart.

BACKUP()
For XRF-capable IMS systems, controls the automatic restart of TCP/IP and VTAM links after an XRF takeover. Before you change the BACKUP value, stop the physical link and each assigned logical link .

n A numeric integer in the range 1 - 7 that sets the priority for reestablishing the session.

NO Specifying BACKUP=NO suppresses the automatic restart of the MSC physical link.

Although BACKUP prioritizes the order in which IMS restarts TCP/IP and VTAM links, the active requests might be completed in any order because of variables such as internal VTAM conflicts and pacing, and Internet Protocol network traffic.

BACKUP is an MSC physical link attribute that you can set by using the CREATE MSPLINK, CREATE MSLINK, or UPDATE MSPLINK command. You can display the physical link BACKUP value by using the QUERY MSPLINK and QUERY MSLINK commands.

Important: Even though BACKUP is a physical link attribute, the CREATE MSLINK NAME(*linkname*) SET(MSPLINK(*msplinkname*)) command can also change the BACKUP value of the specified physical link. You can use the QUERY MSPLINK command to display the updated BACKUP value. You can also use the QUERY MSLINK command and specify the MSC logical links, including the newly created logical link, that are assigned to the physical link to display the updated BACKUP value.

BUFSIZE()
Specifies the input and output buffer sizes for each logical link that is assigned to this physical link. The valid range of buffer sizes for all link types is 1024 - 65536 bytes. The same buffer size must be specified by the IMS systems at each end of a physical link.

For VTAM and TCP/IP links, stop the physical link and each assigned logical link before you change the BUFSIZE value.

The same buffer size must be specified by the IMS systems at each end of a physical link. If the buffer sizes are not equal, the logical link will fail to restart with message DFS3218.

IMS initializes the MSC link buffers to the size specified on the BUFSIZE parameter.

If you are using bandwidth mode, a BUFSIZE value of 1024 is too small to send multiple messages with one buffer. Specify a value of at least 4096.

You can change the size of a buffer of an individual logical link by using the type-2 command UPDATE MSLINK NAME (*linkname*) SET(BUFSIZE(*new_bufsize*)).

ICONPLKID()
For TCP/IP physical links, defines a 1- to 8-character alphanumeric ID by which IMS Connect knows the MSC physical link. This ID must match the ID defined to IMS Connect on the LCLPLKID parameter of the MSC configuration statement that defines the physical link to IMS Connect.

You can use the ICONPLKID keyword to modify the MSC physical link to reference a different IMS Connect MSC configuration statement.

This ID can also be specified on the LCLPLKID keyword of the MSPLINK macro.

IMSCON()
For TCP/IP physical links, specifies the IMSplex name of the local IMS Connect instance that manages the TCP/IP connections for this physical link. IMSCON accepts a 1- to 8- alphanumeric value.

You can use the IMSCON keyword to switch the IMS Connect instance that an MSC physical link uses.

MODETBL()
Changes the name of the default VTAM logon mode table entry (logon mode table) for this physical link. This changes the name for each logical link currently assigned to this physical link. This name is usually established by system definition. To reset this field to its null state (as if no mode table name was specified at system definition), specify MODETBL().

You cannot use the SET(MODETBL) parameter for a non-VTAM MSC link.

The physical link and each assigned logical link must be stopped before changing the MODETBL.

MSPLINK()
Specifies the 1- to 8-alphanumeric new physical link name for this physical link. This keyword also allows you to change the name of a reserved link to a real name and activate it. The name specified in MSPLINK() must be unique. If the command is successful, the old physical link name specified on the NAME keyword is no longer valid.

Only one NAME parameter can be specified to change the MSPLINK name.

The physical link must be stopped before changing the MSPLINK.

If the command is successful for a physical link in an environment where the IMSRSC repository is enabled, the logical links that reference the old physical link name are updated with the new physical link name, marked as updated, and marked as needing to be exported. The new physical link and the updated logical links will be automatically exported at the next IMS checkpoint. If the old MSC physical link name definition exists in the IMSRSC repository, it should be deleted with a DELETE DEFN command.

NODE()
Specifies a 1- to 8-alphanumeric name for the remote VTAM node (APPLID) name for the VTAM line. This node is the VTAM node name of the remote IMS system at the other end of the link. This node is the label on the VTAM APPL statement for the remote IMS system. For an MSC

VTAM link communicating with an XRF complex, the node name must be the VTAM USERVAR that is associated with the partner XRF complex.

The physical link and each assigned logical link must be stopped before changing the NODE.

RMTIMS()

For TCP/IP physical links, defines the remote IMS system that this physical link connects to. Enter the 1- to 8- alphanumeric IMSID of the remote IMS system.

You can use the RMTIMS keyword to change the remote IMS system that an MSC physical link references.

SESSION()

For TCP/IP and VTAM physical link types only, specifies the number of parallel sessions that can be active for the physical link. Enter a value in the range 1 - 1018.

If you specify a larger number than you currently need, you can dynamically assign more logical links to the physical link later.

Because a logical parallel session uses the same amount of storage for control blocks and buffers as a physical link, allocating many parallel sessions can use an excessive amount of common storage area. Predetermine how much common storage area you are likely to need.

If logical links are already assigned to the physical link and the value you specify for SESSION is less than the current number of assigned logical links, the number of logical links that can be started is limited to the new SESSION value specified. You will be unable to start the remaining logical links assigned to the physical link.

START()

Specifies the attributes to start.

GENLOGON

For IMS systems that participate in a TCP/IP generic resource group, enables the IMS systems that process this command to resume accepting logical link connection requests to the generic resource group on the specified physical link.

START(GENLOGON) is independent of LOGON, which is used for links that do not use the TCP/IP generic resource group. If STOP(LOGON) has been previously issued, you must issue START(LOGON), either separately or with the GENLOGON parameter, before the IMS system can resume starting and accepting logical links that do not use TCP/IP generic resources.

LOGON

For MSC TCP/IP and VTAM physical link types only. For TCP/IP-type links, START(LOGON) allows the starting of the logical links assigned to the physical link. For VTAM-type links, START(LOGON) enables logical links to logon to the physical link. START(LOGON) and STOP(LOGON) are mutually exclusive.

START(LOGON) is independent of GENLOGON, which is used for links that use a TCP/IP generic resource group. If STOP(GENLOGON) has been previously issued, you must issue START(GENLOGON), either separately or with the LOGON parameter, before the IMS system can resume starting and accepting logical links that use TCP/IP generic resources.

STOP()
  Specifies the attributes to stop.

  GENLOGON
    For IMS systems that participate in a TCP/IP generic resource group, prevents the IMS systems that process this command from either starting or accepting the specified TCP/IP physical link.

    The STOP(GENLOGON) keyword is applicable only to TCP/IP-type physical links that are defined in IMS systems that participate in a TCP/IP generic resource group.

    The STOP(GENLOGON) does not affect links that are already in session.

    Use STOP(GENLOGON) to control where the restart message from a remote link restart is accepted within the local IMSplex. Issue STOP(GENLOGON) against every IMS system in the TCP/IP generic resource group, except the IMS system with which you need to establish link affinity.

    After an UPDATE MSPLINK NAME(*plnkname*) STOP(GENLOGON) command is issued, the QUERY MSPLINK command displays a local status of STOGENLGN.

    STOP(GENLOGON) does not prevent the IMS system from accepting links that do not use the generic resource group. GENLOGON is independent of LOGON and both can be specified together on the STOP() keyword to prevent the IMS system from starting or accepting all links, regardless of whether they use TCP/IP generic resources.

  LOGON
    For MSC TCP/IP and VTAM physical link types only.

    For TCP/IP links, STOP(LOGON) prevents the IMS systems that process this command from starting of the physical link or accepting start requests for the physical link from a link partner. The LOGON keyword does not affect TCP/IP physical links that are used for TCP/IP generic resources.

    The LOGON keyword is independent of the GENLOGON keyword that is used for physical links that are used with TCP/IP generic resources. Both keywords can be specified together on the STOP() keyword to prevent the IMS system from starting or accepting all links, regardless of whether they use TCP/IP generic resources.

    For VTAM links, STOP(LOGON) prevents logons to the physical link. START(LOGON) and STOP(LOGON) are mutually exclusive.

    After an UPDATE MSPLINK NAME(*plnkname*) STOP(LOGON) command is issued, the QUERY MSPLINK command displays a local status of STOLGN.

    The STOP(LOGON) does not affect links that are already in session.

Usage notes

You can issue this command only through the Operations Manager (OM) API. This command applies to DB/DC and DCCTL systems.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

| TYPE cannot be updated. If you want to update the MSC physical link type, first
| delete it with the command DELETE MSPLINK, and then create it again with the
| command CREATE MSPLINK SET(TYPE()).

| With the MTM type link, you can connect IMS systems that are running in the
| same z/OS system without a hardware link.

| UPDATE MSPLINK for attributes ADDR, ASR, BACKUP, BUFSIZE, ICONPLKID,
| IMSCON, MODETBL, MSPLINK, NODE, RMTIMS, and SESSION marks the MSC
| physical link as needing to be exported. This can be displayed with the QUERY
| MSPLINK SHOW(EXPORTNEEDED) command. If the IMSRSC repository is
| enabled, the MSC physical link will be automatically exported at the next IMS
| checkpoint.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE MSPLINK command and the
type-1 IMS commands that perform similar functions.

Table 489. Type-1 equivalents for the UPDATE MSPLINK command

UPDATE MSPLINK command	Similar IMS type-1 command
UPDATE MSPLINK NAME(*msplinkname*) SET(ASR(ON | OFF))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname*) SET(ICONPLKID(*iconplkid*))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME*msplinkname*) SET(IMSCON(*imsconname*))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname*) SET(MODETBL(*modetablename*))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname*) SET(MSPLINK(*new_msplinkname*))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname*) SET(NODE(*VTAM_node_name*))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname*) SET(RMTIMS(*rmtims*))	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname* | *) START(LOGON)	/RSTART MSPLINK *msplinkname*	ALL
UPDATE MSPLINK NAME(*msplinkname* | *) STOP(LOGON)	/PSTOP MSPLINK *mplinkname*	ALL
UPDATE MSPLINK NAME(*msplinkname* | *) START(GENLOGON)	No similar type-1 IMS command exists.
UPDATE MSPLINK NAME(*msplinkname* | *) STOP(GENLOGON)	No similar type-1 IMS command exists.

Output fields

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated.

error appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning

Provides a brief description of the output field.

Table 490. Output field descriptions for the UPDATE MSPLINK command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
MBR	N/A	IMSplex member that built the output line.
MSPL	N/A	Physical link name.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE MSPLINK command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 491. Return and reason codes for the UPDATE MSPLINK command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE MSPLINK command completed successfully.
X'00000008'	X'00002040'	More than 1 filter value is specified on the UPDATE MSPLINK command.
	X'00002154'	Invalid ADDR value.
	X'00002164'	The node cannot be the local VTAM node name.
	X'00002169'	The RMTIMS value cannot be the same as the local system ID.
	X'0000216A'	Invalid SESSION value.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the following table for details on completion codes.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
	X'0000402E'	Command is valid only on an XRF system.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.

Table 491. Return and reason codes for the UPDATE MSPLINK command (continued)

Return code	Reason code	Meaning
X'00000014'	X'00005004'	The UPDATE MSPLINK command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.

Errors that are unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 492. Completion codes for the UPDATE MSPLINK command

Completion code	Completion code text	Meaning
0		The UPDATE MSPLINK command completed successfully for the resource.
10	NO RESOURCES FOUND	MSPLINK name is invalid, or the specified wildcard parameter does not match any resource names.
11		The link name specified in MSPLINK() already exists as a physical link.
8D		The physical link is not stopped. The link must be stopped for the updates specified.
100	INV SET KEYWORD FOR LINK TYPE	A keyword specified in SET() is invalid for the corresponding physical link type.
10C	INVALID START/STOP FOR LINK TYPE	A keyword specified in START() or STOP() is invalid for the corresponding physical link type.
200	NO DDNAME AVAILABLE FOR CRE	All DDNAMEs from DFSM00000 to DFSM9999 are in use.
201	UNABLE TO ALLOCATE DEVICE	Dynamic allocation of CTC device failed.
202	UNABLE TO OPEN CTC DEVICE	CTC DCB open failed.
203	ADDR IS NOT A CTC DEVICE	The device at the specified address is not a CTC adapter.
204	UNAVAILABLE CTC DEVICE	The CTC device at the address specified on the ADDR parameter is not available.
21B	UNABLE TO DEALLOCATE DEVICE	Dynamic deallocation of CTC device failed.
21C	UNABLE TO CLOSE CTC DEVICE	CTC DCB close failed.

Examples

The following are examples of the UPDATE MSPLINK command:

Example 1 for UPDATE MSPLINK command

TSO SPOC input:
```
UPD MSPLINK NAME(STAR1) SET(MODETBL(LOGON12B), NODE(APPL12B), ASR(ON))
```

TSO SPOC output:

```
MSName    MbrName CC
STAR1     IMSA     0
```

Explanation: This UPDATE MSPLINK command is issued to do the following:

1. Change the VTAM logon mode table entry to LOGON12B.
2. Change the VTAM node name of the remote system at the other end of the link to APPL12B.
3. Override the automatic session restart definitions established by system definition.

Example 2 for UPDATE MSPLINK command

TSO SPOC input:

```
UPD MSPLINK NAME(STAR1, STAR2, STAR2B*) START(LOGON)
```

TSO SPOC output:

```
MSName    MbrName CC    CCTEXT
STAR1     IMSA     0
STAR2     IMSA    10    NOT FOUND
STAR2B    IMSA     0
STAR2B11  IMSA     0
STAR2B12  IMSA     0
```

Explanation: This command enables logon for the specified physical links: STAR1, STAR2B, STAR2B11, and STAR2B12. STAR2 is unknown.

Example 3 for UPDATE MSPLINK command

TSO SPOC input:

```
UPD MSPLINK NAME(PLNK12TA) STOP(LOGON,GENLOGON)
```

TSO SPOC output:

```
MSName    MbrName CC    CCTEXT
PLNK12TA  IMS1     0
```

Explanation: This command prevents IMS1 from starting or accepting any and all logical links on the specified physical link PLNK12TA. The GENLOGON applies to links that use TCP/IP generic resources. The LOGON parameter applies to links that specify IMS1 explicitly.

Issuing the following QUERY MSPLINK NAME(PLNK12TA) SHOW(ALL) command in IMS1 returns the following status:

```
MSPLink   MbrName    CC  Type    RmtIms   LclImsCon  LclPlkID   LclStat
PLNK12TA  IMS1        0  TCPIP   IMS2     HWS1       MSC12      STOGENLGN
```

Example 4 for UPDATE MSPLINK command

TSO SPOC input:

```
UPD MSPLINK NAME(*) STOP(LOGON,GENLOGON)
```

TSO SPOC output:

```
MSPLink  MbrName CC  CCText

PLNK12C  IMS1    10C INVALID START/STOP FOR LINK TYPE
PLNK12M  IMS1    10C INVALID START/STOP FOR LINK TYPE
PLNK12TA IMS1     0
PLNK12V  IMS1     0
```

Explanation:

In the example above each link is a different MSC physical link type:

- PLNK12C is a CTC link
- PLNK12M is an MTM link
- PLNK12TA is a TCPIP link
- PLNK12V is a VTAM link

The command is rejected for the CTC and MTM link types, because they do not support the LOGON and GENLOGON command options. After the command is processed, only the TCPIP and VTAM type links have the STOGENLGN and STOLGN status, as shown in the following example output of the QUERY MSPLINK NAME(*) SHOW(STATUS) command:

```
MSPLink  MbrName CC LclStat

PLNK12C  IMS1    0
PLNK12M  IMS1    0
PLNK12TA IMS1    0  STOGENLGN
PLNK12V  IMS1    0  STOLGN
```

Related concepts:

⮕ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

⮕ Command keywords and their synonyms (Commands)

⮕ List of commands with similar functions for multiple resources (Operations and Automation)

UPDATE ODBM commands

Use the UPDATE ODBM commands to change Open Database Manager (ODBM) settings.

Subsections:

- "UPDATE ODBM START(CONNECTION) command"
- "UPDATE ODBM START(TRACE) command" on page 1232
- "UPDATE ODBM STOP(CONNECTION) command" on page 1234
- "UPDATE ODBM STOP(TRACE) command" on page 1237
- "UPDATE ODBM TYPE(CONFIG) command" on page 1240

UPDATE ODBM START(CONNECTION) command

Use the UPDATE ODBM START(CONNECTION) command to start connections to data stores and aliases for Open Database Manager (ODBM).

Subsections:

- "Environment" on page 1229

- "Syntax"
- "Keywords"
- "Usage notes"
- "Output fields" on page 1230
- "Return, reason, and completion codes" on page 1230
- "Examples" on page 1231

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax

```
►►──┬─UPDATE─┬──ODBM──START(CONNECTION)─────────────────────────────►
    └─UPD────┘
```

```
              ┌─────,────┐
►──┬─DATASTORE(─▼─datastore_name─┴─)─┬──────────────────────────────►◄
   │            ┌────,───┐          │
   └─ALIAS(─────▼─alias_name─┴─)────┘
```
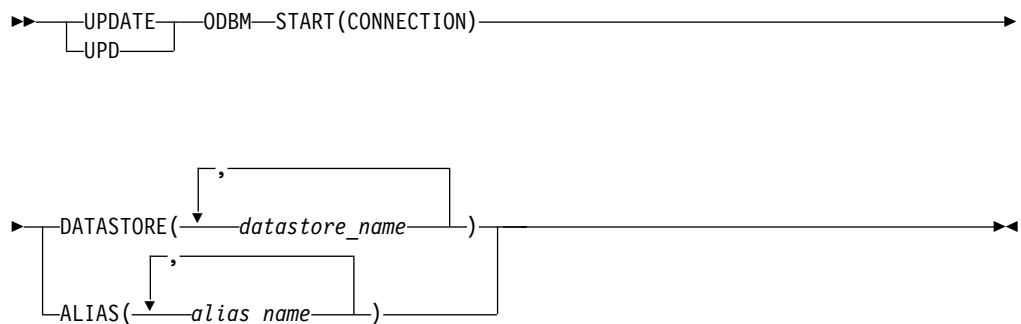
Keywords

The following keywords are valid for the UPDATE ODBM START(CONNECTION) command.

ALIAS()
    Specifies the 1–4 character alias name. Wildcards can be specified for the alias name. You can specify more than one alias name.

    There is no default for ALIAS(). Specifying the ALIAS keyword will start ODBM connections to all the aliases specified with *alias_name*. ALIAS() and DATASTORE() are mutually exclusive.

DATASTORE()
    Specifies the 1–4 character data store name. Wildcards can be specified for the data store name. You can specify more than one data store names.

    There is no default for DATASTORE(). Specifying the DATASTORE keyword will start ODBM connections to all the data stores specified with *datastore_name* as well as their associated aliases. DATASTORE() and ALIAS() are mutually exclusive.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 493. Output field descriptions for the UPDATE ODBM START(CONNECTION) command

Short label	Keyword	Meaning
ALIAS	ALIAS	The 1–4 character alias name that is associated with an IMS data store connection.
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DSTR	N/A	Data store name.
MBR	N/A	Name of the ODBM member that processed the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE ODBM START(CONNECTION) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 494. Return and reason codes for the UPDATE ODBM START(CONNECTION) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPD ODBM START(CONNECTION) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 495. Completion codes for the UPDATE ODBM START(CONNECTION) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM START(CONNECTION) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
3	DATASTORE NOT AVAILABLE	An attempt to start a data store connection failed. The data store is unavailable.
4	DATASTORE CONNECTION FAILED	An attempt to start a data store connection failed. Message CSL4006W accompanies this completion code and explains the AIB return code, AIB reason code, and AIB error extension. (See note)
5	DATASTORE NOT RRS CAPABLE	An attempt to start a data store connection failed. The data store is not RRS capable.
10	NO RESOURCES FOUND	No resources were found.

Note: The command response is encapsulated in <rsp> and </rsp> tags. Message CSL4006W is returned within <msg> and </msg> tags.

Examples

Example 1 for UPDATE ODBM START(CONNECTION) command

TSO SPOC input:
```
UPD ODBM START(CONNECTION) DATASTORE(*)
```

TSO SPOC output:
```
MbrName     DatastoreName   CC
ODBM010D    IMS1             0
ODBM010D    IMS2             0
ODBM020D    IMS3             0
```

Explanation: The UPDATE command will start connections with all data stores that are known to ODBM.

Example 2 for UPDATE ODBM START(CONNECTION) command

TSO SPOC input:
```
UPD ODBM START(CONNECTION) ALIAS(IO1A,IO1B)
```

TSO SPOC output:
```
MbrName     AliasName   CC      DatastoreName
ODBM010D    IO1A         0      IMS1
ODBM010D    IO1B         0      IMS1
```

Explanation: The UPDATE command will start ODBM connections with the ODBM members whose alias names are either IO1A or IO1B.

UPDATE ODBM START(TRACE) command

Use the UPDATE ODBM START(TRACE) command to request that ODBM traces be started.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Output fields"
- "Return, reason, and completion codes" on page 1233
- "Examples" on page 1234

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax

```
►►──┬─UPDATE─┬──ODBM──START(TRACE)──┬─DATASTORE(*)─────────────────────┬──►◄
    └─UPD────┘                      │              ┌─,──────────────┐  │
                                    │              ▼                │  │
                                    └─DATASTORE(─────datastore_name─┴─)┘
```

Keywords

The following keywords are valid for the UPDATE ODBM START(TRACE) command.

DATASTORE()
Specifies the 1–4 character data store name. Wildcards can be specified for the data store name. The *datastore_name* is a repeatable parameter. The default is DATASTORE(*), which applies to all data store names that are known to ODBM.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
    Contains the short label that is generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated.
> *error* appears for output fields that can appear for a non-zero completion
> code. N/A (not applicable) appears for output fields that are always
> returned.

Meaning
> Provides a brief description of the output field.

Table 496. Output field descriptions for the UPDATE ODBM START(TRACE) command

Short label	Keyword	Meaning
ALIAS	N/A	Alias name.
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DSTR	N/A	Data store name.
MBR	N/A	Name of the ODBM member that processed the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE
ODBM START(TRACE) command are standard for all commands entered through
the OM API.

The following table includes the return and reason codes and a brief explanation of
the codes. The return or reason code returned for the command might also indicate
an error from a CSL request.

Table 497. Return and reason codes for the UPDATE ODBM START(TRACE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE ODBM START(TRACE) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes.
A completion code is returned for each action against an individual resource.

Table 498. Completion codes for the UPDATE ODBM START(TRACE) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM START(TRACE) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for UPDATE ODBM START(TRACE) command

TSO SPOC input:

```
UPD ODBM START(TRACE) DATASTORE(IMS1,IMS2)
```

TSO SPOC output:

```
MbrName     DatastoreName    CC
ODBM010D    IMS1              0
ODBM010D    IMS2              0
```

Explanation: The UPDATE command will start ODBM traces for data stores IMS1 and IMS2.

Example 2 for UPDATE ODBM START(TRACE) command

TSO SPOC input:

```
UPD ODBM START(TRACE) DATASTORE(*)
```

TSO SPOC output:

```
MbrName     DatastoreName    CC
ODBM010D    IMS1              0
ODBM010D    IMS2              0
ODBM020D    IMS3              0
```

Explanation: The UPDATE command will start ODBM traces for all data stores that are known to ODBM.

UPDATE ODBM STOP(CONNECTION) command

Use the UPDATE ODBM STOP(CONNECTION) command to stop connections to data stores and aliases for Open Database Manager (ODBM).

Subsections:
- "Environment" on page 1235
- "Syntax" on page 1235
- "Keywords" on page 1235
- "Usage notes" on page 1235
- "Output fields" on page 1236
- "Return, reason, and completion codes" on page 1236

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax

```
►►──┬─UPDATE─┬──ODBM──STOP(CONNECTION)────────────────────────────────►
    └─UPD────┘
```

```
                      ┌─,──────────────────┐
   ►──┬─DATASTORE(──▼──datastore_name───┬──)──┬───────────────────────►◄
      │                                       │
      │            ┌─,───────────────┐        │
      └─ALIAS(──▼──alias_name──────┬──)───────┘
```
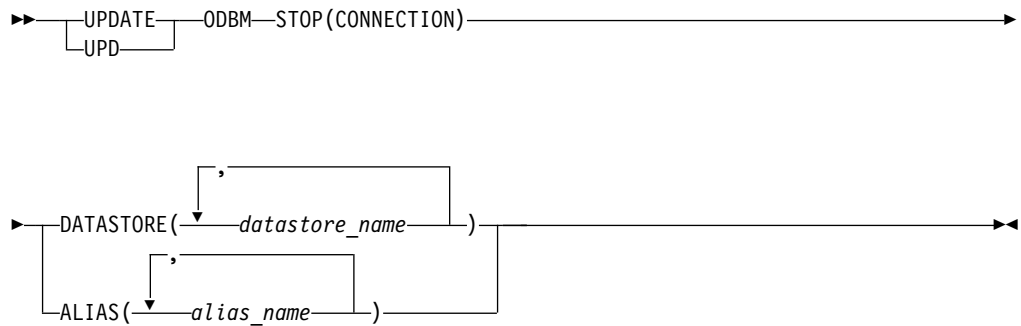
Keywords

The following keywords are valid for the UPDATE ODBM STOP(CONNECTION) command.

ALIAS()
Specifies the 1–4 character alias name. Wildcards can be specified for the alias name. You can specify more than one aliases.

There is no default for ALIAS(). Specifying the ALIAS keyword will stop ODBM connections to all the aliases specified with *alias_name*. ALIAS() and DATASTORE() are mutually exclusive.

DATASTORE()
Specifies the 1–4 character data store name. Wildcards can be specified for the data store name. You can specify more than one data store.

There is no default for DATASTORE(). Specifying the DATASTORE keyword will stop ODBM connections to all the data stores specified with *datastore_name* as well as their associated aliases. DATASTORE() and ALIAS() are mutually exclusive.

To start connections to the data stores, you must issue an UPDATE ODBM START(CONNECTION) DATASTORE() command. Even if the actual data store subsystems are recycled after the UPDATE ODBM STOP(CONNECTION) DATASTORE() command, ODBM will not attempt to connect to the data stores until an UPDATE ODBM START(CONNECTION) DATASTORE() command is issued.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 499. Output field descriptions for the UPDATE ODBM STOP(CONNECTION) command

Short label	Keyword	Meaning
ALIAS	ALIAS	The 1–4 character alias name that is associated with an IMS data store connection.
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DSTR	N/A	Data store name.
MBR	N/A	Name of the ODBM member that processed the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE ODBM STOP(CONNECTION) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 500. Return and reason codes for the UPDATE ODBM STOP(CONNECTION) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE ODBM STOP(CONNECTION) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 501. Completion codes for the UPDATE ODBM STOP(CONNECTION) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM STOP(CONNECTION) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
6	DATASTORE DISCONNECT FAILED	An attempt to stop a data store connection failed. Message CSL4007W accompanies this completion code and explains the AIB return code, AIB reason code, and AIB error text. (See note)
10	NO RESOURCES FOUND	No resources were found.

Note: The command response is encapsulated in <rsp> and </rsp> tags. Message CSL4007W is returned within <msg> and </msg> tags.

Examples

Example 1 for UPDATE ODBM STOP(CONNECTION) command

TSO SPOC input:
```
UPD ODBM STOP(CONNECTION) DATASTORE(IMS*)
```

TSO SPOC output:
```
MbrName     DatastoreName      CC
ODBM010D    IMS1                0
ODBM010D    IMS2                0
ODBM020D    IMS3                0
```

Explanation: The UPDATE command will stop connections with all data stores whose name begin with "IMS".

Example 2 for UPDATE ODBM STOP(CONNECTION) command

TSO SPOC input:
```
UPD ODBM STOP(CONNECTION) ALIAS(IO*)
```

TSO SPOC output:
```
MbrName     AliasName   CC    DatastoreName
ODBM010D    IO1A         0    IMS1
ODBM010D    IO1B         0    IMS1
ODBM010D    IO2A         0    IMS2
ODBM010D    IO2B         0    IMS2
ODBM020D    IO3A         0    IMS3
ODBM020D    IO3B         0    IMS3
```

Explanation: The UPDATE command will stop ODBM connections with the ODBM members whose alias names begin with "IO".

UPDATE ODBM STOP(TRACE) command

Use the UPDATE ODBM STOP(TRACE) command to request that ODBM traces be stopped.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes"
- "Output fields"
- "Return, reason, and completion codes" on page 1239
- "Examples" on page 1240

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax

Keywords

The following keywords are valid for the UPDATE ODBM STOP(TRACE) command.

DATASTORE()
> Specifies the 1–4 character data store name. Wildcards can be specified for the data store name. The *datastore_name* is a repeatable parameter. The default is DATASTORE(*), which applies to all data store names that are known to ODBM.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

Output fields

Short label
> Contains the short label that is generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 502. Output field descriptions for the UPDATE ODBM STOP(TRACE) command

Short label	Keyword	Meaning
ALIAS	N/A	Alias name.
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DSTR	N/A	Data store name.
MBR	N/A	Name of the ODBM member that processed the command.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE ODBM STOP(TRACE) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 503. Return and reason codes for the UPDATE ODBM STOP(TRACE) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE ODBM STOP(TRACE) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 504. Completion codes for the UPDATE ODBM STOP(TRACE) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM STOP(TRACE) command completed successfully.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
10	NO RESOURCES FOUND	No resources were found.

Examples

Example 1 for UPDATE ODBM STOP(TRACE) command

TSO SPOC input:
```
UPD ODBM STOP(TRACE) DATASTORE(*)
```

TSO SPOC output:
```
MbrName    DatastoreName  CC
ODBM010D   IMS1            0
ODBM010D   IMS2            0
ODBM020D   IMS3            0
```

Explanation: The UPDATE command will stop ODBM traces for all data stores that are known to ODBM.

UPDATE ODBM TYPE(CONFIG) command

Use the UPDATE ODBM TYPE(CONFIG) command to update active ODBM configuration.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 1241
- "Output fields" on page 1241
- "Return, reason, and completion codes" on page 1242
- "Examples" on page 1243

Environment

The UPDATE ODBM command is applicable only to the CSL Open Database Manager (ODBM). To issue this command, a CSL type-2 command environment must be enabled and an ODBM instance must be active.

Syntax

```
►►──┬─UPDATE─┬──ODBM──TYPE(CONFIG)──┬──────────────────────────┬──►
    └─UPD────┘                      └─MEMBER(mambername)─┘


►──┬─OPTION(CONNECT)───────┬──────────────────────────────────►◄
   └─OPTION(NOCONNECT)─┘
```
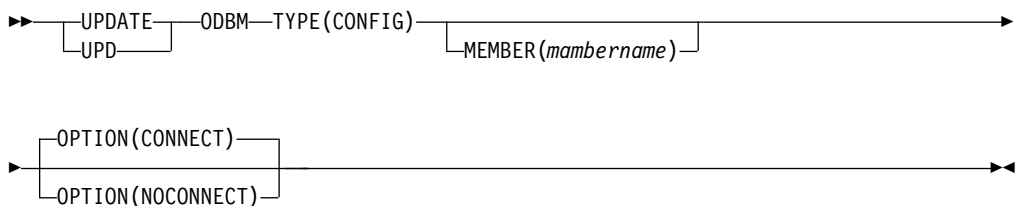
Keywords

The following keywords are valid for the UPDATE ODBM TYPE(CONFIG) command.

MEMBER()
Specifies a 3–character CSLDCxxx suffix that identifies which ODBM configuration PROCLIB member is used to update the active ODBM configuration. The default is the current CSLDCxxx member.

Note: If a new CSLDCxxx member is used to update the active ODBM configuration, the new member becomes the current ODBM configuration PROCLIB member. Also, when restarting ODBM, the ODBMCFG= parameter on the EXEC statement or within the CSLDIxxx initialization PROCLIB member will be used to configure ODBM.

OPTION()
> Specifies whether you want connections with the data stores to be attempted after the active ODBM configuration has been updated.
>
> **CONNECT**
>> Specifies that after updating the active ODBM configuration, all data stores within the new configuration will be connected and all aliases associated with the data stores will be made available for use. CONNECT is the default.
>
> **NOCONNECT**
>> Specifies that after updating the active ODBM configuration, no connections will be attempted. To start connections to the data stores, you must issue an UPDATE ODBM START(CONNECTION) DATASTORE() command. Even if the actual data store subsystems are started or recycled after the UPDATE ODBM TYPE(CONFIG) OPTION(NOCONNECT) command, ODBM will not attempt to connect to the data stores until an UPDATE ODBM START(CONNECTION) DATASTORE() command is issued.

TYPE(CONFIG)
> Specifies to update the active ODBM configuration.

Usage notes

You can issue this command only through the Operations Manager (OM) API.

The syntax for this command is defined in XML and is available to automation programs that communicate with OM.

The UPDATE ODBM TYPE(CONFIG) command allows the user to update the active ODBM configuration by loading an updated copy of the current or an alternate CSLDCxxx ODBM configuration PROCLIB member.

Before processing the UPD ODBM TYPE(CONFIG) command, all data store connections must be stopped on all ODBMs to which the command will be routed.

Output fields

The following table shows the UPDATE ODBM output fields. The columns in the table are as follows:

Short label
> Contains the short label that is generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. *error* appears for output fields that can appear for a non-zero completion code. N/A (not applicable) appears for output fields that are always returned.

Meaning
> Provides a brief description of the output field.

Table 505. Output field descriptions for the UPDATE ODBM TYPE(CONFIG) command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
CFG	N/A	ODBM configuration PROCLIB member (CSLDCxxx) used to establish the new active configuration.
MBR	N/A	IMSplex member that built the output line. The ODBM identifier of the ODBM that built the output line.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE ODBM TYPE(CONFIG) command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 506. Return and reason codes for the UPDATE ODBM TYPE(CONFIG) command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE ODBM TYPE(CONFIG) command completed successfully.
X'04000008'	X'00002004'	Invalid command keyword or invalid command keyword combination.
X'04000008'	X'00002008'	Insufficient number of keywords.
X'04000008'	X'00002014'	Invalid character in resource name.
X'04000008'	X'0000203C'	Invalid parameter specified.
X'0400000C'	X'00003000'	At least one request was successful.
X'0400000C'	X'00003004'	No requests were successful.
X'04000014'	X'00005000'	CSLDALCB allocation failed.
X'04000014'	X'0000500C'	CSLDDSCB allocation failed.
X'04000014'	X'00005010'	BPEGETM failed to acquire a parse work area.
X'04000014'	X'00005028'	BPEPARSE internal error.
X'04000014'	X'00005034'	An OM response buffer request failed.
X'04000014'	X'00005038'	A CSLDCMD0 GETBUF request failed to get a command buffer.
X'04000014'	X'00005088'	CSLDPRP allocation failed.
X'04000014'	X'0000508C'	BPERDPDS internal failure.

Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 507. Completion codes for the UPDATE ODBM TYPE(CONFIG) command

Completion code	Completion code text	Meaning
0		The UPDATE ODBM TYPE(CONFIG) command completed successfully for the resource.
1	INVALID CHARACTER, RESOURCE NAME	The resource name in the command input has invalid characters.
3	DATASTORE NOT AVAILABLE	An attempt to start a data store connection failed. The data store is unavailable.
4	DATASTORE CONNECTION FAILED	An attempt to start a data store connection failed. Message CSL4006W accompanies this completion code and explains the AIB return code, AIB reason code, and AIB error text.
5	DATASTORE NOT RRS CAPABLE	An attempt to start a data store connection failed. The data store is not RRS capable.
8	CONFIGURATION FILE NOT FOUND	The CSLDCxxx IMS PROCLIB member could not be located.
9	THIS ODBM NOT DEFINED	The current ODBM definition could not be located in the CSLDCxxx IMS PROCLIB member.
A	DUPLICATE ODBM DEFINED	Multiple ODBM definitions were found for the current ODBM.
B	DATASTORES NOT DEFINED	The current ODBM definition has no data stores defined.
C	DUPLICATE DATASTORES DEFINED	Duplicate data store names were found within the current ODBM definition.
D	DATASTORE NAME INVALID	A data store name within the current ODBM definition contains invalid characters.
E	DUPLICATE ALIAS DEFINED	A data store definition within the current ODBM definition contains duplicate alias names.
F	ALIAS NAME INVALID	An alias name defined within the current ODBM definition contains invalid characters.
10	NO RESOURCES FOUND	No resources were found.
11	BPEPARSE ERROR	BPEPARSE failed while parsing the CSLDCxxx IMS PROCLIB member. BPE0003E accompanies this completion code.
12	DATASTORES NOT STOPPED	Data store connections to the current ODBM have not been stopped before issuing the UPD ODBM TYPE(CONFIG) command.

Examples

Example 1 for UPDATE ODBM TYPE(CONFIG) command

TSO SPOC input:
```
UPD ODBM TYPE(CONFIG) MEMBER(009) OPTION(NOCONNECT)
```

TSO SPOC output:

```
MbrName      CC   ConfigName
ODBM010D      0    CSLDC009
ODBM010D      0    CSLDC009
ODBM020D      0    CSLDC009
```

Explanation: The MEMBER(009) keyword indicates that CSLDC009 will be used to configure both ODBM01 and ODBM02. To allow this to occur, both ODBM configurations must be specified in the CSLDC009 PROCLIB member. When using the NAME() keyword, routing of this command requires caution to insure the intended results are achieved.

Example 2 for UPDATE ODBM TYPE(CONFIG) command

TSO SPOC input:
```
UPD ODBM TYPE(CONFIG)
```

TSO SPOC output:
```
MbrName      CC   ConfigName
ODBM010D      0    CSLDC001
ODBM010D      0    CSLDC001
ODBM020D      0    CSLDC002
```

Explanation: This example shows that the current configuration members were, and remain to be, CSLDC001 for ODBM1 and CSLDC002 for ODBM2, respectively.

UPDATE OLREORG command

Use the UPDATE OLREORG command to change the rate, the DEL or NODEL data set disposition flag, or the REL or NOREL release OLR ownership disposition flag of an owned HALDB Online Reorganization (OLR).

If you issue this command as a type-1 command, the command response is returned as a DFS0725I pre-edit message.

Subsections:
- "Environment"
- "Syntax" on page 1245
- "Keywords" on page 1245
- "Usage notes" on page 1247
- "Command responses for /UPDATE OLREORG" on page 1247
- "Output fields" on page 1247
- "Return, reason, and completion codes" on page 1248
- "Examples" on page 1248

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the UPDATE OLREORG command and keywords can be issued.

Table 508. Valid environments for the UPDATE OLREORG command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE OLREORG	X	X	
NAME	X	X	

Command / Keywords	DB/DC	DBCTL	DCCTL
SET	X	X	
OPTION	X	X	

Syntax

```
>>--+-UPDATE---+--OLREORG--NAME--(--+-v--,---------+--)-------------------->
    +-UPD------+                    |   +-partname-+
    +-/UPDATE--+                    +-*-------------+
    +-/UPD-----+
```

```
>--+-v--SET(RATE(rate))---------------+--------------------------------><
      +-OPTION--(--+-DEL---+--)-+
                   +-NODEL-+
                   +-REL---+
                   +-NOREL-+
```

Keywords

The following keywords are valid for the UPDATE OLREORG command:

NAME()

> Specifies one or more PHDAM or PHIDAM HALDB partitions to be updated.
>
> For the type-1 version of the command, you can specify only one partition name.
>
> A parameter with the wildcard character (*) is not allowed, except as NAME(*) for all HALDB partitions.

SET(RATE)

> Specifies the rate at which the HALDB OLR is run.
>
> **rate**
>
> > You can specify a value of 1 to 100 for the rate value.
> >
> > You can use the RATE parameter to control the intensity at which the reorganization runs. This can affect both the reorganization's speed and its impact on the rest of the system. The value you specify for rate is the percentage of elapsed time to be devoted to copying records. The remaining time is to be an intentionally introduced delay in the copying process that minimizes the reorganization's impact on other IMS work and on the whole system.
> >
> > An online reorganization's impact on the system is affected by the available system resources, by total system utilization (including other online reorganizations), by total logging volume, by log contention, and by the intensity at which this reorganization was requested to run. These same factors also affect the speed at which the reorganization runs.

A rate value of 50 specifies that 50% of the elapsed time be spent copying records and the remaining 50% be spent in a delay. This causes the reorganization to run approximately twice as long as it would have run with a rate value of 100.

OPTION()

Allows the specification of the DEL, NODEL, REL, or NOREL options.

DEL | NODEL

Specifies whether the deletion of the inactive data sets is to be attempted when the online reorganization completes. DEL and NODEL are mutually exclusive keywords.

DEL

The deletion of the inactive data sets is to be attempted when the online reorganization completes. The attempted deletion occurs regardless of who created the data sets or when the data sets were created.

NODEL

The deletion of the inactive data sets is not to be attempted when the online reorganization completes.

If the partition is tracked at an RSR tracker site, the OPTION (DEL | NODEL) value in effect at the completion of the HALDB OLR also determines whether the inactive data sets for the shadow partition are deleted at the completion of the tracking of the reorganization.

If the HALDB Online Reorganization is stopped before completion, the DEL or NODEL keyword is not remembered and will need to be specified on the INITIATE OLREORG command that is issued to resume the stopped HALDB OLR.

REL | NOREL

Specifies whether the IMS system retains or releases control of the reorganization when it shuts down or terminates. REL and NOREL are mutually exclusive keywords.

REL

Specifies that the IMS system releases ownership of the OLR if it terminates before the online reorganization is completed. Specifying this keyword allows another IMS system to take control of the suspended reorganization.

The Partition Database record in the RECON data set is updated to show that the owning IMS (OLRIMSID=*ssid*) allows the OLR to be resumed if it terminates the online reorganization process or abnormally terminates itself. `RELEASE OLR OWNERSHIP` appears in a listing of the Partition Database record.

When OLR is active and running on an IMS and the IMS terminates abnormally, and if release OLR ownership is requested, the LIST.DB command will show `OLRIMSID=`*`ssid`* (instead of `OLRIMSID=NULL`) and `RELEASE OLR OWNERSHIP` because IMS was unable to request OLR ownership release.

Requesting release OLR ownership in an XRF environment: In an XRF environment, if release OLR ownership is requested while OLR is running on an active IMS, then OLR will not automatically resume on

the new active IMS after an XRF takeover occurs. You must issue the INIT OLREORG command again on the new active IMS.

NOREL
Specifies that the IMS system retains control of the reorganization when it shuts down or terminates.

Usage notes

If an online reorganization is started with the ALTER option to modify the structure of the database, IMS reorganizes all of the partitions in the database. In this case, if you need to update the online reorganizations of all of the partitions in the database, specify an asterisk in the NAME keyword, for example UPDATE OLREORG NAME(*) SET(RATE($rate$)). You cannot specify the name of the HALDB master database. You can use either the type-1 or type-2 format of the UPDATE OLREORG command to update alter processing; however, if the altered database is shared, only the type-2 command can be routed to all of the IMS systems that share the database.

Command responses for /UPDATE OLREORG

When the /UPDATE OLREORG command is entered as a type-1 command, the command response is returned in a message format.

When the command completes successfully message DFS0725I is returned to the system console and master terminal with a completion code of 0. If the command results in an error, a nonzero completion code or an error message is returned to the master terminal and system console.

```
DFS0725I INITIATE|UPDATE|TERMINATE OLREORG COMMAND FOR DB dbnamexx COMPLETE.
CC= nn
  where: dbnamexx is the HALDB partition name entered on the command
         nn       is the completion code
```

Output fields

This section describes the responses from the OM API for the UPDATE OLREORG command. The following table shows the UPDATE OLREORG output fields. The columns in the table are as follows:

Short label
Contains the short label that is generated in the XML output.

Keyword
Identifies the command keyword that caused the field to be generated.

N/A appears for output fields that are always returned.

Meaning
Provides a brief description of the output field.

Table 509. Output fields for UPDATE OLREORG command

Short label	Keyword	Meaning
PART	N/A	Partition name.
MBR	N/A	The IMS that built the command response line.
CC	N/A	Completion code.

Return, reason, and completion codes

The OM return and reason codes that might be returned as a result of the UPDATE OLREORG command are standard for all commands that are entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 510. Return and reason codes for the UPDATE OLREORG command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE OLREORG command completed successfully.
X'00000004'	X'00001010'	No matches found for filter.
X'00000008'	X'00002008'	Insufficient number of keywords specified.
X'00000008'	X'00002040'	No filter, an invalid filter, or an insufficient number of filters was specified.
X'0000000C'	X'00003000'	At least one request was successful.
X'0000000C'	X'00003004'	None of the requests was successful.
X'00000010'	X'0000400C'	Command issued on an XRF alternate.
X'00000010'	X'00004014'	Command issued on an RSR tracker.
X'00000014'	X'00005000'	A GETMAIN error occurred.

The following table includes an explanation of the completion codes. Errors unique to the processing of UPDATE OLREORG command are returned as completion codes. A completion code is returned for each action attempted on a HALDB partition.

Table 511. Completion codes for the UPDATE OLREORG command

Completion code	Meaning
0	The UPDATE OLREORG command completed successfully for the partition.
10	Resource name is invalid.
14	Resource is not a partition name.
1C	Resource is a partitioned secondary index.
24	No HALDB OLR is in progress.
CF	Parameter value conflict, or invalid parameter value.
1E0	DBRC OLRREL	OLRNOREL failed.

Examples

The following are examples of the UPDATE OLREORG command:

Example 1 for /UPDATE OLREORG

Entry ET:
```
/UPD OLREORG NAME(PDHDOKA) SET(RATE(25))
```

Response ET:

```
DFS0725I UPDATE   OLREORG COMMAND FOR DB PDHDOKA  COMPLETE. CC=   0
```

Explanation: The UPDATE OLREORG command is issued for partition PDHDOKA to change the OLR rate to 25. The command is successful as indicated in the message DSF0725 command response.

Example 2 for UPDATE OLREORG

TSO SPOC input:

```
UPD OLREORG NAME(PDHDOKA,PDHDOKB) SET(RATE(25))
```

TSO SPOC output:

```
Partition MbrName       CC
    PDHDOKA   IMSA        0
    PDHDOKA   IMS1       24
    PDHDOKB   IMSA        0
    PDHDOKB   IMS1       24
```

OM API input:

```
CMD (UPD OLREORG NAME(PDHDOKA,PDHDOKB) SET(RATE(25)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.2.0</omvsn>
<xmlvsn>1    </xmlvsn>
<statime>2003.168 21:17:57.712194</statime>
<stotime>2003.168 21:17:57.713062</stotime>
<staseq>B996297E02942007</staseq>
<stoseq>B996297E02CA6487</stoseq>
<rqsttkn1>USRT005 10141757</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003000</rsn>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC     </styp>
<rc>0000000C</rc>
<rsn>00003004</rsn>
<rsntext>No requests were successful</rsntext>
</mbr>
</cmderr>
<cmd>
<master>IMSA    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>OLREORG        </kwd>
<input>UPD OLREORG NAME(PDHDOKA,PDHDOKB) SET(RATE(25)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PART" llbl="Partition" scope="LCL" sort="A" key="1"
 scroll="NO" len="9" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="N" key="0" scroll="NO"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="N" key="0" scroll="YES"
 len="4" dtype="INT" align="right" />
</cmdrsphdr>
<cmdrspdata>
```

```
<rsp> PART(PDHDOKA ) MBR(IMSA     ) CC(    0) </rsp>
<rsp> PART(PDHDOKB ) MBR(IMSA     ) CC(    0) </rsp>
<rsp> PART(PDHDOKA ) MBR(IMS1     ) CC(   24) </rsp>
<rsp> PART(PDHDOKB ) MBR(IMS1     ) CC(   24) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE OLREORG command is issued for partitions
PDHDOKA and PDHDOKB to update the OLR rate to 25. The command is
successful at IMSA, where OLR is in progress and is not successful at IMS1, where
no OLR is in progress. A completion code of 24 is returned in the IMS1 response.

Related concepts:

➦ How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

➦ Command keywords and their synonyms (Commands)

UPDATE OTMADESC command

Use the UPDATE OTMADESC command to update an existing OTMA destination
descriptor without restarting IMS.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1252
- "Usage notes" on page 1255
- "Output fields" on page 1256
- "Return, reason, and completion codes" on page 1256
- "Examples" on page 1258

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 512. Valid environments for the UPDATE OTMADESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE OTMADESC	X		X
NAME	X		X
SET	X		X

Syntax

>>─┬─────────────────────┬─><
   └─EXIT(─┬─N─┬─)────────┘
           └─Y─┘

A (Descriptor TYPE=IMSCON):

>├─┬──┬─>
   └─ADAPTER(*adaptername*)─CONVRTR(*convertername*)─┘

>─┬──┬─┬────────────────────────┬─>
   └─RMTIMSCON(*rmtimsconnect*)─RMTIMS(*remote_IMS*)─┘ └─RMTTRAN(*trancode*)─┘

>─SMEM(─┬─N─┬─)─SYNTIMER(*syntimer*)─TMEMBER(*tmem*)─TPIPE(*tpipename*)─────────>
        └─Y─┘

>─┬──────────────────────┬─┤
   └─USERID(*user_id*)─┘

B (Descriptor TYPE=IMSTRAN):

>├─┬─────────────────┬─┬──────────────────┬─┬────────────────────────┬─>
   └─TMEMBER(*tmem*)─┘ └─TPIPE(*tpipename*)─┘ └─LTERMOVR(*ltermname*)─┘

>─┬─────────────────────────┬─┬─────────────────┬─┬──────────────────────────┬─┤
   └─REPLYCHK(─┬─Y─┬─)─┘   └─SMEM(─┬─N─┬─)─┘   └─SYNTIMER(*timeout value*)─┘
              └─N─┘   └─Y─┘
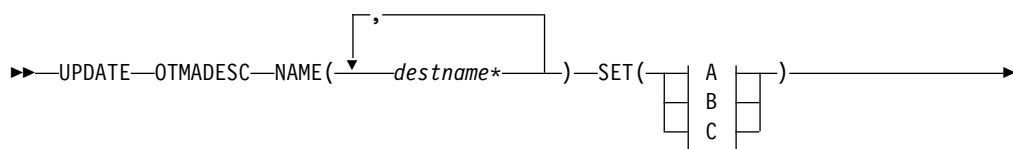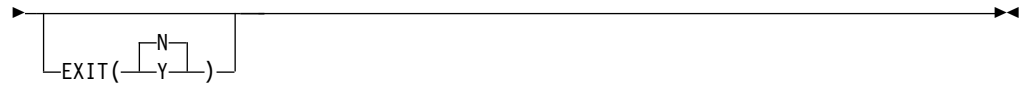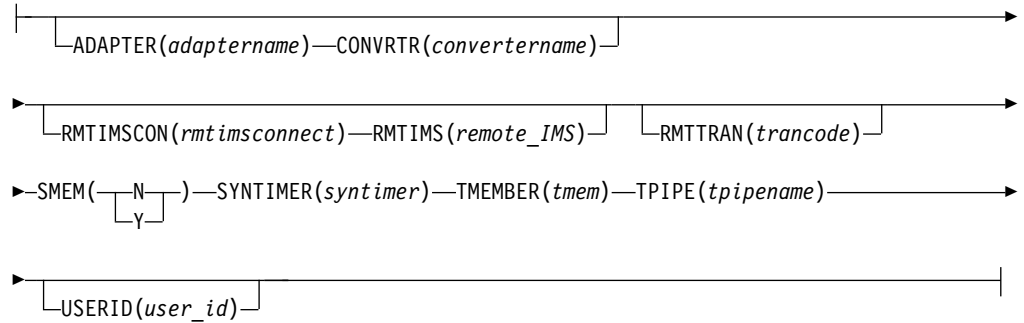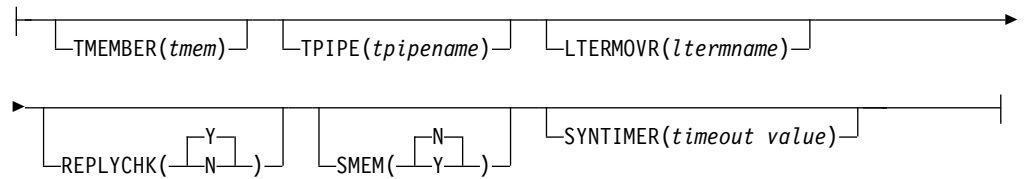
C (Descriptor TYPE=MQSERIES):

>├──TMEMBER(*tmem*)─┬──────────────────────┬─┬───────────────────────────┬─>
                     └─TPIPE(*tpipename*)─┘   └─MQREPORT(─┬─COPYMTOC─┬─)─┘
                                                          ├─NEWMSGID─┤
                                                          ├─PASSMSG1─┤
                                                          ├─PASSCORR─┤
                                                          └─NONE─────┘

>─┬──────────────────┬─┬────────────────────────┬─>
   └─USERID(*userid*)─┘ └─MQFORMAT(─┬─MQIMSVS─┬─)─┘
                                    ├─MQIMS───┤
                                    ├─NONE────┤
                                    └─MQSTR───┘

>─┬──────────────────────────────────────┬─┬───────────────────────────┬─>
   └─MQRTF(*mqmd_replytoformatname*)─┘   └─MQMSGID(*mqmd_msgid*)─┘

>─┬─────────────────────────────────┬──MQRTQ(*mqmd_replytoqueuename*)──────────>
   └─MQCORREL(*mqmd_correlid*)─┘

>─┬──┬─┬──────────────────────────────────┬─┤
   └─MQRTQMGR(*mqmd_replytoqueuemgrname*)─┘   └─MQAPPLID(*mqmd_appidentitydata*)─┘

```
►─┬─────────────────┬──┬──────────────┬──┬────────────────┬──────────────┤
  │          ┌─Y─┐  │  │       ┌─N─┐  │  │         ┌─N─┐  │
  └─MQCOPYMD(─┴─N─┴─)─┘  └─SYNCTP(─┴─Y─┴─)─┘  └─MQPERST(─┴─Y─┴─)─┘
```
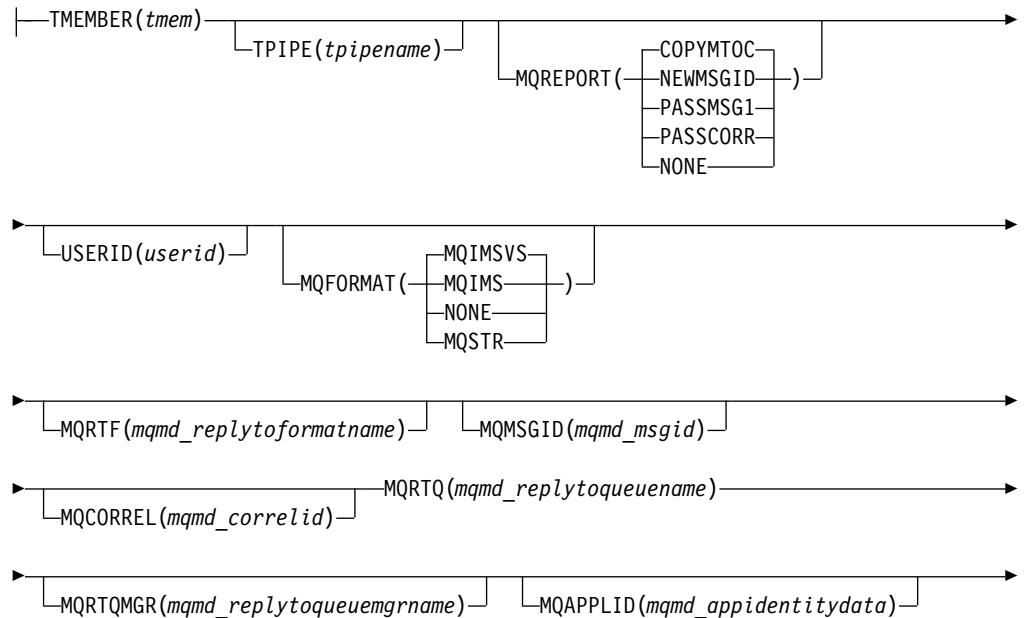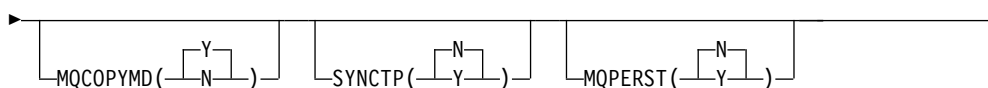
Keywords

The following keywords are valid for the UPDATE OTMADESC command:

NAME(*destname**)
   A required keyword whose value cannot be modified.

SET()
   Specifies to update the descriptors with at least one of the following keywords. The update applies to each of the descriptor names specified in the NAME keyword. If an asterisk is appended to the descriptor name, the update will not be applied to the group of names it is trying to mask.

   ADAPTER(*adaptername*)
      An optional parameter for TYPE(IMSCON) that identifies the IMS Connect adapter. *adaptername* is a 1- to 8-character adapter name. To clear the value of this keyword, specify the keyword with no value, for example, ADAPTER(). If the ADAPTER keyword is deleted, the CONVRTR keyword is also deleted. ADAPTER and TYPE(NONOTMA) are mutually exclusive.

   CONVRTR(*convertername*)
      A required parameter only if the ADAPTER keyword is specified. *convertername* is a 1- to 8-character converter name used by the adapter. The value of this keyword will be automatically deleted if the value of the ADAPTER keyword is being deleted. Coding this keyword without the ADAPTER keyword or with TYPE(NONOTMA) is not valid.

   EXIT(N|Y)
      When TYPE=IMSTRAN, this is an optional parameter used to indicate whether the IMS user exit (DFSCMUX0) can override the descriptor routing information for late messages.

      For other descriptor types, this is an optional parameter used to specify whether the OTMA routing exits (DFSYPRX0 and DFYDRU0) can override the descriptor switch destination from ALT IOPCB.

   LTERMOVR(*ltermname*)
      When TYPE=IMSTRAN, this is an optional parameter used to specify a logical terminal name that overrides the name specified in the application I/O PCB. If an override name is also specified in the AIBRSNM2 field of the AIB, that name is used instead of the name in the descriptor. If an override name is not specified in the descriptor or the AIB, the IMS application terminal symbolic (PSTSYMBO) is used as the default logical terminal name for the target transaction.

   MQAPPLID(*mqmd_appidentitydata*)
      When TYPE=MQSERIES, this is an optional parameter that specifies a 1- to 32-character value for the MQMD_APPLIDENTITYDATA field. This field is used to pass data to the receiving IBM MQ application. The default is 32 blanks.

   MQCOPYMD(Y | N)
      When TYPE=MQSERIES, this is an optional parameter that specifies whether the original input values of the MQMD data structure from the MQ IMS Bridge are copied to the output message.

MQCORREL(*mqmd_correlid*)
When TYPE=MQSERIES, this is an optional parameter that specifies a 1- to 24-character value for the MQMD_CORRELID field. This value is passed directly to the receiving IBM MQ application if the value of the MQREPORT parameter is PASSCORR. The default is 0.

MQFORMAT(MQIMSVS | MQIMS | NONE | MQSTR)
When TYPE=MQSERIES, this is an optional parameter that specifies value for the MQMD_FORMAT field. The default value is MQIMSVS.

MQMSGID(*mqmd_msgid*)
When TYPE=MQSERIES, this is an optional parameter that specifies a 1- to 24-character value for the MQMD_MSGID field. This value is passed directly to the receiving IBM MQ application if the value of the MQREPORT parameter is PASSMSG1.

MQPERST(N | Y)
When TYPE=MQSERIES, this is an optional parameter that specifies whether messages that use this descriptor are persistent on a SYNC TPIPE. Setting this parameter to Y is analogous to setting X'40' for the output flag value of the DFSYDRU0 exit routine.

MQREPORT(COPYMTOC | NEWMSGID | PASSMSG1 | PASSCORR | NONE)
When TYPE=MQSERIES, this is an optional parameter that specifies the MQMD_REPORT field value for messages that use the descriptor. The default value is COPYMTOC. Each supported value corresponds to an equivalent value in the MQMD data structure:

Parameter value
> **Equivalent MQMD structure value**

NEWMSGID
> MQRO_NEW_MSG_ID

PASSMSGI
> MQRO_PASS_MSG_ID

COPYMTOC
> MQRO_COPY_MSG_ID_TO_CORREL_ID

PASSCORR
> MQRO_PASS_CORREL_ID

PASSMSCO
> MQRO_PASS_MSG_ID and MQRO_PASS_CORREL_ID

NONE
> MQRO_NONE

MQRTF(*mqmd_replytoformat*)
When TYPE=MQSERIES, this is a 1- to 8-character value for the MQMD_REPLYTOFORMAT field. The default is 0.

MQRTQ(*mqmd_replytoqueuename*)
When TYPE=MQSERIES, this is a required parameter that specifies a 1- to 48-character value for the MQMD_REPLYTOQ field.

MQRTQMGR(*mqmd_replytoqueuemanagername*)
When TYPE=MQSERIES, this is an optional parameter that specifies a 1- to 48-character value for the MQMD_REPLYTOQMGR field. If this parameter is not specified, MQSeries® will use the local queue manager.

REPLYCHK(Y|N)
Specifies whether an IMS application replies to the IOPCB for a synchronous program switch request. When REPLYCHK=YES and the ICAL target application does not reply to the IOPCB or switch to another transaction, IMS gives the error return code X'0100', with reason code X'0110', and extended reason code X'0014' instead of a timeout to the ICAL call. The default is Y.

RMTIMS
An optional 1- to 8-character name of a remote, destination IMS system for ALTPCB output messages. The RMTIMS value must match the value specified on the ID parameter of a DATASTORE statement in the configuration member of the remote IMS Connect instance specified on the RMTIMSCON parameter. This parameter is valid only when TYPE(IMSCON) is specified. To clear the value of this parameter, specify the parameter with no value, for example, RMTIMS(). Clearing the value of RMTIMS also clears the value of RMTIMSCON.

RMTIMSCON
An optional 1- to 8-character name of a connection to the remote IMS Connect instance that manages TCP/IP communications for the remote IMS system that is named on the RMTIMS parameter. This value must match the value specified on the ID parameter of the RMTIMSCON statement in the configuration file of the local IMS Connect instance that is managing TCP/IP communications for the IMS system in which the OTMA ALTPCB output messages are originating. This parameter is valid only when TYPE(IMSCON) is specified. To clear the value of this parameter, specify the parameter with no value, for example, RMTIMSCON(). Clearing the value of RMTIMSCON also clears the value of RMTIMS.

RMTTRAN
An optional 1- to 8-character name of the transaction name to use at the remote, destination IMS system named on the RMTIMS parameter. When this parameter is specified with RMTIMSCON and RMTIMS parameters, IMS OTMA sends the ALTPCB output messages to the remote IMS system for transaction processing. This parameter is valid only when TYPE(IMSCON) is specified. To clear the value of this parameter, specify the parameter with no value, for example, RMTTRAN().

SMEM(N | Y)
An optional parameter that can either be a Y or N value to indicate whether the TMEMBER name specified in the TMEMBER parameter is a super member. To clear the value of this keyword, specify the keyword with no value, for example, SMEM(). Deleting SMEM() will default to SMEM(N). If the TMEMBER name is a super member, the length of the TMEMBER name has a maximum of 4 characters. SMEM and TYPE(NONOTMA) are mutually exclusive.

SYNCTP(N | Y
When TYPE=MQSERIES, this parameter specifies whether a SYNC TPIPE must be created for messages that use this descriptor. Setting this parameter to Y is analogous to setting X'80' for the output flag value of the DFSYDRU0 exit routine.

SYNTIMER(*syntimer*)
An optional parameter used for synchronous callout processing. In synchronous callout processing, a request is canceled if an ACK/NAK or response is not received by OTMA from the client within the time

specified. This value is expressed in hundredths of a second or milliseconds. The value can be numeric with a maximum length of six digits. The value has a range of 0 through 999999 and when zero is specified, it will default to 10 seconds or 1000 hundredths of a second. If you specify SYNTIMER() or SYNTIMER( ), the timeout value will be reset as if no timeout value has been specified.

If TYPE(IMSTRAN) is specified, this parameter is the timeout value for synchronous program switch ICAL calls. If a timeout value is also specified in the AIB interface, IMS will compare the timeout values and use the lower value.

TMEMBER(*tmem***)**
A required parameter for TYPE(IMSCON). If the TYPE of the descriptor is being changed from IMSCON to NONOTMA, the value of this keyword will be deleted. If the TYPE of the descriptor is being changed from NONOTMA to IMSCON, then the TMEMBER keyword must be coded. *tmem* is a 16-character OTMA TMEMBER name or a 4-character super member. TMEMBER and TYPE(NONOTMA) are mutually exclusive.

An optional parameter for TYPE(IMSTRAN). The *tmem* value is a 1- to 16-character OTMA TMEMBER name. IMS queues late responses from a synchronous program switch to this OTMA TMEMBER. The destination name is used as the default OTMA TPIPE name if the TPIPE parameter is not specified in the descriptor.

TPIPE(*tpipename***)**
An optional parameter that is a 1- to 8-character TPIPE name when TYPE(IMSCON) is specified. To delete the value of this keyword, specify the keyword with no value, for example, TPIPE(). If this keyword is not specified, the TPIPE name is the destination name specified in the NAME keyword. TPIPE and TYPE(NONOTMA) are mutually exclusive.

An optional parameter for TYPE(IMSTRAN) that specifies a 1- to 8-character TPIPE name. This parameter defaults to the destination name. IMS queues late responses from a synchronous program switch to this OTMA TPIPE. A value for the TMEMBER parameter is required when TPIPE is specified.

USERID
For TYPE=IMSCON, this is an optional 1- to 8-character RACF user ID. When this parameter is specified with RMTIMSCON, RMTTRAN, and RMTIMS parameters, a remote, destination IMS system uses the USERID value to perform transaction authorization. The value of USERID specified in the OTMA destination descriptor overrides the user ID provided by the IMS application program that issued the ISRT call to the OTMA ALTPCB. This parameter is valid only when TYPE(IMSCON) is specified.

For TYPE=MQSERIES, this is a 1- to 8-character user ID that corresponds with the MQMD_USERIDENTIFIER field in the MQMD data structure. This field allows the OTMA descriptor to override the default user ID in the MQMD structure. Specify the value *USERID* to use the IMS user ID value from the PST.

Usage notes

The UPDATE OTMADESC command is used to modify existing destination routing descriptors that were either included in DFSYDTx or added by using the CREATE OTMADESC command. A log record will be written to track changes and

to persist from one IMS restart to another. The log record will also be used to track the currency of updates to an XRF alternate and to an RSR tracking environment. The checkpoint record is X4035 and the log record for the CREATE, UPDATE, and DELETE commands is X221B.

The UPDATE OTMADESC command cannot change the TYPE value for a destination descriptor. If you want to change the type for a descriptor, you must create a new descriptor with the CREATE OTMADESC command or by modifying the DFSYDTx member of the IMS.PROCLIB data set. You can then delete the existing descriptor.

Output fields

The following table shows the UPDATE OTMADESC output fields. The columns in the table are:

Short label
Contains the short label generated in the XML output.

Long label
Contains the column heading for the output field in the formatted output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 513. Output fields for the UPDATE OTMADESC command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code for the line of output. The completion code indicates whether IMS was able to process the command for the specified resource. See "Return, reason, and completion codes" for more information. The completion code is always returned.
CCTXT	CCText	N/A	Completion code text that briefly explains the meaning of the non-zero completion code. This field is returned only for an error completion code.
DEST	DestName	NAME	Destination name.
MBR	MbrName	N/A	Member name.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the UPDATE OTMADESC command. The OM return and reason codes that might be returned as a result of the UPDATE OTMADESC command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 514. Return and reason codes for the UPDATE OTMADESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. See Table 515 for details.
X'02000008'	X'00002000'	The command contains an invalid verb or no client is registered for the verb.
X'02000008'	X'00002004'	The command contains an invalid primary keyword or no client is registered for the keyword.
X'02000008'	X'00002028'	The command contains an invalid keyword.
X'02000008'	X'0000202C'	The command contains an unknown positional parameter.
X'02000008'	X'00002034'	The command contains an incomplete keyword parameter.
X'02000008'	X'00002038'	The command is missing a required parameter.
X'02000008'	X'0000203C'	The command contains an invalid keyword parameter value.

The following table includes an explanation of the completion codes. Errors unique to the processing of this command are returned as completion codes. A completion code is returned for each action against an individual resource.

Table 515. Completion codes for the UPDATE OTMADESC command

Completion code	Completion code text	Meaning
0	Command completed successfully	The UPDATE OTMADESC command completed successfully for the resource.
151	Descriptor not found	The descriptor name does not exist.
153	Adapter blank, Convrtr not blank	The value of the ADAPTER keyword is blank and the value of the CONVRTR keyword is not blank. Either both keywords must be blank or both keywords must contain a valid name.
154	Adapter not blank, Convtr blank	The value of the ADAPTER keyword is not blank and the value of the CONVRTR keyword is not blank. Either both keywords must be blank or both keywords must contain a valid name.
156	SMEM=Y, super mbr name GT 4 char	The super member name in the TMEMBER keyword must have a maximum length of 4 characters.
159	TMEMBER is required for IMSCON	The type is being changed from NONOTMA to IMSCON, but the value of TMEMBER is blank.
162	Descriptor not available for upd	During update processing, the descriptor was deleted by another user.

Table 515. Completion codes for the UPDATE OTMADESC command (continued)

Completion code	Completion code text	Meaning
167	SYNTIMER must have numeric value	The timeout value must have a value expressed in numbers within parenthesis.
169	SYNTIMER has nonnumeric value	The value must not contain alphabetic characters or any character that is not numeric.

Examples

The following are examples of the UPDATE OTMADESC command:

Example 1 for UPDATE OTMADESC command

TSO SPOC input:
```
UPDATE OTMADESC NAME(OTMACL*,OTMACL99) SET(TYPE(NONOTMA))
```

TSO SPOC output:
```
DestName    MbrName    CC
OTMACL99    IMSA        0
OTMACL*     IMSA        0
```

Explanation: This UPDATE command changes the output type to NONOTMA. This keyword will delete any other keywords (such as TMEM or TPIPE), because no other keywords are relevant for TYPE(NONOTMA).

Example 2 for UPDATE OTMADESC command

TSO SPOC input:
```
UPDATE OTMADESC NAME(OTMACL*,OTMACL99) SET(TPIPE(HWS1TP02))
```

TSO SPOC output:
```
DestName    MbrName    CC
OTMACL99    IMSA        0
OTMACL*     IMSA        0
```

Explanation: This UPDATE command changes the TPIPE name to HWS1TP02. The TPIPE name will be changed from HWS1TP01 to HWS1TP02 for OTMACL99 and the TPIPE for OTMACL* will be set to HWS1TP02.

Example 3 for UPDATE OTMADESC command

TSO SPOC input:
```
UPDATE OTMADESC NAME(OTMACL*) SET(TPIPE())
```

TSO SPOC output:
```
DestName   MbrName     CC
OTMACL*    IMSA         0
```

Explanation: This UPDATE command for descriptor OTMACL* will delete the TPIPE name by coding TPIPE(). An asterisk in the UPDATE command does not

update the group of names the asterisk is masking. It updates only the OTMACL*
entry in the table of destination routing descriptors. It does not update
OTMACL99.

Example 4 for UPDATE OTMADESC command

TSO SPOC input:

```
UPDATE OTMADESC NAME(OTMACL01) SET(RMTIMS(IMS3) RMTTRAN(TRAN03))
```

TSO SPOC output:

```
DestName  MbrName  CC
OTMACL01  IMS1      0
```

Explanation: When the UPDATE OTMADESC command is issued to IMS1, the
command changes the name of the destination remote IMS system to IMS3 and the
transaction to be scheduled for processing messages on IMS3 to TRAN03.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

UPDATE PGM command

Use the UPDATE PGM command to update program resources.

A program resource defines the resource requirements for application programs
that run under the control of the DB/TM environment, as well as for application
programs that access databases through DBCTL. Program resources combined with
transactions define the scheduling and resource requirements for an application
program.

Subsections:
- "Environment"
- "Syntax" on page 1260
- "Keywords" on page 1260
- "Usage notes" on page 1267
- "Equivalent IMS type-1 commands" on page 1268
- "Output fields" on page 1268
- "Return, reason, and completion codes" on page 1269
- "Examples" on page 1273

Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which
you can use the commands and keywords.

Table 516. Valid environments for the UPDATE PGM command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE PGM	X	X	X
NAME	X	X	X

Table 516. Valid environments for the UPDATE PGM command and keywords (continued)

Command / Keywords	DB/DC	DBCTL	DCCTL
OPTION	X	X	X
SET	X	X	X
START	X	X	X
STOP	X	X	X

Syntax

```
>>--+-UPDATE-+--PGM--NAME(--+-,--+--)----------------------------->
    +-UPD----+                 name


>----+-START(--+-,--+----------)---------------------------------->< 
     |         +-SCHD----+                  +-OPTION(ALLRSP)-+
     |         +-TRACE---+
     |         +-REFRESH-+
     |
     +-STOP(--+-,--+---)-----------------------------------
     |        +-SCHD----+
     |        +-TRACE---+
     |
     +-START(--+-,--+---)--STOP(--+-,--+---)-----------------------
     |         +-SCHD----+         +-SCHD----+
     |         +-TRACE---+         +-TRACE---+
     |
     +-SET(--+-,--+------------------------)
             +-BMPTYPE(--+-Y-+--)----------+
             |           +-N-+
             +-DOPT(--+-N-+--)-------------+
             |        +-Y-+
             +-FP(--+-E-+--)---------------+
             |      +-N-+
             +-GPSB(--+-N-+--)-------------+
             |        +-Y-+
             +-LANG(--+-ASSEM--+--)--------+
             |        +-COBOL--+
             |        +-JAVA---+
             |        +-PASCAL-+
             |        +-PLI----+
             +-LOCK(--+-ON--+--)-----------+
             |        +-OFF-+
             +-RESIDENT(--+-N-+--)---------+
             |            +-Y-+
             +-SCHDTYPE(--+-PARALLEL-+--)--+
             |            +-SERIAL---+
             +-TRANSTAT(--+-N-+--)---------+
                          +-Y-+
```
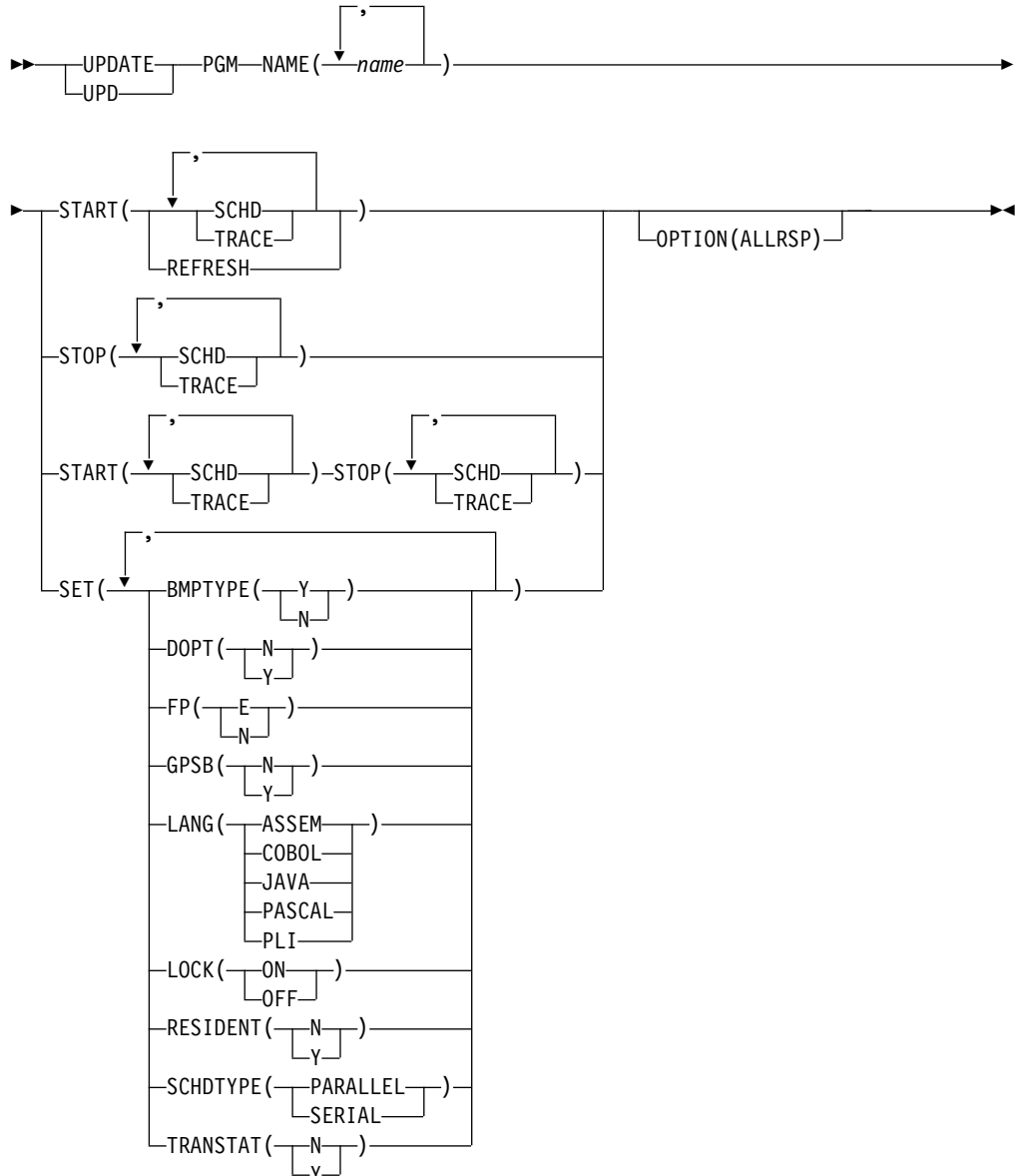
Keywords

The following keywords are valid for the UPDATE PGM command:

NAME

Specifies the 1-8 character name of the program. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the resource names that are processed. For NAME(*), command responses are returned only for the resource names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the resource names that are processed.

OPTION

Specifies functions to be performed along with the command.

> **ALLRSP**
>
> Indicates that the response lines are to be returned for all resources that are processed on the command. The default action is to return response lines only for the resources that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SET

Specifies the attribute values to be changed.

> **BMPTYPE**
>
> BMP type option. Specifies whether the program runs in a BMP type region (Y) or not (N). A BMP type region can be a BMP region or a JBP region. Updating this attribute stops the program from scheduling during command processing.
>
> PSBs scheduled by DB2 stored procedures, by programs running under WebSphere Application Server, and by other users of the ODBA interface might be defined with BMPTYPE Y or N.
>
> **Y** The program runs in a BMP type region. It might run in an IMS BMP region or a JBP region. Any associated transactions are assigned normal and limit priority values of zero.
>
> **N** The program does not run in a BMP type region. It might run in an IMS TM MPP, JMP, or IFP region or it might use either the ODBA or DRA interface. This specification should be used for programs running in IMS TM MPP, JMP, and IFP regions, or PSBs scheduled by CICS programs using DBCTL and other users of the DRA interface. This is the default.
>
> Keyword combination rules include the following:
> - FP(E) and BMPTYPE(Y) are mutually exclusive.

> **DOPT**
>
> Specifies the dynamic option.
>
> **N** The PSB associated with this program is not located dynamically. The control blocks for the PSB must exist in an ACBLIB or the IMS catalog, otherwise the program is set to a NOTINIT status, and cannot be scheduled.
>
> **Y** The PSB associated with this program is located dynamically. Each time the program that is associated with this PSB is scheduled, the latest copy of the PSB is loaded. When the program terminates, the PSB is deleted from the PSB pool.
>
> Until the PSB is required to process a transaction, the ACB for the PSB does not need to be in the ACBLIB or, if the IMS management of ACBs is enabled, in the IMS catalog. A new version of the PSB can be

defined and added to the ACBLIB or the IMS catalog at any time. IMS loads the new version of the ACB the next time the PSB is scheduled.

A DOPT PSB cannot be scheduled if the control blocks for the DBDs that the PSB references are not in the ACBLIB or the IMS catalog.

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:
- DOPT(Y) and GPSB(Y) are mutually exclusive.
- DOPT(Y) and LANG(JAVA) is a valid combination.
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- RESIDENT(Y) and DOPT(Y) are mutually exclusive.
- SCHDTYPE(PARALLEL) and DOPT(Y) are mutually exclusive.

FP Specifies the Fast Path option.

E The program is a Fast Path-exclusive application program. This implicitly defines a wait-for-input (WFI) application program. Either a transaction or a routing code that can be assigned by the user Input Edit/Routing exit routine must be defined for the Fast Path-exclusive application, in order for this program to be usable.

N The program is not a Fast Path-exclusive application program. When FP(N) is specified, any attempt to use Fast Path resources or commands might yield unpredictable results.

Updating this attribute is rejected if any routing codes or transactions that reference this program have conflicting attributes. To update this attribute, you might need to delete the routing codes and transactions that reference it. Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:
- FP(E) requires Fast Path to be defined.
- LANG(JAVA) and FP(E) are mutually exclusive.
- BMPTYPE(Y) and FP(E) are mutually exclusive.

GPSB
Specifies the generated PSB option.

N The PSB associated with the program is not generated by IMS. The control blocks for the PSB must exist in the ACBLIB or, if the IMS management of ACBs is enabled, in the IMS catalog, otherwise the program is set to a NOTINIT status, and cannot be scheduled.

Y The PSB and application control block (ACB) that are associated with the program are generated by IMS. The scheduling process of all environments generates a PSB that contains an I/O PCB and an alternate modifiable PCB.

You do not need to generate the PSB and ACB or submit DDL to define the PSB. The ACB for the PSB is not loaded from either ACBLIB or, if the IMS management of ACBs is enabled, the IMS catalog. Thus, I/O to retrieve the ACB from either the ACBLIB or the IMS catalog is eliminated.

The generated PSB contains an I/O PCB named IOPCBbbb and a modifiable, alternate PCB named TPPCB1bb. With an alternate modifiable PCB, an application can use the CHNG call to change the output destination and send output to a destination other than the input destination.

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:
- DOPT(Y) and GPSB(Y) are mutually exclusive.
- DOPT(Y) and LANG(JAVA) is a valid combination.
- GPSB(Y) requires LANG.
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- RESIDENT(Y) and GPSB(Y) are mutually exclusive.

LANG

Specifies the language interface of the program for a GPSB, or defines a DOPT(Y) program as using the Java language.

In order to define a DOPT program using the Java language, the program must be defined with DOPT(Y) and LANG(JAVA). DOPT PSBs are not loaded at IMS restart; they are loaded every time the program is scheduled. When the program is scheduled for the first time, IMS does not know the language until after the program is scheduled in a region and the PSB is loaded. Unless LANG(JAVA) is defined for the DOPT(Y) program, the program is incorrectly scheduled in a non-Java region.

The LANG parameters and their meanings are identified in the following table.

LANG parameter	Meaning
ASSEM	Assembly language
COBOL	COBOL
JAVA	Java
PASCAL	Pascal
PLI	PL/I

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- DOPT(Y) and LANG(JAVA) is a valid combination.
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- FP(E) and LANG(JAVA) are mutually exclusive.

LOCK

Specifies that the LOCK status is to be set on or off. SET(LOCK(ON | OFF)) cannot be specified with any other SET attribute. SET(LOCK(ON | OFF)) can be specified with the START or STOP keyword.

ON Locks the program and prevents it from being scheduled.

OFF

Unlocks the program and allows it to be scheduled.

RESIDENT

Specifies the resident option. The RESIDENT(N) option takes effect immediately. The RESIDENT(Y) option takes effect at the next restart, unless an error is encountered such as no ACB for the PSB for the program, or if the program was updated as RESIDENT(Y) after the checkpoint from which this IMS is performing emergency restart.

N The PSB associated with the program is not resident in storage. The PSB is loaded at scheduling time.

Y The PSB associated with the program is made resident in storage at IMS cold start or restart. IMS loads the PSB and initializes it.

A resident PSB is accessed in local storage, which avoids I/O to retrieve the ACB.

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:
- DOPT(Y) and RESIDENT(Y) are mutually exclusive.
- GPSB(Y) and RESIDENT(Y) are mutually exclusive.

SCHDTYPE

Specifies whether this application program can be scheduled into more than one message region or batch message region simultaneously.

PARALLEL

The application program can be scheduled in multiple regions simultaneously.

SERIAL

The application program can be scheduled in only one region at a time.

Updating this attribute to SCHDTYPE(SERIAL) is rejected if a transaction that references this program is defined with a parallel limit count other than 65535.

Updating this attribute stops the program from scheduling for the duration of command processing.

Keyword combination rules include the following:
- DOPT(Y) and SCHDTYPE(PARALLEL) are mutually exclusive.

TRANSTAT

Specifies whether transaction level statistics should be logged. The value specified has meaning only if the program is a JBP or a non-message driven BMP. If Y is specified, transaction level statistics are written to the log in an X'56FA' log record.

N Transaction level statistics should not be logged.

Y Transaction level statistics should be logged.

The TRANSTAT keyword on the UPDATE PGM or UPDATE PGMDESC command gives the user the ability to override the system default or the current value of the TRANSTAT parameter. If the TRANSTAT keyword is

omitted on the UPDATE PGM command, the current transaction level statistics setting is unchanged for the program.

START
Specifies attributes that are to be started.

> **SCHD**
> Starts scheduling of the application program. Or, resets the NOTINIT status if set because of an invalid database or DMB, in case the database or DMB has been created since the scheduling failure. If the database or DMB is still invalid when the program is next scheduled, the program will be marked with a status of NOTINIT again.

> **TRACE**
> Starts the tracing of the DL/I portion of Data Communications (DC) for the application program. Each DL/I call to a TPPCB, issued by the user application program, is traced on entry to and exit from the DC call handler DFSDLA30. On entry to DFSDLA30 a type 6701-LA3A record is written, on exit from DFSDLA30 a type 6701-LA3B record is written.
>
> Each record contains the following items if applicable:
> - TPPCB
> - Up to 64 bytes of the I/O area
> - SMB
> - PST
>
> If the batch message program (BMP) being traced is the IBM IMS Queue Control Facility for z/OS (5697-E99), a 6701-MRQB record is logged by the IMS Queue Control Facility module DFSQMRQ0. The default program name for the IMS Queue Control Facility BMP is MRQPSB, and can be overridden on the MSGQUEUE system definition macro.
>
> Items logged in the 6701-MRQB record, if applicable, are:
> - TPPCB
> - AIB
> - I/O area
> - PST
> - QTPDST
> - QSAPWKAD
> - QMBA
> - PSTDCA
> - REG14-12

> **REFRESH**
> Specifies that the regions the program is scheduled in are posted so that a new refreshed copy of the program can be obtained on the next schedule of the program.
>
> The UPDATE PGM START(REFRESH) command is supported for programs scheduled in the following region types:
> - MPP pseudo-wait-for-input (PWFI) regions in which the program is scheduled and the program is not preloaded by the DFSMPLxx PROCLIB member
> - JMP PWFI regions in which the specified program name is scheduled

- MPP, JMP, and message-driven BMP regions in which the program is scheduled and that are running a transaction defined as WFI=YES

The UPDATE PGM START(REFRESH) command is not supported for the following region types:

- MPP regions where the program is loaded by the DFSMPLxx PROCLIB member
- IFP regions
- JBP regions
- Non-message-driven BMP regions

The UPDATE PGM START(REFRESH) command is not valid in a DBCTL environment or on an XRF alternate.

During UPDATE PGM START(REFRESH) command processing, if the region is waiting for a message on SUBQ6 in `WAIT-MESSAGE` or `WAIT-INPUT` status, the region is posted and the program receives a `QC` status code. If the region is active and is processing a message, the program receives a `QC` status code after the current work is processed and before the next GU call is made. If there is new work on the queue for the program, the next GU call will obtain the refreshed program if the program was refreshed in the PGMLIB before the command was issued.

For MPP regions that are coded with the DBLDL=*xx* parameter, if the program name specified is in the dependent region BLDL list, the program is removed from the BLDL list when the region is posted. For JMP regions that are posted, the JVM will be automatically recycled and then reloaded so that the next schedule of the program can obtain a refreshed copy of the Java program. When the JVM is reloaded, either the DFSJVMMS (JVMOPMAS=), DFSJVMEV (ENVIRON=), DFSJVMAP configuration members are refreshed or, if the //STDENV DD statement is used, the shell scripts that specify the Java environment variables and options are reloaded. For a Java program, any changes to the class path can also be refreshed by the UPDATE PGM START(REFRESH) command.

You can use the UPDATE PGM START(REFRESH) command when application programs have been modified and the program needs to pick up the new changes. When the program is rescheduled, it will pick up the application program changes.

When the regions are posted, the MPP and JMP regions remain available to schedule other work. Programs that are running in BMP regions receive a `QC` status when the UPDATE PGM START(REFRSH) command is issued. On receiving the `QC` status code,

- If the program handles the `QC` status code correctly and terminates, the BMP region is terminated and the region must be restarted.
- If the program does not handle the `QC` status code and does not terminate but it issues a subsequent GU/CHKP call, the program will receive a U330 abend during the processing of the subsequent GU/CHKP call. The region must be restarted.

When the region is restarted, it will pick up the updated program from the PGMLIB.

If a program has alias programs defined, both the program name and the alias program name need to be specified on the command so that a refreshed copy can be loaded at the next schedule of the programs.

The UPDATE PGM START(REFRESH) command is applicable in both DRD and non-DRD environments. The command is processed by each IMS that receives the command. The command processing is not coordinated among all IMS systems where the program is scheduled.

The UPDATE PGM START(REFRESH) command is not allowed for IMS internal programs such as DBF#FPU0 and for the IMS PSB for QCF MRQPSB.

When NAME(*) is specified with START(REFRESH) and OPTION(ALLRSP) is not specified, the command response consists of only the program names that are scheduled in the PWFI region or in a region with a WFI=YES transaction. When OPTION(ALLRSP) is specified, all the program names in IMS are returned.

The UPDATE PGM START(REFRESH) command is processed by all IMS systems that receive the command.

The START(REFRESH) keyword is not allowed with START(SCHD), START(TRACE), SET(), or STOP().

STOP
Specifies attributes that are to be stopped.

SCHD
Stops scheduling of the application program.

TRACE
Stops the tracing of the DL/I portion of Data Communications (DC) for the application program.

Usage notes

Resources exist for the life of the IMS unless they are deleted by using a DELETE command. Resources are recoverable across an IMS warm start or emergency restart. Resources are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE PGM command can be issued only through the OM API. This command applies to DB/DC, DBCTL and DCCTL systems.

This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE PGM command specified with SET() is not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined), except for SET(LOCK()) and SET(TRANSTAT()).

This command is recoverable.

The UPDATE commands are not allowed for IMS-defined resources, except with keywords START, STOP, SET(LOCK(ON|OFF)), and SET(TRANSTAT(Y|N)). For the UPDATE PGM command, this means Fast Path utility program DBF#FPU0.

The UPDATE PGM command changes a MODBLKS program to a dynamic program with a definition type of UPDATE, if the BMPTYPE, DOPT, FP, GPSB, LANG, RESIDENT, SCHDTYPE, or TRANSTAT value is changed.

Each program is updated individually, unlike the online change process where either all programs are updated or no programs are updated. Most runtime

resource definition values for a program can be updated only if the program is not in use. If the program is in use, the update fails.

You can update the status of a program (LOCK, START, STOP) while the program is in use.

If all the attributes specified by the UPDATE command are already defined for the resource, no update is actually made, no resources are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary processor usage when no action needs to be taken.

The following program attributes cannot be updated if online change for MODBLKS is enabled: BMPTYPE, DOPT, FP, GPSB, LANG, RESIDENT, SCHDTYPE.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE PGM command and the type-1 IMS commands that perform similar functions.

Table 517. Type-1 equivalents for the UPDATE PGM command

UPDATE command	Similar IMS type-1 command
UPDATE PGM START(REFRESH)	/PSTOP REGION TRAN *xxx*

Output fields

The following table shows the UPDATE PGM output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Long label
Contains the long label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 518. Output fields for the UPDATE PGM command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code
CCTXT	CCText	*error*	Completion code text that briefly explains the nonzero completion code.
ERRT	ErrorText	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
JOB	JobName	START(REFRESH)	Job Name of the program that is posted.

Table 518. Output fields for the UPDATE PGM command (continued)

Short label	Long label	Keyword	Meaning
MBR	MbrName	N/A	IMSplex member that built the output line.
PGM	PgmName	PGM	Program name.
RGNN	RgnNum	START(REFRESH)	Region Number where the program is scheduled.
RGNT	RgnType	START(REFRESH)	Region type where the program is scheduled.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 519. Return and reason codes for the UPDATE PGM command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000004'	X'00002008'	Invalid number of keywords. Either a SET, START, or STOP keyword is required.
X'00000008'	X'00002040'	START(REFRESH) was specified with START(TRACE) or START(SCHD).
X'00000008'	X'00002044'	The UPDATE PGM command is not processed because the same attribute value was specified for the START and STOP filters. The attributes SCHD and TRACE can be specified only with the START or STOP keyword, not both. Confirm that only one START | STOP attribute is specified on the command.
X'00000008'	X'00002048'	Invalid SET attribute.
X'00000008'	X'00002106'	DOPT(Y) is mutually exclusive with RESIDENT(Y).
X'00000008'	X'00002107'	DOPT(Y) is mutually exclusive with SCHDTYPE(PARALLEL).
X'00000008'	X'0000210B'	FP(E) is mutually exclusive with BMPTYPE(Y).
X'00000008'	X'0000210D'	FP(E) is mutually exclusive with LANG(JAVA).
X'00000008'	X'00002113'	GPSB(Y) is mutually exclusive with DOPT(Y).
X'00000008'	X'00002114'	GPSB(N) is mutually exclusive with LANG.
X'00000008'	X'00002115'	GPSB(Y) is mutually exclusive with RESIDENT(Y).
X'00000008'	X'00002132'	DOPT(Y) not supported with LANG except for LANG(JAVA).

Table 519. Return and reason codes for the UPDATE PGM command (continued)

Return code	Reason code	Meaning
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), only resources with nonzero completion codes are returned. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004017'	Not valid in DBCTL environment.
X'00000010'	X'00004024'	No Fast Path defined, FP(E) is invalid.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.
X'00000014'	X'00005010'	Latch could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 520. Completion codes for the UPDATE PGM command

Completion code	Completion code text	Meaning	Error text
0		Command completed successfully for program.	
10	NO RESOURCES FOUND	Program name is invalid, or the wildcard parameter specified does not match any program names.	
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this program. This could also mean this command, if the program is specified by more than one specific or wildcard parameter.	

Table 520. Completion codes for the UPDATE PGM command (continued)

Completion code	Completion code text	Meaning	Error text
29	DOPT=Y/RESIDENT=Y CONFLICT	Program update failed because dynamic DOPT(Y) option conflicts with resident RESIDENT(Y) option.	
2A	DOPT=Y/PARALLEL CONFLICT	Program update failed because dynamic DOPT(Y) option conflicts with parallel schedule SCHDTYPE(PARALLEL) option.	
2F	FP=E/BMPTYPE=Y CONFLICT	Program update failed because Fast Path exclusive FP(E) option conflicts with BMP type BMPTYPE(Y).	
36	RTC/FP=N PGM CONFLICT	Program update failed because non-Fast Path option FP(N) conflicts with Fast Path exclusive transaction that references this program.	
37	FP=E/LANG=JAVA CONFLICT	Program update failed because Fast Path exclusive FP(E) option conflicts with Java language LANG(JAVA).	
3E	FP=N/FP=E PGM CONFLICT	Program update failed because Fast Path exclusive FP(E) option conflicts with non-Fast Path transaction that references this program.	*tranname* (8 chars)
3F	FP=P/BMPTYPE=Y CONFLICT	Program update failed because BMP type program conflicts with Fast Path potential transaction that references this program.	*tranname* (8 chars)
40	PARLIM/ SCHDTYPE=SERIAL CONFLICT	The SCHDTYPE value cannot be changed to SERIAL because a transaction that references the program has a PARLIM value that is something other than 65535. The program definition is not updated.	
43	GPSB=Y/DOPT=Y CONFLICT	Program updated failed because generated PSB GPSB(Y) option conflicts with dynamic DOPT(Y) option.	

Table 520. Completion codes for the UPDATE PGM command (continued)

Completion code	Completion code text	Meaning	Error text
46	GPSB=N/LANG CONFLICT	Generated PSB option N (GPSB(N)) conflicts with the language option (LANG()). The program definition is not updated.	
47	GPSB=Y/RESIDENT=Y CONFLICT	Program update failed because generated PSB GPSB(Y) option conflicts with the resident RESIDENT(Y) option.	
48	Not allowed for IMS resources	The specified command is not allowed for an IMS defined resource.	
73	PROGRAM SCHEDULED	Program is scheduled.	
7A	RTC/FP=N PGM CONFLICT	Program to be updated as FP(N) conflicts with a routing that references the program. The routing code name is returned as error text. Suggested actions: Issue DELETE RTC command or DELETE TRAN (if associated with FP exclusive transaction) to delete the routing code.Or Issue UPDATE RTC command to update the program to another name.	*rtcodename* (8 chars)
97	DOPT=Y/LANG CONFLICT	Program update failed because dynamic option DOPT(Y) conflicts with LANG specified. DOPT(Y) is only supported with LANG(JAVA).	
99	NOT INITIALIZED	Update program failed because the program was not successfully initialized. You can use the QRY PGM SHOW(STATUS) command to return additional information regarding the reason for the NOTINIT status.	
B9	REQUIRES LANG	Program update failed because the DOPT(Y) or GPSB(Y) option specified requires language LANG to be specified or already defined.	

Table 520. Completion codes for the UPDATE PGM command (continued)

Completion code	Completion code text	Meaning	Error text
BA	Not allowed for MRQ PSB	The specified command is not allowed for the MRQ PSB, which is used for QCF.	
1D7	IMPORT CHANGE LIST ERROR	The resource is in the process of being imported from the change list in the IMSRSC repository or was not successfully imported from the change list. The resource cannot be updated until it is successfully imported from the repository.	
220	Program is not scheduled	The specified program name is not scheduled in any region and so it cannot be posted.	
221	Not supported for region type	The specified program name is scheduled in a region type that is not supported. IFP and JBP region types are not supported.	
222	Stop Region Pending	The UPDATE PGM START(REFRESH) command is not processed because a /STOP REGION command is pending for the region and the region is terminating.	
223	Not PWFI region or WFI=Y tran	The specified program name is not scheduled in a PWFI region or in a non-PWFI region with a WFI=N transaction and thus it is not posted.	
224	Preloaded in DFSMPLxx list	The specified program is preloaded in the dependent region because it is in the DFSMPLxx preload list and thus it is not posted.	
225	Non message driven BMP region	The specified program is scheduled in a non-message driven BMP region.	

Examples

The following are examples of the UPDATE PGM command:

Example 1 for UPDATE PGM command

TSO SPOC input:

```
UPDATE PGM NAME(BADNAME,AUTPSB2,CDEBS,BMP011,BAD*) SET(SCHDTYPE(PARALLEL))
```

TSO SPOC output:

```
Response for: UPDATE PGM NAME(BADNAME,AUTPSB2,CDEBS,BMP011,BAD*) SET(SCHDTYPE(PARALLEL))
PgmName   MbrName    CC CCText
AUTPSB2   IMS1        0
BAD*      IMS1       10 NO RESOURCES FOUND
BADNAME   IMS1       10 NO RESOURCES FOUND
BMP011    IMS1       2A DOPT=Y/PARALLEL CONFLICT
CDEBS     IMS1        0
```

OM API input:

```
CMD(UPDATE PGM NAME(BADNAME,AUTPSB2,CDEBS,BMP011,BAD*) SET(SCHDTYPE(PARALLEL)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.311 00:55:21.981467</statime>
<stotime>2006.311 00:55:21.981970</stotime>
<staseq>BFAADF0EA2A1B548</staseq>
<stoseq>BFAADF0EA2C12E48</stoseq>
<rqsttkn1>USRT011 10165521</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>PGM             </kwd>
<input>UPDATE PGM NAME(BADNAME,AUTPSB2,CDEBS,BMP011,BAD*)
SET(SCHDTYPE(PARALLEL)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="PGM" llbl="PgmName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>PGM(BADNAME ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>PGM(AUTPSB2 ) MBR(IMS1) CC(   0) </rsp>
<rsp>PGM(CDEBS   ) MBR(IMS1) CC(   0) </rsp>
<rsp>PGM(BMP011  ) MBR(IMS1) CC(  2A) CCTXT(DOPT=Y/PARALLEL CONFLICT)
```

```
</rsp>
<rsp>PGM(BAD*    ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update some programs so that they can be scheduled in multiple regions simultaneously. The update completed successfully for programs AUTPSB2 and CDEBS, as shown by the completion code 0. The update failed for program BMP011 with completion code 2A, indicating that the SCHDTYPE(PARALLEL) attribute conflicts with the DOPT(Y) attribute already defined for program BMP011. The update fails for program BADNAME and for parameter BAD* with completion code 10, since program BADNAME does not exist and no program name starts with BAD.

Example 2 for UPDATE PGM command

TSO SPOC input:

```
UPD PGM NAME(PGM*,PGX) START(REFRESH)
```

TSO SPOC output:

```
PgmName   MbrName  CC CCText                    RgnNum RgnType JobName
PGMA      IMS1     0                                 1 MPP     MPP1
PGMA      IMS1     0                                 3 MPP     MPP2
PGMA1     IMS1     221 Not supported for region type 2 IFP     IFP1
PGMA2     IMS1     223 Not PWFI region or WFI=Y tran 8 MPP     MPP3
PGMB      IMS1     220 Program is not scheduled
PGMMPL1   IMS1     224 Preloaded in DFSMPLxx list     9 MPP     MPP4
PGMNBMP1  IMS1     225 Non message driven BMP region 10 BMP     BMP1
PGMY      IMS1     0                                 4 JMP     JMP1
PGX       IMS1      10 Name not found
```

OM API input:

```
CMD(UPD PGM NAME(PG*) START(REFRESH))
```

Explanation: PGMA is scheduled in region IDs 1 and 3, and these regions are posted. PGMA1, which is scheduled in Region 2, cannot be refreshed because Region 2 is an IFP. PGMA2 is scheduled in Region 8, which is not PWFI and is not running a WFI=YES transaction, and thus is not posted. PGMB is not currently scheduled. PGMMPL1 is scheduled in MPP and is preloaded in DFSMPLxx PROCLIB member and thus is not posted. PGMNBMP1 is scheduled in a non-message driven BMP, which is not posted. PGMY is scheduled in Region 4, which is processing work, and this region will be posted when the work is complete. PGX is not a valid program.

Example 3 for UPDATE PGM command

TSO SPOC input:

```
UPD PGM NAME(*) START(REFRESH)
```

TSO SPOC output:

```
PgmName   MbrName  CC CCText                    RgnNum RgnType    JobName
PGM1      IMS1     0                                 6 MPP        MPP1
PGMA      IMS1     0                                 1 MPP        MPP2
PGMA      IMS1     0                                 3 MPP        MPP3
PGMA1     IMS1     221 Not supported for region type 2 IFP        IFP1
PGMA2     IMS1     223 Not PWFI region or WFI TRAN   8 MPP        MPP4
PGMY      IMS1     0                                 4 JMP        JMP1
```

| `CMD(UPD PGM NAME(PGM*) START(REFRESH))`

| **Explanation:** Because OPTION(ALLRSP) is not specified with NAME(*), only the
| program names that are scheduled in PWFI regions or regions with WFI=YES
| transactions are returned. APGM1 is scheduled in region ID 6, and this region is
| posted. PGMA is scheduled in region IDs 1 and 3, which are posted. PGMA1 is
| scheduled in Region 2, which cannot be posted because it is an IFP. PGMA2 is
| scheduled in Region 8, which is not PWFI and is not running a WFI=YES
| transaction, and so this region is not posted. PGMY is scheduled in Region 4,
| which is processing work, and this region will be posted when the work is
| complete.

Related concepts:

➦ How to interpret CSL request return and reason codes (System Programming
APIs)

➦ Diagnosing problems in the Queue Control Facility Message Requeuer
(Diagnosis)

Related tasks:

➦ Updating runtime application program resource and descriptor definitions
with the UPDATE command (System Definition)

Related reference:

➦ /LOCK command (Commands)

➦ Command keywords and their synonyms (Commands)

"/START PGM command" on page 858

"/STOP PGM command" on page 906

Chapter 28, "/TRACE commands," on page 961

"/UNLOCK PGM command" on page 1001

"/TRACE PGM command" on page 979

➦ IMS Queue Control Facility overview

UPDATE PGMDESC command

Use the UPDATE PGMDESC command to update program descriptors. A
descriptor is a model that can be used to create descriptors or resources.

Updating a descriptor changes only the attributes explicitly specified on the
UPDATE command. Attributes not specified retain their existing values. Any
program resource or descriptor can be created using this descriptor as a model, by
specifying the CREATE LIKE(DESC(*descriptor_name*)) command. Any descriptor or
resource that was already created using this descriptor is not updated.

Subsections:
- "Environment" on page 1277
- "Syntax" on page 1277
- "Keywords" on page 1277
- "Usage notes" on page 1281
- "Output fields" on page 1281
- "Return, reason, and completion codes" on page 1282

- "Examples" on page 1284

Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 521. Valid environments for the UPDATE PGMDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE PGMDESC	X	X	X
NAME	X	X	X
OPTION	X	X	X
SET	X	X	X

Syntax

Keywords

The following keywords are valid for the UPDATE PGMDESC command:

NAME

Specifies the 1-8 character name of the program descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the descriptor names that are processed. For NAME(*), command responses are returned only for the descriptor names that resulted in

an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the descriptor names that are processed.

OPTION
Specifies additional functions to be performed along with the command.

ALLRSP
Indicates that the response lines are to be returned for all descriptors that are processed on the command. The default action is to return response lines only for the descriptors that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SET
Specifies the attribute values to be changed.

BMPTYPE
BMP type option. Specifies whether the program executes in a BMP type region or not. A BMP type region can be a BMP region or a JBP region.

PSBs scheduled by DB2 stored procedures, by programs running under WebSphere Application Server, and by other users of the ODBA interface may be defined with BMPTYPE Y or N.

Y The program executes in a BMP type region. It can execute in an IMS BMP region or a JBP region. Any associated transactions are assigned normal and limit priority values of zero.

N The program does not execute in a BMP type region. It may execute in an IMS TM MPP, JMP, or IFP region, or it may use either the ODBA or DRA interface. This specification should be used for programs running in IMS TM MPP, JMP, and IFP regions, or PSBs scheduled by CICS programs using DBCTL and other users of the DRA interface. This is the default.

Keyword combination rules include the following:
• BMPTYPE(Y) and FP(E) are mutually exclusive.

DEFAULT(Y)
Specifies this descriptor as the default, which resets the existing default descriptor to DEFAULT(N). When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a program descriptor called DFSDSPG1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor. Since only one program descriptor can be the default at one time, only one descriptor name can be specified with DEFAULT(Y).

DOPT
Specifies the dynamic option.

N The PSB associated with this program is not located dynamically. The control blocks for the PSB must exist in an ACBLIB or the IMS catalog, otherwise the program is set to a NOTINIT status, and cannot be scheduled.

Y The PSB associated with this program is located dynamically. Each time the program that is associated with this PSB is scheduled, the latest copy of the PSB is loaded. When the program terminates, the PSB is deleted from the PSB pool.

Until the PSB is required to process a transaction, the ACB for the PSB does not need to be in the ACBLIB or, if the IMS management of ACBs is enabled, in the IMS catalog. A new version of the PSB can be defined and added to the ACBLIB or the IMS catalog at any time. IMS loads the new version of the ACB the next time the PSB is scheduled.

A DOPT PSB cannot be scheduled if the control blocks for the DBDs that the PSB references are not in the ACBLIB or the IMS catalog.

Keyword combination rules include the following:
- DOPT(Y) and GPSB(Y) are mutually exclusive.
- DOPT(Y) and LANG(JAVA) is a valid combination.
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- RESIDENT(Y) and DOPT(Y) are mutually exclusive.
- SCHDTYPE(PARALLEL) and DOPT(Y) are mutually exclusive.

FP Specifies the Fast Path option.

E The program is a Fast Path-exclusive application program. This implicitly defines a wait-for-input (WFI) application program. Either a transaction or a routing code that can be assigned by the user Input Edit/Routing exit routine must be defined for the Fast Path-exclusive application, in order for this program to be usable.

N The program is not a Fast Path-exclusive application program. When FP(N) is specified, any attempt to use Fast Path resources or commands will yield unpredictable results.

Keyword combination rules include the following:
- FP(E) requires Fast Path to be defined.
- LANG(JAVA) and FP(E) are mutually exclusive.
- BMPTYPE(Y) and FP(E) are mutually exclusive.

GPSB
Specifies the generated PSB option.

N The PSB associated with the program is not generated by IMS. The control blocks for the PSB must exist in the ACBLIB or, if the IMS management of ACBs is enabled, in the IMS catalog, otherwise the program is set to a NOTINIT status, and cannot be scheduled.

Y The PSB and application control block (ACB) that are associated with the program are generated by IMS. The scheduling process of all environments generates a PSB that contains an I/O PCB and an alternate modifiable PCB.

You do not need to generate the PSB and ACB or submit DDL to define the PSB. The ACB for the PSB is not loaded from either ACBLIB or, if the IMS management of ACBs is enabled, the IMS catalog. Thus, I/O to retrieve the ACB from either the ACBLIB or the IMS catalog is eliminated.

The generated PSB contains an I/O PCB named IOPCBbbb and a modifiable, alternate PCB named TPPCB1bb. With an alternate modifiable PCB, an application can use the CHNG call to change the output destination and send output to a destination other than the input destination.

Keyword combination rules include the following:
- DOPT(Y) and GPSB(Y) are mutually exclusive.
- DOPT(Y) and LANG(JAVA) is a valid combination.
- GPSB(Y) requires LANG.
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- RESIDENT(Y) and GPSB(Y) are mutually exclusive.

LANG
Specifies the language interface of the application program for a GPSB.

The LANG parameters and their meanings are identified in the following table.

LANG parameter	Meaning
ASSEM	Assembler
COBOL	COBOL
JAVA	Java
PASCAL	Pascal
PLI	PL/I

Keyword combination rules include the following:
- LANG is invalid with GPSB(N), except if DOPT(Y) and LANG(JAVA).
- DOPT(Y) and LANG(JAVA) is a valid combination.
- LANG(JAVA), DOPT(Y), and GPSB(N) is a valid combination.
- FP(E) and LANG(JAVA) are mutually exclusive.

RESIDENT
Specifies the resident option.

N The PSB associated with the program is made to be not resident in storage. The PSB is loaded at scheduling time.

Y The PSB associated with the program is made to be resident in storage immediately. IMS loads the PSB and initializes it. The PSB is removed from the PSB pool, if applicable. A resident PSB is accessed in local storage, which eliminates I/O to retrieve the ACB.

Keyword combination rules include the following:
- DOPT(Y) and RESIDENT(Y) are mutually exclusive.
- GPSB(Y) and RESIDENT(Y) are mutually exclusive.

SCHDTYPE

Specifies whether this application program can be scheduled into more than one message region or batch message region simultaneously.

PARALLEL
The application program can be scheduled in multiple regions simultaneously.

SERIAL
The application program can be scheduled in only one region at a time.

Updating this attribute to SCHDTYPE(SERIAL) is rejected if a transaction referencing this program is defined with a parallel limit count other than 65535.

Keyword combination rules include the following:

- DOPT(Y) and SCHDTYPE(PARALLEL) are mutually exclusive.

TRANSTAT
Specifies whether transaction level statistics should be logged. The value specified has meaning only if the program is a JBP or a non-message driven BMP. If Y is specified, transaction level statistics are written to the log in a X'56FA' log record.

N   Transaction level statistics should not be logged.

Y   Transaction level statistics should be logged.

The TRANSTAT keyword on the UPDATE PGMDESC command gives the user the ability to override the system default or the current value of the TRANSTAT parameter. If the TRANSTAT keyword is omitted on the UPDATE PGMDESC command, the current transaction level statistics setting is unchanged for the program.

Usage notes

If all the attributes specified by the UPDATE command are already defined for the descriptor, no update is actually made, no descriptors are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

Descriptors exist for the life of the IMS unless they are deleted using a DELETE command. Descriptors are recoverable across an IMS warm start or emergency restart. Descriptors are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE PGMDESC command can be issued only through the OM API. This command applies to DB/DC, DBCTL and DCCTL systems.

This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE PGMDESC commands are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

This command is recoverable.

If the descriptor is the IMS-defined program descriptor (DFSDSPG1) the only attributes that can be updated are the DEFAULT (SET(DEFAULT(Y))) and the TRANSTAT (SET(TRANSTAT(Y|N|))) attributes.

Each descriptor is updated individually. Individual updating does not work like online change where either all descriptors are updated or no descriptors are updated.

Output fields

The following table shows the UPDATE PGMDESC output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
> Provides a brief description of the output field.

Table 522. Output fields for the UPDATE PGMDESC command

Short label	Keyword	Meaning
CC	N/A	Completion code
CCTXT	*error*	Completion code text that briefly explains the nonzero completion code.
DESC	PGMDESC	Program descriptor name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
MBR	N/A	IMSplex member that built the output line.
OLDDEF	PGMDESC	Old default descriptor name, if this descriptor is updated to be the default by specifying DEFAULT(Y). The old default descriptor is no longer the default.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 523. Return and reason codes for the UPDATE PGMDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000004'	X'00002008'	Invalid number of keywords. A SET keyword is required.
X'00000008'	X'00002048'	Invalid SET attribute.
X'00000008'	X'00002106'	DOPT(Y) mutually exclusive with RESIDENT(Y).
X'00000008'	X'00002107'	DOPT(Y) mutually exclusive with SCHDTYPE(PARALLEL).
X'00000008'	X'0000210B'	FP(E) mutually exclusive with BMPTYPE(Y).
X'00000008'	X'0000210D'	FP(E) mutually exclusive with LANG(JAVA).
X'00000008'	X'00002113'	GPSB(Y) mutually exclusive with DOPT(Y).
X'00000008'	X'00002114'	GPSB(N) is mutually exclusive with LANG.
X'00000008'	X'00002115'	GPSB(Y) mutually exclusive with RESIDENT(Y).

Table 523. Return and reason codes for the UPDATE PGMDESC command (continued)

Return code	Reason code	Meaning
X'00000008'	X'00002132'	DOPT(Y) not supported with LANG except for LANG(JAVA).
X'00000008'	X'00002133'	Multiple name parameters were specified with DEFAULT(Y). Only one descriptor can be the default at a time.
X'0000000C'	X'00003000'	Command was successful for some descriptors but failed for others. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), only descriptors with non-zero completion codes are returned. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined, FP(E) is invalid.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 524. Completion codes for the UPDATE PGMDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for program descriptor.
10	NO RESOURCES FOUND	Program descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this program descriptor. This could also mean this command, if the descriptor is specified by more than one specific or wildcard parameter.
29	DOPT=Y/RESIDENT=Y CONFLICT	Program descriptor update failed because dynamic DOPT(Y) option conflicts with resident RESIDENT(Y) option.

Table 524. Completion codes for the UPDATE PGMDESC command (continued)

Completion code	Completion code text	Meaning
2A	DOPT=Y/PARALLEL CONFLICT	Program descriptor update failed because dynamic DOPT(Y) option conflicts with parallel schedule SCHDTYPE(PARALLEL) option.
2F	FP=E/BMPTYPE=Y CONFLICT	Program descriptor update failed because Fast Path exclusive FP(E) option conflicts with BMP type BMPTYPE(Y).
37	FP=E/LANG=JAVA CONFLICT	Program descriptor update failed because Fast Path exclusive FP(E) option conflicts with Java language LANG(JAVA).
40	PARLIM/ SCHDTYPE=SERIAL CONFLICT	The SCHDTYPE value cannot be changed to SERIAL because a transaction that references the program has a PARLIM value that is something other than 65535. The program definition is not updated.
43	GPSB=Y/DOPT=Y CONFLICT	Program descriptor updated failed because generated PSB GPSB(Y) option conflicts with dynamic DOPT(Y) option.
46	GPSB=N/LANG CONFLICT	Generated PSB option N (GPSB(N)) conflicts with the language option (LANG()). The program definition is not updated.
47	GPSB=Y/RESIDENT=Y CONFLICT	Program descriptor update failed because generated PSB GPSB(Y) option conflicts with the resident RESIDENT(Y) option.
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for the IMS descriptor or resource. DFSDSPG1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
8A	WILDCARD PARAMETER NOT SUPPORTED	A wildcard parameter was specified with DEFAULT(Y). Only one descriptor can be the default at a time.
97	DOPT=Y/LANG CONFLICT	Program descriptor update failed because dynamic option DOPT(Y) conflicts with LANG specified. DOPT(Y) is only supported with LANG(JAVA).
1D7	IMPORT CHANGE LIST ERROR	The descriptor is in the process of being imported from the change list in the IMSRSC repository or was not successfully imported from the change list. The descriptor cannot be updated until it is successfully imported from the repository.

Examples

The following are examples of the UPDATE PGMDESC command:

Example 1 for UPDATE PGMDESC command

TSO SPOC input:

```
UPDATE PGMDESC NAME(*) SET(BMPTYPE(Y)) OPTION(ALLRSP)
```

TSO SPOC output:

```
Response for: UPDATE PGMDESC NAME(*) SET(BMPTYPE(Y)) OPTION(ALLRSP)
DescName MbrName    CC CCText
DESC001  IMS1        0
DESC002  IMS1        0
DESC003  IMS1        0
DESC004  IMS1        0
DESC005  IMS1        0
DFSDSPG1 IMS1       48 NOT ALLOWED FOR IMS RESOURCE
DOPTDESC IMS1        0
FPEDESC  IMS1       2F FP=E/BMPTYPE=Y CONFLICT
GPSBDESC IMS1        0
PARLDESC IMS1        0
RESDESC  IMS1        0
TLSDESC  IMS1        0
```

OM API input:

```
CMD(UPDATE PGMDESC NAME(*) SET(BMPTYPE(Y)) OPTION(ALLRSP))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2006.312 23:30:46.559300</statime>
<stotime>2006.312 23:30:46.559923</stotime>
<staseq>BFAD4FE141E44D08</staseq>
<stoseq>BFAD4FE1420B3F48</stoseq>
<rqsttkn1>USRT011 10153046</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>PGMDESC         </kwd>
<input>UPDATE PGMDESC NAME(*) SET(BMPTYPE(Y)) OPTION(ALLRSP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
```

```
<cmdrspdata>
<rsp>DESC(DESC004 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC005 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC001 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DOPTDESC) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(RESDESC ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DFSDSPG1) MBR(IMS1) CC(  48) CCTXT(NOT ALLOWED FOR IMS RESOURCE) </rsp>
<rsp>DESC(TLSDESC ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC002 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(PARLDESC) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC003 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1) CC(  2F) CCTXT(FP=E/BMPTYPE=Y CONFLICT)
 </rsp>
<rsp>DESC(GPSBDESC) MBR(IMS1) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update all program descriptors to be BMPTYPE(Y). The update completed successfully for most program descriptors, as shown by completion code 0. The update failed for IMS-defined descriptor DFSDSPG1 with completion code 48, because the only attribute that can be updated for DFSDSPG1 is DEFAULT(Y). The update failed for program descriptor FPEDESC with completion code 2F, because the BMPTYPE(Y) attribute conflicts with the Fast Path exclusive FP(E) attribute already defined for program descriptor FPEDESC.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

Command keywords and their synonyms (Commands)

UPDATE POOL command

Use the UPDATE POOL command to dynamically add, update, or delete OSAM or VSAM subpools, or to specify values associated with storage used by the Fast Path 64-bit buffer manager.

Subsections:
- "Environment"
- "Syntax" on page 1287
- "Keywords" on page 1287
- "Usage notes for UPDATE POOL TYPE(FPBP64)" on page 1288
- "Usage notes for UPDATE POOL TYPE(DBAS)" on page 1289
- "Output fields" on page 1290
- "Return, reason, and completion codes" on page 1291
- "Examples" on page 1293

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 525. Valid environments for the UPDATE POOL command and keywords.

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE POOL	X	X	X

Table 525. Valid environments for the UPDATE POOL command and keywords (continued).

Command / Keywords	DB/DC	DBCTL	DCCTL
MEMBER	X	X	
SECTION	X	X	
SET	X	X	X
TYPE	X	X	X

Restriction: The dynamic database buffer pools function is not supported in a DBCTL warm standby environment.

Syntax
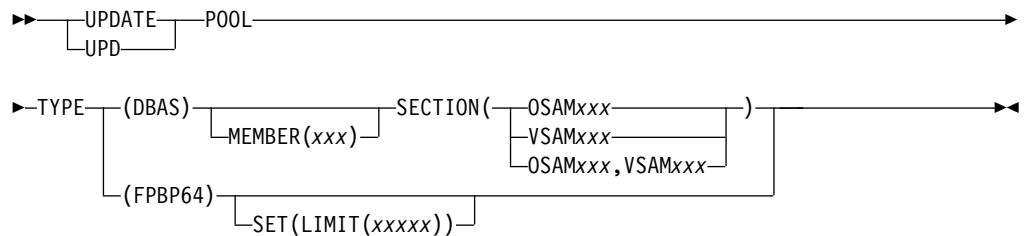
Keywords

The following keywords are valid for the UPDATE POOL command:

TYPE()

This required keyword specifies which type of UPDATE POOL command to issue.

DBAS

This option specifies that the command dynamically adds, updates, or deletes OSAM or VSAM subpools. TYPE(DBAS) is mutually exclusive with other options on the TYPE() keyword. The command is rejected with a syntax error if multiple options are specified.

The UPDATE POOL TYPE(DBAS) command is not valid for DCCTL environments.

FPBP64

This option specifies that the command sets or changes values for storage used by the Fast Path 64-bit buffer manager. TYPE(FPBP64) is mutually exclusive with other options on the TYPE() keyword. The command is rejected with a syntax error if multiple options are specified.

SET()

Specifies the attribute values to be changed. This keyword is valid only if TYPE(FPBP64) is specified.

LIMIT

Specifies the maximum amount of 64-bit storage used by the Fast Path 64-bit buffer manager.

xxxxx

You can specify the value in bytes, kilobytes, or megabytes, as follows:

- *xxxxx* - Value in bytes. The maximum value is 2,147,483,647 bytes.
- *xxxxx*K - Value in kilobytes. The maximum value is 2,097,151 KB.

- *xxxxx*M - Value in megabytes. The maximum value is 2,047 MB.

  The valid range is between 2GB-1 megabytes and the initial size of the buffer pool at IMS startup.

You can also set this value at IMS startup by specifying the parameter FPBP64M on the DFSDFxxx member of the IMS PROCLIB data set.

MEMBER(*xxx*)
This keyword is valid only if TYPE(DBAS) is specified. It is optional. The *xxx* on the MEMBER keyword specifies a 1- to 3-alphanumeric character value that represents the suffix to a DFSDFxxx member name. The identified DFSDFxxx member is read in and parsed for OSAM and VSAM subpool definitions. If the keyword is not specified, the DFSDFxxx member specified by the DBA on system initialization is used as the default. The DFSDFxxx member specified on system initialization can be determined from message DFS1929I on the system console or log. The command fails if the MEMBER keyword is specified with an invalid *xxx* value or if the DFSDFxxx member is not found.

Note: This keyword is optional and by default the command will always refer to the DFSDFxxx member specified on system initialization. Normally the DFSDFxxx member specified on system initialization should be the only one available. The purpose of this keyword is to allow the capability to have the command point to an alternate DFSDFxxx member (different from the system default) for test purposes. In other cases, the MEMBER keyword is expected to be omitted.

SECTION()
This keyword is valid only if TYPE(DBAS) is specified. It is required if TYPE(DBAS) is specified. This keyword specifies the section names in the DFSDFxxx member to read in for subpool definitions. It must be followed with a valid section name. One or both of the following section name types can be specified with this keyword:

OSAM*xxx*
Specifies the OSAM section in the DFSDFxxx member to read and process. The *xxx* value is the 1- to 3-alphanumeric character suffix of the OSAM section name. The indicated OSAM section must exist in the DFSDFxxx member and must contain valid OSAM subpool definition statements. Only one OSAM section name can be specified on the command.

VSAM*xxx*
Specifies the VSAM section in the DFSDFxxx member to read and process. The *xxx* value is the 1- to 3-alphanumeric character suffix of the VSAM section name. The indicated VSAM section must exist in the DFSDFxxx member and must contain valid VSAM subpool definition statements. Only one VSAM section name can be specified on the command.

If you are reassigning the DBD association (for example, changing a database data set association from one subpool to another subpool), you must stop and restart the database. The reason for the stop and restart is that the subpool assignment is done during data set open time.

Usage notes for UPDATE POOL TYPE(FPBP64)

The value that can be changed by the UPDATE POOL TYPE(FPBP64) SET(LIMIT(*xxxxx*)) command is not recoverable.

Enter the UPDATE POOL TYPE(FPBP64) SET(LIMIT(*xxxxx*)) command after an IMS restart, if needed. Because this command updates the value set by FPBP64M in the DFSDFxxx member, you might want to update FPBP64M with a new value before IMS cold restart.

Usage notes for UPDATE POOL TYPE(DBAS)

Determination of success, failure, or partial success of the command

There can be multiple change requests in the OSAMxxx or VSAMxxx sections of the DFSDFxxx member of the IMS PROCLIB data set. The UPDATE POOL TYPE(DBAS) command processes the change requests serially. When the UPDATE POOL TYPE(DBAS) command is completed, the results can be success, failure, or partial success.

The UPDATE POOL TYPE(DBAS) command cannot be canceled or aborted after it is issued.

The command can complete either before or after the TSO SPOC timeout occurs:
*   If it completes before the TSO SPOC timeout occurs, the results are shown in the TSO SPOC with completion codes next to each requested change indicating success, failure, or partial success of the request.
*   If it completes after the TSO SPOC timeout occurs, the command continues to run in the background. The TSO SPOC is no longer available to display the completion codes for each change requested by the command.

    A series of targeted QUERY POOL TYPE(DBAS) commands can be issued to determine the success or failure of the requested changes.

    You can also use the QUERY POOL TYPE(DBAS) command after an emergency restart to determine the success or failure of the requested changes.

    It is also possible to use the Operations Manager (OM) audit trail to determine which changes succeed, which changes fail, and which changes succeed partially. The token rqsttkn1 can be used to associate the commands in the OM audit trail to the command responses.

Command completion if storage is unavailable

There are circumstances where the storage requested by the UPDATE POOL TYPE(DBAS) command cannot be satisfied. Because IMS cannot cancel or abort the command, it attempts to find the next best solution to satisfy the command, which means the command might result in less storage being allocated for the requested subpool change.

When the storage requested by the UPDATE POOL TYPE(DBAS) command cannot be satisfied, both OSAM and VSAM make a sequence of attempts to find a smaller amount of storage.

If a reduced amount of buffer pool storage is allocated as a result of the UPDATE POOL TYPE(DBAS) command, an X'EC' completion code is displayed to indicate the condition.

The minimum number of buffers for a VSAM buffer pool is 3. If a VSAM pool cannot allocate this minimum number of buffers, the databases that are using this pool are left without a buffer assignment. An internal /DBRECOVERY command is

issued for all these databases. The UPD POOL command must be issued again to create the buffer pools. A subsequent /STA DB command must be issued to reestablish the buffer pool connection.

Effect of long-running BMPs on the command

For VSAM, a long-running batch message processing (BMP) program that does not take any checkpoint can affect the ability of the command to complete, since the command cannot be canceled or aborted. Either the subpool is eventually freed up by the BMP and the command can complete, or the subpool is not freed up by the BMP and the command waits indefinitely for the subpool to become available for the change. This effect applies to any subpool in the same VSAM local shared resource pool (that is, under the same POOLID statement).

OSAM and VSAM have different mechanisms for freeing up the target subpool. In general, OSAM waits for the target subpool to become unowned, while VSAM waits for DL/I activity for the target subpool to reach commit points. Since there is no timeout for this command, all partition specification tables (PSTs) with subpools affected by this command wait until the command completes.

Spreading database data sets across many subpools tends to lessen the impact of a change to any one of these subpools.

Buffer pool statistics and database data set reassignment

Buffer pool statistics are handled differently for VSAM and for OSAM following an UPDATE POOL TYPE(DBAS) command. For VSAM, the buffer pool statistics are reset and the old statistics are not carried over. It is advisable to issue a QUERY POOL command for the VSAM buffer pool statistics before issuing the command. The OSAM statistics are carried over and are not reset with the command.

When a database data set is reassigned from one buffer pool to a different buffer pool, the database data set must be closed and reopened. For OSAM, the closing and reopening of the database data set must be done explicitly. In other words, it is not performed as part of the command. For VSAM, the database data set must also be closed and reopened. However, if there is a corresponding change to the target buffer pool along with the reassignment of the database data set (for example, an increase in buffers), then the closing and opening of the database data set is done implicitly by the command.

Output fields

The following table shows the UPDATE POOL output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the column heading for the output field in the formatted output.

SHOW parameter
> Identifies the parameter on the SHOW keyword that caused the field to be generated. N/A appears for output fields that are always returned. *Error* appears for output fields that are returned only in case of an error.

Meaning
> Provides a brief description of the output field.

Table 526. Output fields for the UPDATE POOL command

Short label	Long label	SHOW parameter	Meaning
BUFS	BufSize	*Error*	Buffer size
CC	CC	N/A	Completion code. The completion code indicates whether IMS was able to process the command for the specified resource. The completion code is always returned.
CCTXT	CCText	*Error*	Completion code text that briefly explains the nonzero completion code. This field is returned only for an error completion code.
ERRT	ErrorText	*Error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
ID	PoolId	*Error*	OSAM subpool or VSAM shared pool ID
MBR	MbrName	N/A	The IMSplex member that built the output line. The IMS identifier of the IMS for which the database information is displayed. The IMS identifier is always returned.
SEC	Section	*Error*	Section of the DFSDFxxx member that is processed
STMT	Stmt	*Error*	OSAM or VSAM definition statement

The following are examples of the output headers for the UPDATE POOL TYPE(DBAS) command:

- Output headers for a successful UPDATE POOL TYPE(DBAS) command:

```
Section  MbrName CC
-------- ------- --
```

- Output headers for the UPDATE POOL TYPE(DBAS) command with errors:

```
Section  MbrName CC CCText Stmt   BufSize PooId  ErrorText
-------- ------- -- ------ ------ ------  -----  ---------
```

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 527. Return and reason codes for the UPDATE POOL command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The command completed successfully. The command output contains a line for each resource, accompanied by its completion code.
X'00000008'	X'00002004'	Invalid command keyword or invalid command keyword combination
X'00000008'	X'00002014'	Invalid character in the member name.
X'00000008'	X'00002040'	Invalid option was specified.
X'00000008'	X'00002122'	Invalid PDS member name. The specified DFSDFxxx member was not found.

Table 527. Return and reason codes for the UPDATE POOL command (continued)

Return code	Reason code	Meaning
X'00000008'	X'0000212B'	Invalid section name.
X'00000008'	X'0000212C'	There was a read error on the PDS member.
X'0000000C'	X'00003000'	At least one request was successful.
X'0000000C'	X'00003004'	No requests were successful.
X'00000010'	X'00004000'	IMS restart in progress.
X'00000010'	X'0000400C'	The command failed, because it is not valid on an XRF alternate IMS.
X'00000010'	X'00004014'	The command failed, because it is not valid on an RSR tracking IMS.
X'00000010'	X'00004016'	The command failed, because it is not valid in a DCCTL environment.
X'00000010'	X'0000401C'	The command failed, because it is not valid on the FDR region.
X'00000010'	X'00004024'	The command failed, because FP=N is in the startup parameter.
X'00000010'	X'00004404'	The command failed, because FPBP64=N is in the startup parameter.
X'00000010'	X'0000440C'	A smaller storage usage limit than what is currently in use was specified on the command. The storage usage limit is not changed.
X'00000014'	X'00005000'	IMODULE GETSTOR error.
X'00000014'	X'00005004'	DFSOCMD0 GETBUF error.
X'00000014'	X'00005FFF'	Unexpected internal error while initializing environment.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 528. Completion codes for the UPDATE POOL command

Completion code	Completion code text	Meaning
0		The command completed successfully.
8	COMMAND COMPLETE FOR SOME	The command completed successfully. Duplicate parameters are ignored.
B		A smaller storage usage limit than what is currently in use was specified on the UPDATE POOL TYPE(FPBP64) SET(LIMIT(*xxxxx*)) command. The storage usage limit is not changed.
10	NO RESOURCES FOUND	The specified resource (subpool, DBD, or DCB) is not found or not defined.
11	DUPLICATE RESOURCE ALREADY EXISTS	Duplicate IOBF or POOLID statements specified. The duplicates are ignored.
4A	IN USE	The specified subpool is currently in use.
60	GETMAIN STORAGE ERROR	Failed on a GETMAIN call for the subpool ID table.

Table 528. Completion codes for the UPDATE POOL command (continued)

Completion code	Completion code text	Meaning
64	GETSTOR STORAGE ERROR	Failed on a GETSTOR call for buffers, subpool, or coupling facility.
B2	IMS STATE ERROR	IMS is not in a state to perform an IMS shutdown.
EA	DYNAMIC BUFFER POOL FAILURE	The hiperspace specified for the buffer size is less than 4 KB. The DSNUM value on the DBD statement is invalid. This resulted in one of the following: • Locked buffers • Excess locked buffers • Reduced buffer allocation • Subpool or buffer page fix error • CF error of invalid buffer size
EB	RESOURCE UNABLE TO BE QUIESCED	Activity against a resource was unable to be quiesced by the command. The resource can be a buffer or a subpool.
EC	REDUCED BUFFER ALLOCATION	Unable to allocate the number of buffers specified. A reduced number of buffers is used.
ED	MINIMUM BUFFER ALLOCATION	Unable to allocate the number of buffers specified. The minimum number of buffers is used.
1C0	POOLID ERROR IN DFSDF MEMBER	Unable to process the POOLID statement in the DFSDFxxx member of the IMS PROCLIB data set. Refer to the ErrorText column in the command output for further information on the error.
1C1	DBD ERROR IN DFSDF MEMBER	Unable to process the DBD statement in the DFSDFxxx member of the IMS PROCLIB data set. Refer to the ErrorText column in the command output for further information on the error.

Examples

The following are examples of the UPDATE POOL command:

Example 1 for UPDATE POOL command

TSO SPOC input:
```
UPD POOL TYPE(FPBP64) SET(LIMIT(1000M))
```

TSO SPOC output:
```
Pool Type       MbrName   CC
FPBP64          SYS3       0
```

OM API input:
```
CMD(UPD POOL TYPE(FPBP64) SET(LIMIT(1000M)))
```

OM API output:

```
<imsout>
 <ctl>
 <omname>OM1OM    </omname>
 <omvsn>1.4.0</omvsn>
 <xmlvsn>20   </xmlvsn>
 <statime>2008.318 21:54:49.340430</statime>
 <stotime>2008.318 21:54:49.340724</stotime>
 <staseq>C34A9721FEE0EC72</staseq>
 <stoseq>C34A9721FEF34C72</stoseq>
 <rqsttkn1>USRT004 10135449</rqsttkn1>
 <rc>00000000</rc>
 <rsn>00000000</rsn>
 </ctl>
 <cmd>
 <master>SYS3     </master>
 <userid>USRT004 </userid>
 <verb>UPD </verb>
 <kwd>POOL          </kwd>
 <input>UPD POOL TYPE(FPBP64) SET(LIMIT(1000M)) </input>
 </cmd>
 <cmdrsphdr>
 <hdr slbl="BTYP" llbl="Pool Type" scope="LCL" sort="a" key="1"
  scroll="no" len="8" dtype="CHAR" align="left" />
 <hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="1"
  scroll="yes" len="8" dtype="CHAR" align="left" />
 <hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="no"
  len="4" dtype="CHAR" align="right" />
 <hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
  scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
 </cmdrsphdr>
 <cmdrspdata>
 <rsp>btyp(FPBP64) MBR(SYS3    ) CC(   0)</rsp>
 </cmdrspdata>
</imsout>
```

Explanation: The buffer usage limit for the Fast Path 64-bit buffer manager was
changed to 1000M (1 GB) and the command ended successfully.

Example 2 for UPDATE POOL command

TSO SPOC input:
```
UPDATE POOL TYPE(DBAS) MEMBER(001) SECTION(OSAM001,VSAMTUE)
```

TSO SPOC output:
```
Section  MbrName   CC
OSAM001  IMS1       0
VSAMTUE  IMS1       0
```

Explanation: This is an example of a successful invocation of the UPDATE POOL
command to dynamically reconfigure OSAM and VSAM subpools using the
OSAM001 and VSAMTUE sections of the DFSDFxxx member.

Example 3 for UPDATE POOL command

TSO SPOC input:
```
UPDATE POOL TYPE(DBAS) MEMBER(001) SECTION(OSAM001,VSAMTUE)
```

TSO SPOC output:
```
Section  MbrName  CC CCText                  Stmt  BufSize PoolID ErrorTxt
OSAM001  IMS1      8 COMMAND COMPLETE FOR SOME
OSAM001  IMS1     11 RESOURCE ALREADY EXISTS IOBF  1024    OSM2   PARSED ENTRY INV
VSAMTUE  IMS1      0
```

Explanation: This is an example of a partially successful invocation of the UPDATE POOL command to dynamically reconfigure OSAM and VSAM subpools using the OSAM001 and VSAMTUE sections of the DFSDFxxx member. Some (but not all) of the definitions within the OSAM001 section were successfully implemented.

Example 4 for UPDATE POOL command

TSO SPOC input:

```
UPDATE POOL TYPE(DBAS) MEMBER(001) SECTION(OSAM123,VSAM123)
```

TSO SPOC output:

```
Section  MbrName   CC CCText                      Stmt   BufSize PoolID ErrorTxt
OSAM123  IMS1       0
VSAM123  IMS1       C  COMMAND COMPLETE FOR NONE
VSAM123  IMS1      1C0 POOLID ERROR IN DFSDF MEMBER   POOLID   4096  VB10   DLET POOL 0 INV
VSAM123  IMS1      1C0 POOLID ERROR IN DFSDF MEMBER   POOLID   8192  VB10   DLET POOL 0 INV
VSAM123  IMS1      1C0 POOLID ERROR IN DFSDF MEMBER   POOLID   1024  VB11   DLET POOL 0 INV
VSAM123  IMS1      1C0 POOLID ERROR IN DFSDF MEMBER   POOLID   8192  VB11   DLET POOL 0 INV
VSAM123  IMS1      1C0 POOLID ERROR IN DFSDF MEMBER   POOLID   2048  VB12   DLET POOL 0 INV
VSAM123  IMS1      1C0 POOLID ERROR IN DFSDF MEMBER   POOLID   8192  VB12   DLET POOL 0 INV
VSAM123  IMS1      1C0 POOLID ERROR IN DFSDF MEMBER   POOLID  16384  VB12   DLET POOL 0 INV
VSAM123  IMS1      1C0 POOLID ERROR IN DFSDF MEMBER   POOLID  32768  VB12   DLET POOL 0 INV
```

Explanation: This is an example of a failed invocation of the UPDATE POOL command to dynamically reconfigure OSAM and VSAM subpools using the OSAM123 and VSAM123 sections of the DFSDFxxx member. None of the definitions within the VSAM123 section were successfully implemented.

Example 5 for UPDATE POOL command

TSO SPOC input:

```
UPDATE POOL TYPE(DBAS) MEMBER(001) SECTION(OSAMFEB,VSAMTUE)
```

TSO SPOC output:

```
Log for . . : UPDATE POOL TYPE(DBAS) MEMBER(001) SECTION(OSAMFEB,VSAMFEB)

IMSplex . . . . . : PLEX1
Routing . . . . . :
Start time. . . . : 2011.013 22:10:26.70
Stop time . . . . : 2011.013 22:10:26.78
Return code . . . : 0200000C
Reason code . . . : 00003004
Reason text . . . : No requests were successful.
Command master. . : IMS1


          Return    Reason
MbrName   Code      Code      Reason text
--------  --------  --------  --------------
IMS1      00000008  0000212B  Invalid section
```

Explanation: This is an example of a failed invocation of the UPDATE POOL command to dynamically reconfigure OSAM and VSAM subpools using the OSAM123 and VSAMTUE sections of the DFSDFxxx member. A failure occurred because invalid section names were specified when issuing the UPDATE POOL command.

Example 6 for UPDATE POOL command

TSO SPOC input:

```
UPDATE POOL TYPE(DBAS) MEMBER(001) SECTION(OSAM001,VSAMTUE)
```

TSO SPOC output:

```
Log for . . : UPDATE POOL TYPE(DBAS) MEMBER(001) SECTION(OSAM001,VSAMTUE)

IMSplex . . . . . : PLEX1
Routing . . . . . :
Start time. . . . : 2011.013 22:16:35.12
Stop time . . . . : 2011.013 22:16:35.36
Return code . . . : 0200000C
Reason code . . . : 00003008
Reason text . . . : None of the clients were successful.
Command master. . : IMS1

            Return      Reason
MbrName     Code        Code      Reason text
--------    --------    --------  --------------
IMS1        00000008    0000212C  PDS member read error

MbrName  Messages
-------- -----------------------
IMS1     BPE0003E AN ERROR OCCURRED PARSING PROCLIB MEMBER DFSDF001
IMS1     BPE0003E AT LINE 29, CHARACTER 14
IMS1     BPE0003E FAILING TEXT: "VXRBF=(2048,9,D)"
IMS1     BPE0003E INVALID KEYWORD DETECTED
```

Explanation: This is an example of a failed invocation of the UPDATE POOL command to dynamically reconfigure OSAM and VSAM subpools using the OSAM001 and VSAMTUE sections of the DFSDFxxx member. A parsing error occurred while reading the DFSDFxxx member.

Example 7 for UPDATE POOL command

TSO SPOC input:

```
UPDATE POOL TYPE(DBAS) SECTION(OSAM123) MEMBER(001)
```

TSO SPOC output:

```
Log for . . : UPDATE POOL TYPE(DBAS) SECTION(OSAM123) MEMBER(0...

IMSplex . . . . . : PLEX1
Routing . . . . . :
Start time. . . . : 2011.014 10:28:04.48
Stop time . . . . : 2011.014 10:28:59.24
Return code . . . : 00000000
Reason code . . . : 00000000
Reason text . . . :
Command master. . : IMS1

MbrName  Messages
-------- -----------------------
IMS1     DFS3127I WRITE ERROR OCCURRED ON THE RESTART DATA SET
```

Explanation: This is an example of a failed invocation of the UPDATE POOL command to dynamically reconfigure OSAM subpools using the OSAM123 section of the DFSDFxxx member. A write error occurred on the restart data set (RDS).

Example 8 for UPDATE POOL command

TSO SPOC input:

```
UPDATE POOL TYPE(DBAS) SECTION(VSAM003)
```

TSO SPOC output:

```
Section  MbrName CC CCText                        Stmt   BufSize  PoolId ErrorText
-------- ------- -- -----------                   ------ -------  ------ ---------
VSAM003  IMS1     8 COMMAND COMPLETE FOR SOME
VSAM003  IMS1     8 COMMAND COMPLETE FOR SOME     VSRBF   1024    VSM2   DUP
```

Explanation: This is an example of a successful invocation of the UPDATE POOL command to dynamically reconfigure VSAM subpools using the VSAM003 section of the DFSDFxxx member. IMS received condition code 8 for one of the POOLID statements for VSM2, because a duplicate VSRBF was specified under the same POOLID. The rest of the statements are still processed. Condition code 8 is treated as a warning message in this situation.

Example 9 for UPDATE POOL command

Specification in DFSDFxxx member:
```
<SECTION=OSAM001>
IOBF=(512,100,N,N)
IOBF=(1024,1000,N,Z,OSM1,Y)
IOBF=(2048,5000,Y,Y,OSM2,A)
IOBF=(4096,5000,N,Y,OSM3,N)
IOBF=(32K,32767,N,N,OSM9,N)
```

TSO SPOC input:
```
UPDATE POOL TYPE(DBAS) SECTION(OSAM001) MEMBER(XYZ)
```

TSO SPOC output:
```
Log for . . : UPDATE POOL TYPE(DBAS) SECTION(OSAM001) MEMBER(XYZ)  More:    >

IMSplex . . . . . : PLEX1
Routing . . . . . :
Start time. . . . : 2009.117 10:24:25.06
Stop time . . . . : 2009.117 10:24:25.07
Return code . . . : 02000008
Reason code . . . : 00002004
Reason text . . . : Invalid command keyword or invalid command keyword combination
Command master. . :

MbrName  Messages
-------- -----------------------
IMS1     BPE0003E AN ERROR OCCURRED PARSING PROCLIB MEMBER DFSDFXYZ
IMS1     BPE0003E AT LINE 22, CHARACTER 18
IMS1     BPE0003E FAILING TEXT: "Z,OSM1,Y)          "
IMS1     BPE0003E UNKNOWN KEYWORD VALUE DETECTED
```

Explanation: In this example, the OSAM001 section in the DFSDFXYZ member is parsed on invocation of the UPDATE POOL command. An invalid option of "Z" is specified for the second IOBF statement in the DFSDFXYZ member.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming APIs)

➡ VSAM subpool definition (System Definition)

➡ Specifying VSAM and OSAM subpools (System Definition)

➡ OSAM subpool definition (System Definition)

➡ Adjusting OSAM and VSAM database buffers (Database Administration)

➡ Monitoring VSAM buffers (Database Administration)

Related tasks:

 OSAM buffers (Database Administration)

Related reference:

 Command keywords and their synonyms (Commands)

 DFSDFxxx member of the IMS PROCLIB data set (System Definition)

UPDATE RM command

Use the UPDATE RM command to enable or disable Resource Manager (RM) to use the IMSRSC repository dynamically.

Subsections:
- "Environment"
- "Syntax"
- "Keywords"
- "Usage notes" on page 1300
- "Output fields" on page 1301
- "Return, reason, and completion codes" on page 1301
- "Examples" on page 1304

Environment

The UPDATE RM command is processed by the Common Service Layer (CSL) RM address space.

Syntax

```
►►──┬─UPDATE─┬──RM──TYPE(REPO)──REPOTYPE(IMSRSC)────────────────────►
    └─UPD────┘

►──SET(──┬─REPO(Y)─────────────────────────────────────────┬──)────────►◄
         │        └─AUDITACCESS(──┬─DEFAULT────┬──)─┘       │
         │                        ├─NOAUDIT────┤           │
         │                        ├─READ───────┤           │
         │                        ├─SECURITY───┤           │
         │                        ├─UPDATE─────┤           │
         │                        └─SYSTEMREAD─┘           │
         ├─REPO(N)─────────────────────────────────────────┤
         └─AUDITACCESS(──┬─DEFAULT────┬──)─────────────────┘
                         ├─NOAUDIT────┤
                         ├─READ───────┤
                         ├─SECURITY───┤
                         ├─UPDATE─────┤
                         └─SYSTEMREAD─┘
```
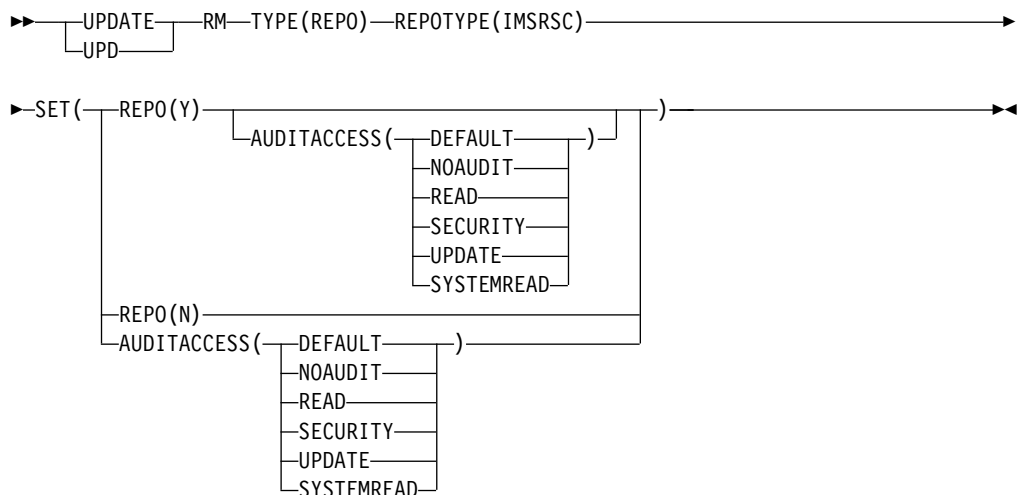
Keywords

The following keywords are valid for the UPDATE RM command:

TYPE

Specifies the resource type to be updated. The only valid value is REPO.

REPOTYPE

Specifies the repository type to be updated. The only valid value is IMSRSC.

SET

Specifies the parameters to be modified.

AUDITACCESS

Specifies the repository audit access level for the specified repository. It can be specified with REPO(Y) to set a different AUDITACCESS value than the default or to change the AUDITACCESS level for the IMSplex.

If the SET(AUDITACCESS()) keyword is specified with REPO(Y) and the Repository Server (RS) does not have AUDIT LOG defined (AUDIT=YES is not specified in the FRPCFG PROCLIB member), the AUDITACCESS(x) value specified is ignored and the AUDITACCESS at RM is set to NOAUDIT. If the SET(AUDITACCESS()) keyword is specified without REPO(Y) and the RS does not have AUDIT LOG defined, the command results in an error reason code.

If the RS is defined with AUDIT LOG (AUDIT=YES), the specified AUDITACCESS value overrides the default audit access level set by the AUDIT_DEFAULT= keyword in the FRPCFG member of the IMS PROCLIB data set. The values that can be specified are:

DEFAULT

Sets the audit access level to the value specified with the AUDIT_DEFAULT parameter in the FRPCFG member.

NOAUDIT

No auditing of member access.

SECURITY

Audit security failures only.

UPDATE

Audit member access with update intent.

READ

Audit member access with read or update intent. Under an audit access rule of READ, system read requests do not cause a read audit record to be generated.

SYSTEMREAD

Audit member access with system-level read, read, or update intent. A read of the resource definition by the system before the update request is identified as a *system read* request. Under an audit access rule of SYSTEMREAD, all read requests, including system read requests, are audited.

REPO

Specifies the repository attribute to be modified.

Y Updates RM to use the repository for the specified repository type.

The RM initialization PROCLIB member (CSLRIxxx), which is used during RM initialization and specified by the RM startup parameter RMINIT=, is to be reread to obtain the REPOSITORY= specifications for the TYPE(IMSRSC) repository.

The REPOSITORY section in the CSLRIxxx member, which is specified on the RM startup parameter RMINIT=, must be updated with any changes before the UPDATE RM command is issued.

The command master RM rereads the REPOSITORY section in the CSLRIxxx member for the TYPE(IMSRSC) repository. During command processing, RM registers to the Repository Server (RS) if RM is not already registered. RM connects to the repository whose name is specified in the REPOSITORY section of the CSLRIxxx member.

If the command is successful at the command master RM, the command master RM communicates the changes to other active RMs in the IMSplex. All RMs in the IMSplex will have the same repository settings.

CSL25*xxx* messages are displayed on the system console at both the command master and the non-master RMs. Any error messages such as CSL2510E or CSL2511E are also sent to the OM API that issued the UPDATE RM command.

If RM is defined to use the resource structure, the command master RM updates the resource structure with the repository name and repository type that it is connected to. Subsequent RMs that are restarted after the change ensure that they are connected to the same repository name and repository type as read from the resource structure.

N Updates RM to not use the repository for the specified repository type. RM disconnects from the repository specified for the repository type.

If the command is successful at the command master RM, the command master RM communicates the changes to other active RMs in the IMSplex. All RMs in the IMSplex will have the same repository settings.

CSL25*xxx* messages are displayed on the system console at both the command master and the non-master RMs. Any error messages such as CSL2510E or CSL2511E are also sent to the OM API that issued the UPDATE RM command.

If RM is defined to use the resource structure, the command master RM updates the resource structure to remove the repository name and repository type. If RM is not connected to any repository, it deregisters from the Repository Server.

The RMINIT parameter or the CSLRIxxx member is not read during the UPDATE RM SET(REPO(N)) processing.

You must modify the CSLRIxxx member to remove the REPOSITORY= statement for the repository type before, or right after, the UPDATE RM SET(REPO(N)) command is issued. If the REPOSITORY= statement is not removed and an RM is started after the UPDATE RM SET(REPO(N)) command is issued, all RMs in the IMSplex will be reconnected to the repository during the RM startup.

Usage notes

The UPDATE RM command can be specified only through the Operations Manager (OM) API.

The UPDATE RM command is defined as ROUTE=ANY to OM. The command is processed by the command master RM that receives the command. The command specifies that the specified section within a PROCLIB member be reread and reprocessed.

Output fields

The following table shows the UPDATE RM output fields. The columns in the table are as follows:

Short label
    Contains the short label generated in the XML output.

Long label
    Contains the long label generated in the XML output.

Keyword
    Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Meaning
    Provides a brief description of the output field.

Table 529. Output fields for UPDATE RM command

Short label	Long label	Keyword	Meaning
CC	CC	N/A	Completion code for the line of output. Completion code is always returned.
CCTXT	CCText	N/A	Completion code text that briefly explains the meaning of the nonzero completion code.
MBR	MbrName	N/A	Resource Manager name.
REPONM	RepositoryName	N/A	Repository name.
REPOTYPE	RepositoryType	N/A	Repository type.

Return, reason, and completion codes

An IMS return and reason code is returned to OM by the UPDATE RM command. The OM return and reason codes that might be returned as a result of the UPDATE RM command are standard for all commands entered through the OM API.

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 530. Return and reason codes for the UPDATE RM command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE RM command completed successfully. The command output contains a line for each resource, accompanied by its completion code. For details, see Table 531 on page 1302.
X'03000004'	X'00001004'	The UPDATE RM command is processed only by the command master; it is ignored by any other system.
X'03000008'	X'0000203C'	An invalid keyword parameter was specified on the UPDATE RM command.

Table 530. Return and reason codes for the UPDATE RM command (continued)

Return code	Reason code	Meaning
X'03000008'	X'00002522'	The RM is unable to process the UPDATE RM command because audit log is not defined for RS, SET(AUDITACCESS(x)) is specified, and no REPO(Y) is specified.
X'0300000C'	X'00003000'	The UPDATE RM command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. For details, see Table 531.
X'0300000C'	X'00003004'	The UPDATE RM command was not successful for any resource. The command output contains a line for each resource, accompanied by its completion code. For details, see Table 531.
X'03000010'	X'00004504'	RM is unable to process the UPDATE RM command because the repository is not defined to RM.
X'03000014'	X'00005030'	Storage for the command response could not be obtained.
X'03000014'	X'00005108'	CQSBRWSE buffer allocation failed.

The following table includes an explanation of the completion code.

Table 531. Completion code for the UPDATE RM command

Completion code	Completion code text	Meaning
0		The UPDATE RM command completed successfully for the resources.
38	CQS UNEXPECTED ERROR	For SET(REPO(Y)), the query call to CQS failed for the query of CSLRPGBL. A CSL2050E message with more details on the CQS error is included in the command output.
60	REPOSITORY ALREADY CONNECTED	For SET(REPO(Y)), the repository is already connected.
61	CONFIGURATION FILE NOT FOUND	For SET(REPO(Y)), the CSLRIxxx configuration file was not found.
62	REPOSITORY SECTION NOT FOUND	For SET(REPO(Y)), there was no REPOSITORY section defined in CSLRIxxx.
63	PARSE ERROR IN CONFIGURATION	For SET(REPO(Y)), the BPE parser returned an error when parsing the REPOSITORY section in CSLRIxxx. A BPE0003E message with more details on the parse error is included in the command output.
64	DUPLICATE REPOSITORY DEFINED	For SET(REPO(Y)), the repository type specified in the command was defined multiple times in CSLRIxxx.
65	INVALID CHARACTERS IN REPO NAME	For SET(REPO(Y)), the repository name specified in the repository definition in CSLRIxxx included invalid characters.

Table 531. Completion code for the UPDATE RM command (continued)

Completion code	Completion code text	Meaning
66	INVALID REPOSITORY NAME	For SET(REPO(Y)), an invalid repository name ("CATALOG") was specified in the repository definition in CSLRIxxx.
67	INVALID CHARACTERS IN GROUP NAME	For SET(REPO(Y)), the XCF group name specified in the repository definition in CSLRIxxx included invalid characters.
68	REPOSITORY DEFINITION NOT FOUND	For SET(REPO(Y)), the repository type specified in the command is not found in CSLRIxxx or CSLRPGBL.
69	GROUP NAME MISMATCH IN REPO DEFN	For SET(REPO(Y)), the XCF group name specified in the repository definition in CSLRIxxx did not match the XCF group name in CSLRPGBL.
6A	REGISTER TO REPO SERVER FAILED	For SET(REPO(Y)), RM attempted to register to the repository server, but the registration attempt failed. A CSL2510E message with more details on the registration failure is included in the command output.
6B	CONNECT TO REPOSITORY FAILED	For SET(REPO(Y)), RM attempted to connect to the repository, but the attempt failed. A CSL2511E message with more details on the connection failure is included in the command output.
6C	UPDATE OF CSLRPGBL FAILED	For SET(REPO(Y)), RM successfully connected to the repository, but the attempt to update the CSLRPGBL resource in the resource structure failed. The operator must determine the status of CQS and the resource structure and reenter the command to complete the connection process.
6D	REPOSITORY NOT CONNECTED	For SET(REPO(N)), RM is not connected to the repository. Either the repository has never been connected, or it has been previously disconnected.
6E	DISCONN FROM REPOSITORY FAILED	For SET(REPO(N)), RM attempted to disconnect from the repository, but the attempt failed. A CSL2511E message with more details on the disconnect failure is included in the command output.
6F	DEREGISTER REPO SERVER FAILED	For SET(REPO(N)), RM attempted to deregister from the Repository Server, but the deregistration attempt failed. A CSL2510E message with more details on the deregistration failure is included in the command output.

Table 531. Completion code for the UPDATE RM command (continued)

Completion code	Completion code text	Meaning
70	REMOVE FROM CSLRPGBL FAILED	For SET(REPO(N)), RM successfully disconnected from the repository, but the attempt to update the CSLRPGBL resource in the resource structure failed. The operator must determine the status of CQS and the resource structure and reenter the command to complete the disconnect process.
71	READ OF PROCLIB MEMBER FAILED	BPERDPDS failed to read the CSLRIxxx PROCLIB member.
72	UNABLE TO GET PSAN LATCH	The command processor was unable to obtain the PSAN latch.
73	UNABLE TO GET PCAN LATCH	The command processor was unable to obtain the PCAN latch.
74	UNABLE TO LOAD MODULES	BPELOAD failed to load modules needed for the specified repository.
75	AUDIT ACCESS UPDATE FAILED	For SET(AUDITACCESS()), RM attempted to update the audit access value in the repository, but the update attempt failed.
76	REFRESH OF CSLRPGBL FAILED	For SET(AUDITACCESS()), RM successfully updated the audit access value in the repository, but the attempt to refresh the CSLRPGBL resource in the resource structure failed. The operator must determine the status of CQS and the resource structure, and reenter the command to complete the refresh process.
77	DOWNLEVEL RM IN IMSPLEX	For SET(REPO(Y)), RM detected an IMS Version 11 or earlier RM in the IMSplex, and is unable to connect to the repository. All RM systems in the IMSplex must be IMS Version 12 or later systems.
78	NO SCI ADDRESS SPACE	For SET(REPO(Y)), RM attempted to query SCI to obtain IMSplex status, but SCI was not available. A CSL2050E message with more details on the SCI error is included in the command output.
79	UNEXPECTED SCI ERROR	For SET(REPO(Y)), RM attempted to query SCI to obtain IMSplex status, but the query failed with an unexpected return code. A CSL2050E message with more details on the SCI error is included in the command output.

Examples

The following are examples of the UPDATE RM command:

Example 1 for UPDATE RM command

TSO SPOC input:

```
UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(Y))
```

TSO SPOC output:

```
RepositoryType MbrName    CC RepositoryName
IMSRSC         RM3RM       0 IMSRSC_REPOSITORY
```

OM API input:

```
CMD(UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(Y)) )
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2011.187 16:55:47.880310</statime>
<stotime>2011.187 16:55:48.214072</stotime>
<staseq>C8079C197D976B1A</staseq>
<stoseq>C8079C19CF1383C8</stoseq>
<rqsttkn1>USRT005 10095547</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmderr>
<mbr name="RM2RM   ">
<typ>RM      </typ>
<styp>MULTRM  </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
<rsntxt>Command not routed to this command processing client.  Client
 is not master.</rsntxt>
</mbr>
<mbr name="RM1RM   ">
<typ>RM      </typ>
<styp>MULTRM  </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
<rsntxt>Command not routed to this command processing client.  Client
 is not master.</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>RM3RM   </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>RM            </kwd>
<input>UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(Y)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="REPOTYP" llbl="RepositoryType" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPONM" llbl="RepositoryName" scope="LCL" sort="n" key="0"
 scroll="yes" len="44" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>REPOTYP(IMSRSC  ) MBR(RM3RM) CC(   0) REPONM(IMSRSC_REPOSITORY
                     ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE RM SET(REPO(Y)) command for the repository is processed by the command master RM RM3. The command response indicates that the repository is enabled successfully at the command master RM. Additionally, because RM is using the resource structure, the command response indicates that the resource structure is updated to indicate that the repository is enabled in the IMSplex and that a directive was sent to RM1 and RM2, the other RMs in the IMSplex, to have them enable the repository. You can issue the QUERY RM command to ensure that the repository is enabled at all RMs in the IMSplex.

Example 2 for UPDATE RM command

TSO SPOC input:
```
UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(AUDITACCESS(UPDATE))
```

TSO SPOC output:
```
RepositoryType MbrName    CC RepositoryName
IMSRSC         RM3RM       0 IMSRSC_REPOSITORY
```

OM API input:
```
CMD(UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(AUDITACCESS(UPDATE)) )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM    </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.187 17:01:05.705662</statime>
<stotime>2011.187 17:01:05.773423</stotime>
<staseq>C8079D4897ABED80</staseq>
<stoseq>C8079D48A836F750</stoseq>
<rqsttkn1>USRT005 10100105</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmderr>
<mbr name="RM2RM    ">
<typ>RM      </typ>
<styp>MULTRM  </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
<rsntxt>Command not routed to this command processing client.  Client
 is not master.</rsntxt>
</mbr>
<mbr name="RM1RM    ">
<typ>RM      </typ>
<styp>MULTRM  </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
<rsntxt>Command not routed to this command processing client.  Client
 is not master.</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>RM3RM    </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>RM              </kwd>
<input>UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(AUDITACCESS(UPDATE))
</input>
</cmd>
<cmdrsphdr>
```

```
<hdr slbl="REPOTYP" llbl="RepositoryType" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPONM" llbl="RepositoryName" scope="LCL" sort="n" key="0"
 scroll="yes" len="44" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>REPOTYP(IMSRSC  ) MBR(RM3RM) CC(    0) REPONM(IMSRSC_REPOSITORY
                     ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE RM SET(AUDITACCESS()) command is processed by the command master RM RM3. The command response indicates that the auditaccess value was updated in the repository to be used for subsequent audit requests. Additionally, because RM is using the resource structure, the command response indicates that the audit access value was updated in the resource structure and that a directive was sent by RM3 to RM1 and RM2, the other active RMs in the IMSplex, to update their audit access values.

Example 3 for UPDATE RM command

TSO SPOC input:
```
UPD RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(N))
```

TSO SPOC output:
```
RepositoryType MbrName    CC RepositoryName
IMSRSC         RM3RM        0 IMSRSC_REPOSITORY
```

OM API input:
```
CMD(UPD RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(N)) )
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.5.0</omvsn>
<xmlvsn>20   </xmlvsn>
<statime>2011.187 16:36:18.444602</statime>
<stotime>2011.187 16:36:18.542958</stotime>
<staseq>C80797BE3AD3A85E</staseq>
<stoseq>C80797BE52D6ED15</stoseq>
<rqsttkn1>USRT005 10093618</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmderr>
<mbr name="RM2RM   ">
<typ>RM      </typ>
<styp>MULTRM  </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
<rsntxt>Command not routed to this command processing client.  Client
 is not master.</rsntxt>
</mbr>
<mbr name="RM1RM   ">
<typ>RM      </typ>
```

```
<styp>MULTRM  </styp>
<rc>02000004</rc>
<rsn>00001008</rsn>
<rsntxt>Command not routed to this command processing client.  Client
 is not master.</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>RM3RM   </master>
<userid>USRT005 </userid>
<verb>UPD </verb>
<kwd>RM              </kwd>
<input>UPD RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(N)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="REPOTYP" llbl="RepositoryType" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="REPONM" llbl="RepositoryName" scope="LCL" sort="n" key="0"
 scroll="yes" len="44" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>REPOTYP(IMSRSC  ) MBR(RM3RM) CC(    0) REPONM(IMSRSC_REPOSITORY
                         ) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE RM SET(REPO(N)) command for the repository is
processed by the command master RM RM3. The command response indicates that
the repository is no longer enabled at the command master RM. Additionally,
because RM is using the resource structure, the command response indicates that
the resource structure is updated to indicate that the repository is not enabled in
the IMSplex and that a directive was sent to the other RMs to have them disable
their usage of the repository.

Related concepts:

➤ How to interpret CSL request return and reason codes (System Programming
APIs)

➤ CSL RM initialization with the IMSRSC repository (System Administration)

➤ CSL RM management of the IMSRSC repository (System Administration)

Related reference:

➤ Command keywords and their synonyms (Commands)

UPDATE RTC command

Use the UPDATE RTC command to update Fast Path routing codes. A Fast Path
routing code can be used by the Fast Path Input Edit/Routing Exit Routine
(DBFHAGU0) to route a transaction to a different application program for
processing.

Subsections:
- "Environment" on page 1309
- "Syntax" on page 1309

- "Keywords"
- "Usage notes" on page 1310
- "Output fields" on page 1311
- "Return, reason, and completion codes" on page 1312
- "Examples" on page 1313

Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 532. Valid environments for the UPDATE RTC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE RTC	X		X
NAME	X		X
OPTION	X		X
SET	X		X
START	X		X
STOP	X		X

Syntax

Keywords

The following keywords are valid for the UPDATE RTC command:

NAME
Specifies the 1-8 character name of the routing code. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the resource names that are processed. For NAME(*), command responses are returned only for the resource names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the resource names that are processed.

OPTION
Specifies additional functions to be performed along with the command.

ALLRSP

Indicates that the response lines are to be returned for all resources that are processed on the command. The default action is to return response lines only for the resources that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SET

Specifies the attribute values to be changed.

INQ

Specifies the inquiry option.

N This is not an inquiry routing code.

Y This is an inquiry routing code. Any message associated with the routing code is an inquiry transaction. This option should be specified only for programs that do not cause a change to a database. Programs are prohibited from issuing Insert, Delete, or Replace calls to a database when processing an inquiry transaction.

Updating this attribute quiesces the routing code for the duration of command processing.

PGM

The name of the application program associated with the routing code. The program must exist and be defined with a BMPTYPE of N.

Updating this attribute quiesces the routing code for the duration of command processing.

START

Specifies attributes that are to be started.

Q Starts queuing of messages.

STOP

Specifies attributes that are to be stopped.

Q Stops queuing of messages.

Usage notes

Resources exist for the life of the IMS unless they are deleted using a DELETE command. Resources are recoverable across an IMS warm start or emergency restart. Resources are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE RTC command can be issued only through the OM API. Fast Path must be installed on the system. This command applies to DB/DC and DCCTL systems.

This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE RTC commands specified with SET() are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

This command is recoverable.

The UPDATE RTC command changes a MODBLKS routing code to dynamic, if the INQ or PGM attribute is changed.

Each routing code is updated individually, unlike the online change process where either all routing codes are updated or no routing codes are updated. The runtime resource definition for a routing code can be updated only if the routing code is not in use. If the routing code is in use, the update fails. An exception to this rule is status. You can update the status of a routing code while it is in use. In a sysplex environment, the update might succeed on some IMS systems and fail on others. In order to maximize the likelihood that the update will succeed, perform the following steps before attempting the update:

- Check for work in progress with a QRY RTC SHOW(WORK) command and either wait for the work to finish or address the work in progress. See the output fields for the QUERY RTC command for examples of work that might cause the delete to fail. Examples of work include a command in progress for the routing code or the routing code is active.
- The routing code cannot be updated if the program is scheduled. If the program is scheduled, you must stop the region before you issue the UPDATE RTC command.

Runtime resource definition attributes include the following: INQ and PGM. If all the attributes specified by the UPDATE command are already defined for the resource, no update is actually made, no resources are quiesced, no log record is created, and a completion code of zero is returned.

The following routing code attributes cannot be updated if online change for MODBLKS is enabled: INQ, PGM.

Output fields

The following table shows the UPDATE RTC output fields. The columns in the table are as follows:

Short label
        Contains the short label generated in the XML output.

Keyword
        Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
        Provides a brief description of the output field.

Table 533. Output fields for the UPDATE RTC command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the non-zero completion code.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a non-zero completion code and contains information that further explains the completion code.
MBR	N/A	IMSplex member that built the output line.
RTC	RTC	Routing code name.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 534. Return and reason codes for the UPDATE RTC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000008'	X'00002008'	Required keywords were not specified.
X'00000008'	X'00002123'	Invalid program name.
X'00000008'	X'00002048'	Invalid SET attribute.
X'0000000C'	X'00003000'	Command was successful for some resources but failed for others. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), only resources with non-zero completion codes are returned. See the completion code table for details.
X'00000010'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'0000431C'	Program is quiesced. Cannot quiesce program.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 535. Completion codes for the UPDATE RTC command

Completion code	Completion code text	Meaning
0		Command completed successfully for routing code.
10	NO RESOURCES FOUND	Routing code name is invalid, or the wildcard parameter specified does not match any resource names.

Table 535. Completion codes for the UPDATE RTC command (continued)

Completion code	Completion code text	Meaning
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this routing code. This could also mean this command, if the resource is specified by more than one specific or wildcard parameter. Or, the routing code is updating the program name and another command is in progress for the program.
39	FP=E TRAN FOR RTC EXISTS	The routing code was created by IMS for a Fast Path exclusive transaction and cannot be updated with the UPDATE RTC command. The attributes of the routing code can be updated only with the UPDATE TRAN command for the Fast Path exclusive transaction.
61	DFSBCB STORAGE ERROR	DFSBCB storage error. Could not get storage for RCTE control block.
73	PROGRAM SCHEDULED	Program is scheduled.
7A	RTC/FP=N PGM CONFLICT	Program specified is not Fast Path exclusive.
90	INTERNAL ERROR	Internal error.
99	NOT INITIALIZED	Routing code update failed because the routing code is not initialized. QUERY RTC STATUS(NOTINIT) displays the reason why the routing code is not initialized, for example, the associated program is not defined. Correct the definition error and issue the UPD RTC START(Q) command to initialize the routing code.
1D7	IMPORT CHANGE LIST ERROR	The resource is in the process of being imported from the change list in the IMSRSC repository or was not successfully imported from the change list. The resource cannot be updated until it is successfully imported from the repository.

Examples

The following are examples of the UPDATE RTC command:

Example 1 for UPDATE RTC command

TSO SPOC input:
```
UPD  RTC NAME(BADNAME,FPTRN02,SMQFP7*,BAD*) SET(PGM(DBFSAMP3))
```

TSO SPOC output:

```
Response for: UPD  RTC NAME(BADNAME,FPTRN02,SMQFP7*,BAD*) SET(PGM(DBFSAMP3))
RtcName  MbrName   CC CCText
BAD*     IMS1      10 NO RESOURCES FOUND
BADNAME  IMS1      10 NO RESOURCES FOUND
FPTRN02  IMS1       0
SMQFP7A  IMS1       0
SMQFP7B  IMS1       0
SMQFP7C  IMS1       0
SMQFP71  IMS1       0
SMQFP72  IMS1       0
SMQFP73  IMS1       0
```

OM API input:

```
CMD(UPDATE RTC NAME(BADNAME,FPTRN02,SMQFP7*,BAD*) SET(PGM(DBFSAMP3)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.311 23:50:02.663055</statime>
<stotime>2006.311 23:50:02.663790</stotime>
<staseq>BFAC125257C8FB80</staseq>
<stoseq>BFAC125257F6E500</stoseq>
<rqsttkn1>USRT011 10155002</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS    </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>RTC            </kwd>
<input>UPD  RTC NAME(BADNAME,FPTRN02,SMQFP7*,BAD*) SET(PGM(DBFSAMP3))
</input>
</cmd>
<cmdrsphdr>
<hdr slbl="RTC" llbl="RtcName" scope="LCL" sort="a" key="1" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>RTC(BADNAME ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>RTC(FPTRN02 ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(SMQFP7A ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(BAD*    ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>RTC(SMQFP7B ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(SMQFP7C ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(SMQFP71 ) MBR(IMS1) CC(   0) </rsp>
```

```
<rsp>RTC(SMQFP72 ) MBR(IMS1) CC(   0) </rsp>
<rsp>RTC(SMQFP73 ) MBR(IMS1) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE RTC command is issued to update the program for several routing codes. The update completed successfully for most of the routing codes, as shown by completion code 0. The update fails for routing code BADNAME and for parameter BAD* with completion code 10 because routing code BADNAME does not exist and no routing code name starts with BAD.

Related concepts:

➡ How to interpret CSL request return and reason codes (System Programming APIs)

Related reference:

➡ Command keywords and their synonyms (Commands)

"/START RTC command" on page 862

"/STOP RTC command" on page 915

UPDATE RTCDESC command

Use the UPDATE RTCDESC command to update Fast Path routing code descriptors.

A descriptor is a model that can be used to create descriptors or resources. Updating a descriptor changes only the attributes explicitly specified on the UPDATE command. Attributes not specified retain their existing values. Any routing code resource or descriptor can be created using this descriptor as a model, by specifying the CREATE LIKE(DESC(*descriptor_name*)) command. Any descriptor or resource that was already created using this descriptor is not updated.

Subsections:
- "Environment"
- "Syntax" on page 1316
- "Keywords" on page 1316
- "Usage notes" on page 1317
- "Output fields" on page 1317
- "Return, reason, and completion codes" on page 1318
- "Examples" on page 1320

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 536. Valid environments for the UPDATE RTCDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE RTCDESC	X		X
NAME	X		X
OPTION	X		X
SET	X		X

Syntax

```
            ┌─────────┐                        ┌──────────────────────┐
►►──┬─UPDATE─┬──RTCDESC──NAME(─▼─name─┴─)──SET(─▼─┬─DEFAULT(Y)──────┬─┬─)──►
    └─UPD────┘                                    ├─INQ(─┬─N─┬─)────┤
                                                  │      └─Y─┘      │
                                                  └─PGM(name)───────┘

►──┬──────────────────┬──►◄
   └─OPTION(ALLRSP)────┘
```

Keywords

The following keywords are valid for the UPDATE RTCDESC command:

NAME

Specifies the 1-8 character name of the routing code descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the descriptor names that are processed. For NAME(*), command responses are returned only for the descriptor names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the descriptor names that are processed.

OPTION

Specifies additional functions to be performed along with the command.

ALLRSP

Indicates that the response lines are to be returned for all descriptors that are processed on the command. The default action is to return response lines only for the descriptors that resulted in an error. It is valid with NAME(*) only. ALLRSP is ignored for other NAME values.

SET

Specifies the attribute values to be changed.

DEFAULT(Y)

Specifies this descriptor as the default, which resets the existing default descriptor to DEFAULT(N). When a descriptor or resource is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines a routing code descriptor called DBFDSRT1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor. Because only one routing code descriptor can be the default at one time, only one routing code descriptor name can be specified with DEFAULT(Y).

INQ

Specifies the inquiry option.

N This is not an inquiry routing code.

Y This is an inquiry routing code. Any message associated with the routing code is an inquiry transaction. This should be specified only for programs that do not cause a change to a database. Programs are

prohibited from issuing Insert, Delete, or Replace calls to a database when processing an inquiry transaction.

PGM

> The name of the application program associated with the routing code. The program must exist and be defined with a BMPTYPE of N.

Usage notes

If all the attributes specified by the UPDATE command are already defined for the descriptor, no update is actually made, no descriptors are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

Descriptors exist for the life of the IMS unless they are deleted using a DELETE command. The descriptors are recoverable across an IMS warm start or emergency restart. Descriptors are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE RTCDESC command can be issued only through the OM API. Fast Path must be installed on the system. This command applies to DB/DC and DCCTL systems.

This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE RTCDESC commands are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

These commands are recoverable.

If the descriptor is the IMS-defined routing code descriptor (DBFDSRT1), the only attribute that can be updated is the DEFAULT attribute.

Each descriptor is updated individually. Individual updating does not work like online change where either all descriptors are updated or no descriptors are updated.

Output fields

The following table shows the UPDATE RTCDESC output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
> Provides a brief description of the output field.

Table 537. Output fields for the UPDATE RTCDESC command

Short label	Keyword	Meaning
CC	N/A	Completion code.

Table 537. Output fields for the UPDATE RTCDESC command (continued)

Short label	Keyword	Meaning
CCTXT	*error*	Completion code text that briefly explains the nonzero completion code.
DESC	RTCDESC	Routing code descriptor name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
MBR	N/A	IMSplex member that built the output line.
OLDDEF	RTCDESC	Old default descriptor name, if this descriptor is updated to be the default by specifying DEFAULT(Y). The old default descriptor is no longer the default.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 538. Return and reason codes for the UPDATE RTCDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000008'	X'00002008'	Required keywords were not specified.
X'00000008'	X'00002123'	Invalid program name.
X'00000008'	X'00002048'	Invalid SET attribute.
X'00000008'	X'00002133'	Multiple name parameters were specified with DEFAULT(Y). Only one descriptor can be the default at one time.
X'0000000C'	X'00003000'	Command was successful for some descriptors but failed for others. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), only descriptors with nonzero completion codes are returned. See the completion code table for details.
X'00000010'	X'00003004'	Command was not successful for any of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined.
X'00000010'	X'00004120'	Online change phase is in progress.

Table 538. Return and reason codes for the UPDATE RTCDESC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'0000431C'	Program is quiesced. Cannot quiesce program.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 539. Completion codes for the UPDATE RTCDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for routing code descriptor.
10	NO RESOURCES FOUND	Routing code descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this routing code descriptor. This could also mean this command, if the resource is specified by more than one specific or wildcard parameter.
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for the IMS descriptor or resource. DBFDSRT1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
61	DFSBCB STORAGE ERROR	DFSBCB storage error. Could not get storage for RCTE control block.
7A	RTC/FP=N PGM CONFLICT	Program specified is not Fast Path exclusive.
8A	WILDCARD PARAMETER NOT SUPPORTED	A wildcard parameter was specified with DEFAULT(Y). Only one descriptor can be the default at one time.
90	INTERNAL ERROR	Internal error.
1D7	IMPORT CHANGE LIST ERROR	The descriptor is in the process of being imported from the change list in the IMSRSC repository or was not successfully imported from the change list. The descriptor cannot be updated until it is successfully imported from the repository.

Examples

The following are examples of the UPDATE RTCDESC command:

Example 1 for UPDATE RTCDESC command

TSO SPOC input:
```
UPD RTCDESC NAME(*) SET(PGM(BMPFPE01)) OPTION(ALLRSP)
```

TSO SPOC output:
```
Response for: UPD RTCDESC NAME(*) SET(PGM(BMPFPE01)) OPTION(ALLRSP)
DescName MbrName   CC CCText
DBFDSRT1 IMS1      7A RTC/FP=N PGM CONFLICT
DESC001  IMS1      7A RTC/FP=N PGM CONFLICT
DESC002  IMS1      7A RTC/FP=N PGM CONFLICT
DESC003  IMS1      7A RTC/FP=N PGM CONFLICT
DESC004  IMS1      7A RTC/FP=N PGM CONFLICT
DESC005  IMS1      7A RTC/FP=N PGM CONFLICT
RTCDESC1 IMS1      7A RTC/FP=N PGM CONFLICT
```

OM API input:
```
CMD(UPDATE RTCDESC NAME(*) SET(PGM(BMPFPE01)) OPTION(ALLRSP))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.312 00:04:22.046943</statime>
<stotime>2006.312 00:04:22.047492</stotime>
<staseq>BFAC1585EA4DF64A</staseq>
<stoseq>BFAC1585EA70488A</stoseq>
<rqsttkn1>USRT011 10160422</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003004</rsn>
<rsntxt>No requests were successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>RTCDESC         </kwd>
<input>UPD RTCDESC NAME(*) SET(PGM(BMPFPE01)) OPTION(ALLRSP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
```

```
 scroll="yes" len="32" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DESC004 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP=N PGM CONFLICT)
</rsp>
<rsp>DESC(DESC005 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP=N PGM CONFLICT)
</rsp>
<rsp>DESC(DESC001 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP=N PGM CONFLICT)
</rsp>
<rsp>DESC(RTCDESC1) MBR(IMS1) CC(  7A) CCTXT(RTC/FP=N PGM CONFLICT)
</rsp>
<rsp>DESC(DBFDSRT1) MBR(IMS1) CC(  7A) CCTXT(RTC/FP=N PGM CONFLICT)
</rsp>
<rsp>DESC(DESC002 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP=N PGM CONFLICT)
</rsp>
<rsp>DESC(DESC003 ) MBR(IMS1) CC(  7A) CCTXT(RTC/FP=N PGM CONFLICT)
</rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE RTCDESC command is issued to update all routing
code descriptors to reference program BMPFPE01. The update fails for all routing
codes with completion code 7A because a routing code cannot be associated with a
non-Fast Path program.

Related concepts:

 How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

 Command keywords and their synonyms (Commands)

UPDATE TRACE command

Use the UPDATE TRACE command to change the settings for the type-2 traces.

Subsections:
- "Environment"
- "Syntax" on page 1322
- "Keywords" on page 1322
- "Usage notes" on page 1323
- "Output fields" on page 1323
- "Return, reason, and completion codes" on page 1324
- "Example" on page 1325

Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which
you can use the UPDATE TRACE command and keywords.

Table 540. Valid environments for the UPDATE TRACE *command and keywords*

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE TRACE	X	X	X
LEVEL	X	X	X
NAME	X	X	X
TYPE	X	X	X
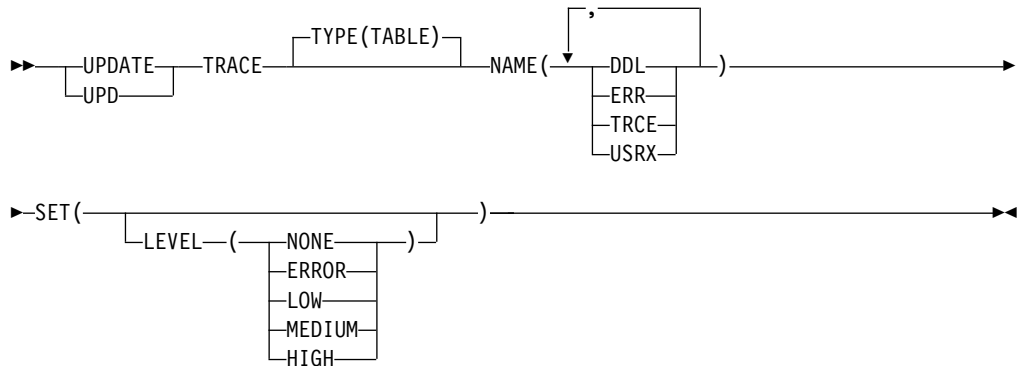
The following keywords are valid for the UPDATE TRACE command:

NAME

Specifies the 1-4 character name of the type-2 trace to update. You can specify a single trace name or a comma-separated list of trace names.

The following values are valid for the NAME keyword:

DDL   Data definition language (DDL) trace table.

ERR   Error trace table.

TRCE   Trace services trace table.

USRX   User exit services trace table.

SET

Specifies the attribute values to be changed.

LEVEL

An optional parameter that sets the new tracing level for the specified trace. If LEVEL is omitted, the level of the specified trace is not changed.

Important: You cannot change the level for the ERR trace type. IMS sets the level to HIGH for ERR traces to ensure that error diagnostics are captured. Any level that you specify for the ERR trace is ignored.

If the trace level for the trace is originally NONE and is then updated to any other level, the trace storage is allocated if needed and the trace is turned on.

If the IMS system is restarted, any changes that were made to the trace storage by using the UPDATE TRACE command are not recovered. Instead, the parameters that are defined in the DIAGNOSTICS_STATISTICS section of the DFSDFxxx member of the IMS proclib data set are used.

You can specify the following levels:

NONE
> The trace is not active. If the trace is active when this command is issued, it is turned off, but the trace storage is not deallocated.

ERROR
> Only traces for error or exception conditions are written into the table.

LOW
> Only major event trace entries are written into the table.

MEDIUM
> Major event trace entries and some minor event trace entries are written into the table.

HIGH
> All trace entries are written into the table.

TYPE
Specifies the trace type.

> **TABLE**
> Specifies a type-2 table trace. This is the default value.

Usage notes

This command can be issued only through the Operations Manager API. This command applies to DB/DC, DBCTL and DCCTL systems.

The command output for the UPDATE TRACE command is defined in XML and is available to automation programs that communicate with OM. The output contains an entry for each trace that is named in the UPDATE TRACE command. The command is routed to all IMS's in the IMSplex as its default routing.

Output fields

The following table shows the UPDATE TRACE output fields. The columns in the table are as follows:

Short label
> Contains the short label generated in the XML output.

Long label
> Contains the long label generated in the XML output.

Keyword
> Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned.

Scope Identifies the scope of the output field.

Meaning
> Provides a brief description of the output field.

Table 541. Output fields for the UPDATE TRACE command

Short label	Long label	Keyword	Scope	Meaning
CC	CC	N/A	N/A	Completion code for the line of output. The completion code is always returned.
CCTXT	CCText	*error*	N/A	Completion code text that briefly explains the nonzero completion code.

Table 541. Output fields for the UPDATE TRACE command (continued)

Short label	Long label	Keyword	Scope	Meaning
MBR	MbrName	N/A	N/A	IMSplex member that built the output line. Member name is always returned.
NAME	TraceName	NAME	LCL	Trace name.
TYPE	TraceType	TYPE	LCL	Type of the trace.

Return, reason, and completion codes

The return and reason codes that can be returned as a result of the UPDATE TRACE command are standard for all commands that are entered through the OM API.

The following table contains the return code, reason code, and an explanation of the codes for the UPDATE TRACE command.

Table 542. Return and reason codes for the UPDATE TRACE command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	The UPDATE TRACE command completed successfully.
X'00000008'	X'00002040'	The UPDATE TRACE command failed. No filter, an invalid filter, or an insufficient number of filters specified.
X'0000000C'	X'00003000'	The UPDATE TRACE command was successful for at least one trace. The UPDATE TRACE command failed for one or more traces. The completion code indicates the reason for the error. The completion codes that can be returned by the UPDATE TRACE command are listed in the UPDATE TRACE completion code table.
X'0000000C'	X'00003004'	The UPDATE TRACE command failed for all traces specified. The completion code indicates the reason for the error. The completion codes that can be returned by the UPDATE TRACE command are listed in the UPDATE TRACE completion code table.
X'00000014'	X'00005004'	The UPDATE TRACE command processing terminated because a DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005FFF'	The UPDATE TRACE command processing terminated due to an internal error.

Errors unique to the processing of this command are returned as a completion code. A completion code is returned for each action against an individual resource.

Table 543. Completion codes for the UPDATE TRACE command

Completion code	Meaning
0	The UPDATE TRACE command completed successfully for the trace.
10	No resources found. In this case, the resource is the trace name. None of the traces specified are known to the IMS system that processed the command. The trace might have been typed in error. Confirm that the correct spelling of the trace is specified on the command.

Table 543. Completion codes for the UPDATE TRACE command (continued)

Completion code	Meaning
5F	Invalid character is used in the trace name. Remove all invalid characters in the specified name or names.
8A	Wildcards are not supported. A wildcard was specified in a **NAME** parameter. Remove all wildcard characters, * and %, from the specified trace name or names.
250	Trace allocation error. The trace could not be allocated.

Example

TSO SPOC input:

```
UPD TRACE TYPE(TABLE) NAME(TRCE,ERR) SET(LEVEL(LOW))
```

TSO SPOC output:

```
Response for: UPD TRACE TYPE(TABLE) NAME(TRCE,ERR) SET(LEVEL(LOW))
TraceName MbrName   CC TraceType
ERR       IMS1       0 Table
ERR       IMS2       0 Table
TRCE      IMS1       0 Table
TRCE      IMS2       0 Table
```

OM API input:

```
CMD(UPD TRACE TYPE(TABLE) NAME(TRCE,ERR) SET(LEVEL(LOW)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.7.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2017.180 20:30:29.245120</statime>
<stotime>2017.180 20:30:29.258457</stotime>
<staseq>D2C2F74A440C002B</staseq>
<stoseq>D2C2F74A474D932B</stoseq>
<rqsttkn1>USRT011 10133029</rqsttkn1>
<rc>00000000</rc>
<rsn>00000000</rsn>
</ctl>
<cmd>
<master>IMS2    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>TRACE         </kwd>
<input>UPD TRACE TYPE(TABLE) NAME(TRCE,ERR) SET(LEVEL(LOW)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="NAME" llbl="TraceName" scope="LCL" sort="a" key="1"
 scroll="no" len="4" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2"
 scroll="no" len="8" dtype="CHAR" align="left" skipb="no" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" align="left" skipb="yes" />
<hdr slbl="TYPE" llbl="TraceType" scope="LCL" sort="n" key="0"
 len="8" dtype="CHAR" align="left" skipb="yes" />
</cmdrsphdr>
<cmdrspdata>
<rsp>NAME(ERR ) MBR(IMS2) CC(   0) TYPE(Table   ) </rsp>
```

```
|                    <rsp>NAME(TRCE) MBR(IMS2) CC(   0) TYPE(Table   ) </rsp>
|                    <rsp>NAME(ERR ) MBR(IMS1) CC(   0) TYPE(Table   ) </rsp>
|                    <rsp>NAME(TRCE) MBR(IMS1) CC(   0) TYPE(Table   ) </rsp>
|                  </cmdrspdata>
|                </imsout>
```

| **Explanation:**
| The UPDATE TRACE command was routed to IMS1 and IMS2. IMS1 and IMS2
| both processed the command and returned successful completion codes for the
| ERR table trace and the TRCE table trace. Note that the level of the ERR table trace
| is not changed because the level of this table is always set to HIGH, and the
| command is ignored for the ERR table trace.

UPDATE TRAN command

Use the UPDATE TRAN command to update transaction resources.

Subsections:
- "Environment"
- "Syntax"
- "Keywords" on page 1329
- "Usage notes" on page 1348
- "Equivalent IMS type-1 commands" on page 1349
- "Output fields" on page 1350
- "Return, reason, and completion codes" on page 1351
- "Examples" on page 1358

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from
which the UPDATE TRAN command and keywords can be issued.

Table 544. Valid environments for the UPDATE TRAN command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE TRAN	X		X
CLASS	X		X
NAME	X		X
OPTION	X		X
SCOPE	X		X
SET	X		X
START	X		X
STOP	X		X

Syntax

```
             ┌─────────,──────────────────┐        ┌ A ┐
►─┬──────────────────────────────────────────┬──┤ B ├──────────────────►
  └─CLASS(─▼──current_class_number──┬──)─┘     ├ A ─┤ B ├┤
                                               └ C ┘


                                      ┌─SCOPE(ALL)─────┐
►─┬────────────────────────────────┬──┼────────────────┼──────────────►◄
  │          ┌──────,──────┐        │  └─SCOPE(ACTIVE)─┘
  └─OPTION──(─▼──┬─AFFIN─┬──┬─,)─┘
                 └─ALLRSP─┘
```

A:

```
              ┌──────,──────┐
├─START(─▼──┬─Q───────┬──)───────────────────────┤
            ├─SCHD────┤
            ├─SUSPEND─┤
            └─TRACE───┘
```

B:

```
              ┌──────,──────┐
├─STOP(─▼──┬─Q─────┬──)────────────────────────┤
           ├─SCHD──┤
           └─TRACE─┘
```

C:

```
          ┌─────────────────,◄──────────────────────┐
├──SET(──▼──AOCMD(──┬──CMD──┬──)──────────────────────┐──)─────────────────────┤
│                   ├──N────┤                         │
│                   ├──TRAN─┤                         │
│                   └──Y────┘                         │
│         ──CLASS(class)───────────────────           │
│         ──CMTMODE(──┬──MULT──┬──)───────             │
│                     └──SNGL──┘                       │
│         ──CONV(──┬──N──┬──)──────────────            │
│                  └──Y──┘                             │
│         ──CPRI(value)────────────────────           │
│         ──DCLWA(──┬──N──┬──)─────────────            │
│                   └──Y──┘                            │
│         ──DIRROUTE(──┬──N──┬──)──────────            │
│                      └──Y──┘                         │
│         ──EDITRTN(──┬──name──┬──)────────            │
│                     └──0─────┘                       │
│         ──EDITUC(──┬──N──┬──)────────────            │
│                    └──Y──┘                           │
│         ──EMHBSZ(size)───────────────────           │
│         ──EXPRTIME(seconds)──────────────           │
│         ──FP(──┬──E──┬──)────────────────           │
│                ├──N──┤                               │
│                └──P──┘                               │
│         ──INQ(──┬──N──┬──)───────────────           │
│                 └──Y──┘                              │
│         ──LCT(value)─────────────────────           │
│         ──LOCK(──┬──ON───┬──)────────────           │
│                  └──OFF──┘                           │
│         ──LPRI(value)────────────────────           │
│         ──MAXRGN(number)─────────────────           │
│         ──MSGTYPE(──┬──MULTSEG──┬──)─────            │
│                     └──SNGLSEG──┘                    │
│         ──MSNAME(name)───────────────────           │
│         ──NPRI(value)────────────────────           │
│         ──PARLIM(value)──────────────────           │
│         ──PGM(name)──────────────────────           │
│         ──PLCT(value)────────────────────           │
│         ──PLCTTIME(hundredths of seconds)─           │
│         ──RECOVER(──┬──N──┬──)───────────            │
│                     └──Y──┘                          │
│         ──REMOTE(──┬──N──┬──)────────────            │
│                    └──Y──┘                           │
│         ──RESP(──┬──N──┬──)──────────────            │
│                  └──Y──┘                             │
│         ──SEGNO(number)──────────────────           │
│         ──SEGSZ(size)────────────────────           │
│         ──SERIAL(──┬──N──┬──)────────────            │
│                    └──Y──┘                           │
│         ──SIDL(localsysid)───────────────           │
│         ──SIDR(remotesysid)──────────────           │
│         ──SPASZ(size)────────────────────           │
│         ──SPATRUNC(──┬──R──┬──)──────────            │
│                      └──S──┘                         │
│         ──TRANSTAT(──┬──N──┬──)──────────            │
│                      └──Y──┘                         │
│         └──WFI(──┬──N──┬──)──────────────┘
│                  └──Y──┘
```

Keywords

The following keywords are valid for the UPDATE TRAN command:

CLASS()
> Selects the transactions associated with the specified class or classes to be updated.

NAME(*tranname*)
> Specifies the 1-8 character name of the transaction. Wildcards can be specified for the *tranname*. The *tranname* is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the resource names that are processed. For NAME(*) command responses are returned only for the resource names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the resource names that are processed.

OPTION()
> Specifies the additional functions to be performed. Following is a list of additional functions:

> **AFFIN**
>> AFFIN is valid with START(SCHD) or STOP(SCHD).

>> When used with START(SCHD), OPTION(AFFIN) indicates that the transaction has local affinity to the IMS and that an inform request should be performed to register interest in the local affinity queue. In a shared-queues (SQ) environment, messages for transactions that have affinity to the local IMS are placed on the coupling facility by using a queue name that consists of the transaction name with the local IMSID or RSENAME (XRF) appended. Local affinity registration, registration for a queue name that consists of the transaction name with the local IMSID or RSENAME (XRF) appended, is performed in addition to the normal transaction registration, registration for a queue name that consists of the transaction name with blanks appended. When you register interest in a queue, IMS™ is notified when messages are placed on the queue for processing.

>> When used with STOP(SCHD), OPTION(AFFIN) indicates that the IMS should unregister interest in the transaction's local affinity queue.

>> After you start a transaction with OPTION(AFFIN), local affinity status (AFFIN) is set and remains set for the transaction until IMS is cold started. The QUERY TRAN NAME(*tranname*) SHOW(ALL) command shows the status of AFFIN. If a subsequent UPDATE TRAN NAME(*tranname*) STOP(SCHD) or UPDATE TRAN NAME(*tranname*) START(SCHD) command is performed, the command toggles registration of interest in both the normal and local affinity transaction queues, whether OPTION(AFFIN) is specified or is not.

>> Local affinity status is not maintained across an IMS cold start. At the end of cold start processing, when shared-queues informs are completed, IMS registers interest in the normal transaction queues only. It does not register interest in the local affinity queues. As a result, messages on a local affinity queue cannot be scheduled. The UPDATE TRAN START(SCHD) OPTION(AFFIN) command corrects this situation by performing an inform request to register interest in the local affinity queue for the IMSID.

>> If the DFSMSCE0 user message routing exit is used to set local affinity for an input transaction message in a shared-queues environment and the IMS

control region is stopped, and subsequently cold started, you need to perform the UPDATE TRAN NAME(*tranname*) START(SCHD) OPTION(AFFIN) command so that an inform request is performed to register interest in the local affinity queue for the IMSID.

ALLRSP

Indicates that the response lines are to be returned for all resources that are processed on the command. The default action is to return response lines only for the resources that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SCOPE()

Specifies where IMS should apply the change. The default is ALL.

ALL

Changes are applied to the active IMS systems to which the command is routed. If you specify that global area status is to be maintained, changes are also applied globally by updating the value maintained by RM. The RM status is updated only by the command master IMS. If global area status is not maintained, the command action is same as the SCOPE(ACTIVE) command.

ACTIVE

Changes are applied to the active IMS systems to which the command is routed.

RM maintains global status information for some transactions. IMS updates that information in RM based on commands. When SCOPE(ALL) is specified, every IMS system that processes the command updates information in its local control blocks. Only the IMS command master calls RM to update the information globally.

SET()

Specifies the attribute values to be changed. A transaction must have the same characteristics in all systems where it is defined when it is shared. These characteristics include:

- Nonconversational or conversational
- SPA size if conversational
- Single-segment or multisegment messages
- Non-inquiry or inquiry
- Recoverable or nonrecoverable

AOCMD

Specifies the AOI option that you want to change, which indicates whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call. If AOCMD is defined as CMD, TRAN, or Y, and the AOI1 execute parameter is defined as AOI1=N, no authorization checking is done and the transaction is permitted to issue CMD and ICMD calls.

CMD

Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, authorization checking is based on which transactions can issue a particular command. In this case, the commands (or the first three characters of the commands) need to be defined to RACF or an equivalent product as a user. The type-1 AOI transactions must be defined as profiles under the TIMS class, and for each transaction, the commands it can issue must be specified.

Defining AOCMD(CMD) requires you to create fewer user IDs than you need to create for the AOCMD(TRAN) definition. However, defining AOCMD(CMD) requires you to create or modify a larger number of resource profiles.

N Indicates that the transaction is not permitted to issue type-1 AOI CMD calls. The transaction is permitted to issue type-2 AOI ICMD calls.

TRAN

Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the transaction code is used for authorization. The first authorization check results in the accessor environment element (ACEE) being built. This environment is kept for use by future authorization checks. The type-1 AOI transaction needs to be defined to RACF or an equivalent product as a user. The transactions will then be specified on RACF PERMIT statements for each command they are allowed to issue from a type-1 AOI transaction. Specifying AOI transactions as users to RACF might conflict with the name of a user already defined to RACF. If this occurs, then either the transaction name or the existing user name needs to be changed.

Y Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the user ID or the program name is used for authorization. For some environments, if a Get Unique call has not yet occurred, the program name is used for authorization.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

CLASS

Specifies the transaction class, which is an attribute used to select a transaction for scheduling. A transaction can be scheduled if there is a message processing region available for that class. The value can be a number from 1 to 999. This value must not exceed the value given (by specification or default) on the MAXCLAS= keyword of the IMSCTRL macro.

Define CPI-C transactions with a different message class from that used for non-CPI-C transactions. IMS handles all CPI-C transactions as priority zero within the transaction class.

CMTMODE

Specifies when database updates and non-express output messages are committed. This operand affects emergency restart.

MULT

Database updates and non-express output messages are committed only when the application program terminates normally, when the processing limit count has been reached, or, in the case of a pseudo-WFI dependent region, when there are no more messages on the queue. For example, if five transactions are processed during a

single schedule of a program, all five are committed only when the fifth one is completed and the program terminates. Until a transaction has been committed, locks for updated database records are not released and non-express output messages are not queued for output. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing.

If the transaction results in the application calling an external subsystem, such as DB2, the Commit Verify exit provided by the external subsystem can determine whether CMTMODE(MULT) is supported. See documentation under the Commit Verify exit routine in *IMS Version 14 Exit Routines*.

SNGL
Database updates and non-express output messages are committed when the application program completes processing each transaction. IMS invokes commit processing either when the application program requests the next message (issues a GU to the IO-PCB), or when the application program terminates. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing. If an application ends abnormally before committing its message, emergency restart requeues the message that was in process before the abend and makes it available for reprocessing.

Keyword combination rules include the following:
- CONV(Y) and CMTMODE(MULT) are mutually exclusive.
- WFI(Y) and CMTMODE(MULT) are mutually exclusive.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

CONV
Specifies the conversation option.

N   The transaction is not conversational.

Y   The transaction is conversational. The transaction message is destined for a conversational program. A conversational program processes transactions made up of several steps. A conversational program receives a message from a terminal, replies to the terminal, but saves the data from the transaction in a scratchpad area (SPA). When the person at the terminal enters more data, the program has the data it saved from the last message in the SPA, so it can continue processing the request without the person at the terminal having to enter the data again.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There

must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- CONV(Y) requires SPASZ and SPATRUNC.
- CMTMODE(MULT) and CONV(Y) are mutually exclusive.
- FP(E) and CONV(Y) are mutually exclusive.
- RECOVER(N) and CONV(Y) are mutually exclusive.
- SPASZ and CONV(N) are mutually exclusive.
- SPATRUNC and CONV(N) are mutually exclusive.

CPRI
Specifies a new value for the current priority of a transaction. The CPRI keyword is not allowed for BMP transactions, because BMP transactions should always have a priority of 0. The new CPRI value takes effect the next time the transaction is scheduled. Valid CPRI parameters are numeric values from 0 to 14.

DCLWA
Specifies the log write-ahead option.

N IMS should not perform log write-ahead. Specify N if input message integrity and the consistency of output messages with associated database updates is not required. DCLWA does not apply to response mode or Fast Path input processing, and is ignored during IMS execution.

Y IMS should perform log write-ahead for recoverable, nonresponse input messages and transaction output messages. This ensures the following:
- A nonresponse input transaction is made recoverable across IMS failures, prior to IMS acknowledging receipt of the input.
- Database changes are made recoverable prior to IMS sending associated output reply messages.
- Information in the log buffers is written to the IMS log, before the associated input acknowledgment or output reply is sent to the terminal.

Define DCLWA(Y) for all VTAM terminal types.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

DIRROUTE
Specifies the MSC directed routing option.

N The application program processing a transaction is not informed of the system which originated the transaction. The name of the originating LTERM is placed in the I/O PCB.

Y The application program processing a transaction is informed of the system which originated the transaction, if MSC directed routing is

used in a multiple IMS system configuration. An MSNAME corresponding to a logical path back to the originating system is placed in the I/O PCB.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

EDITRTN

Specifies the 1- to 8-character name of your transaction input edit routine that edits messages before the program receives the message. This name must begin with an alphabetic character. The specified edit routine (load module) must reside on the USERLIB data set before IMS system definition stage 2 execution. This routine cannot be the same one defined by the system definition TYPE EDIT= parameter.

EDITRTN is used for a Fast Path potential transaction when the transaction is routed to IMS.

For input from LU 6.2 devices, the user edit exit routine DFSLUEE0 is called instead of the transaction input edit routine specified in EDIT.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

To remove an edit routine name from a transaction definition, specify EDITRTN(0).

FP(E) and EDITRTN are mutually exclusive.

EDITUC
Specifies the edit to uppercase option.

N   The input data is not translated to uppercase. It can consist of uppercase and lowercase characters as entered from the terminal.

Y

The input data is to be translated to uppercase before it is presented to the processing program. If FP(E) or FP(P), the transaction is to be translated to uppercase before being presented to the edit/routing exit routine.

Specifying EDITUC(Y) for VTAM terminals prevents the transmission of embedded device control characters.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

EMHBSZ

Specifies the EMH buffer size required to run the Fast Path transaction. This overrides the EMHL execution parameter. If EMHBSZ is not specified, the EMHL execution parameter value is used. The value can be a number from 0 to 30 720.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- EMHBSZ > 0 requires Fast Path to be defined.
- FP(N) and EMHBSZ > 0 are mutually exclusive.

EXPRTIME
Specifies the elapsed time in seconds that IMS can use to cancel the input transaction. After a transaction is submitted to IMS, the transaction could be delayed for processing because of a stopped transaction or a potential system slow down. In that case, the user or client application could time out before the transaction is processed. When IMS eventually schedules and processes the transaction, the response message is no longer wanted. With the elapsed time specified for the transaction, IMS can flag the input transaction as expired so that the system does not waste CPU cycles to process the unwanted transaction.

The value can be a number, in seconds, which can range from 0 to 65535. The default is 0, which means that no expiration time is set for this transaction. The transaction expiration attribute is supported by all of the IMS TM interfaces.

Restriction: The transaction expiration checking is not performed at the GU time for Fast Path transactions, IMS conversational transactions, and program-to-program switch transactions.

FP Specifies the Fast Path option.

E The transaction is processed exclusively as Fast Path. The program must be defined as Fast Path exclusive.

N The transaction is not a candidate for Fast Path processing. The program must be defined as not Fast Path. When FP(N) is specified, any attempt to use Fast Path resources or commands might yield unpredictable results.

P

The transaction is a potential candidate for Fast Path processing. Fast Path-potential transactions must be able to run under two applications: a Fast Path exclusive application and a non-Fast Path application. A Fast Path exclusive application should be defined to which this transaction can be routed. Fast Path–potential transactions must be processed by a user edit/routing exit to determine whether the transaction is actually to be processed by IMS Fast Path. If it is to be processed by IMS Fast Path, the edit/routing exit routine associates the transaction with a routing code. This routing code identifies which Fast Path application program is to process the transaction.

The program defined by PGM() must not be defined as Fast Path exclusive.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

In order to update a transaction from FP(E) to FP(N) or from FP(N) to FP(E) you must also update the transaction to point to a program that has the same FP() attribute. If you do not update the program attribute, the command will fail because of a program conflict.

If a Fast Path exclusive FP(E) transaction is being updated to Fast Path potential FP(P) or non-Fast Path FP(N) transaction and no limit count is specified, a limit count of 65535 is set.

Keyword combination rules include the following:
- CONV(Y) and FP(E) are mutually exclusive.
- EDITRTN and FP(E) are mutually exclusive.
- EMHBSZ>0 and FP(N) are mutually exclusive.
- FP(E) and FP(P) require Fast Path to be defined.
- MSGTYPE(MULTSEG) and FP(E) are mutually exclusive.
- MSGTYPE(MULTSEG) and FP(P) are mutually exclusive.
- MSNAME and FP(E) are mutually exclusive.
- RECOVER(N) and FP(E) are mutually exclusive.
- RECOVER(N) and FP(P) are mutually exclusive.
- RESP(N) and FP(E) are mutually exclusive.
- RESP(N) and FP(P) are mutually exclusive.
- SIDL and FP(E) are mutually exclusive.
- SIDR and FP(E) are mutually exclusive.

INQ

Specifies the inquiry option.

N   This is not an inquiry transaction.

Y

This is an inquiry transaction. If INQ(Y) is specified, you can also specify whether this transaction should be recovered during an IMS emergency or normal restart using the RECOVER() parameter.

This should be specified only for those transactions that, when entered, do not cause a change in any database. Programs are prohibited from issuing ISRT, DLET, or REPL calls to a database when scheduled to process a transaction defined as INQ(Y).

An application program cannot do an SQL INSERT, DELETE, or UPDATE when the IMS transaction is defined with INQ(Y).

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There

must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:

- RECOVER(N) and INQ(N) are mutually exclusive.

LCT

Specifies the limit count. This is the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction. The value can be a number from 1 to 65535. The default is 65535.

The limit count value is ignored for a transaction processed by a BMP.

The limit count value is ignored in a shared-queues environment.

The limit count value does not apply to FP exclusive transactions and is ignored.

LOCK

Specifies that the LOCK status is to be set on or off. SET(LOCK(ON | OFF)) cannot be specified with any other SET attribute. SET(LOCK(ON | OFF)) can be specified with the START or STOP keyword.

ON Locks the transaction and prevents it from being scheduled. LOCK(ON) cannot be specified for Fast Path exclusive transactions, but can be specified for Fast Path potential transactions. LOCK(ON) cannot be specified for transactions for CPI Communications driven programs.

OFF

Unlocks the transaction and allows it to be scheduled.

LPRI

Specifies the limit priority. This is the scheduling priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The value can be a number from 0 through 14.

When the limit priority is used and the scheduling priority is raised to the limit priority, the priority is not reduced to the normal priority until all messages enqueued for this transaction name are processed. If you do not want the limit priority for this transaction, define equal values for the normal priority and limit priority, and a limit count of 65535.

When a transaction is processed exclusively by a batch message program (BMP), define the limit priority as 0. If the program specified by PGM() is defined with a program type of batch, the current priority is forced to be 0. However, a batch message processing region (BMP) can process transactions with current scheduling priorities other than 0.

This priority also controls the priority of messages created by this transaction and sent to a destination in a remote system. See also the discussion on MSC priorities under the NPRI definition.

The limit priority value is ignored for a transaction processed by a BMP.

The limit priority value is ignored in a shared-queues environment.

MAXRGN

Specifies a new value for the maximum number of regions that can be simultaneously scheduled for a given transaction. The transaction must be eligible for parallel scheduling (load balancing). The value of the MAXRGN parameter must be between 0 and the number specified on the MAXPST=control region parameter.

The keyword limits the number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction. When the number of MPP regions is not limited, one transaction might monopolize all available regions. The value can be a number from 0 to the value specified on the MAXPST= control region parameter. MAXRGN(0) means that no limit is imposed.

If you define the application program with a scheduling type of SERIAL, omit the MAXRGN keyword or define it as 0.

The following keyword combinations are mutually exclusive:
- PARLIM(65535) and MAXRGN value greater than 0
- SERIAL(Y) and MAXRGN value greater than 0

MSGTYPE

Specifies the message type (single segment or multiple segment). It specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing.

If MSC-directed routing is used in a multiple IMS system configuration, IMS does not ensure that both the message and the transaction destined to process that message are either single segment or multiple segments.

MULTSEG

The incoming message can be more than one segment in length. It is not eligible for scheduling to an application program until an end-of-message indication is received, or a complete message is created by MFS.

SNGLSEG

The incoming message is one segment in length. It becomes eligible for scheduling when the terminal operator indicates end-of-segment.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- FP(E) and MSGTYPE(MULTSEG) are mutually exclusive.
- FP(P) and MSGTYPE(MULTSEG) are mutually exclusive.

MSNAME

Specifies the one- to eight-character name of the logical link path in a multiple IMS system configuration (MSC). A logical link path is a path between any two IMS systems. The IMS systems are identified by the remote system ID and the local system ID associated with the logical link path. The remote system ID identifies the system in which messages using

this path are to be processed. The local system ID identifies this system. For an UPDATE TRAN command that is changing a transaction to a remote transaction, or changing the MSC path, the new MSNAME must already be defined.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- FP(E) and MSNAME are mutually exclusive.
- SIDL and MSNAME are mutually exclusive.
- SIDR and MSNAME are mutually exclusive.

NPRI

Specifies the normal scheduling priority. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The normal priority is assigned to the transaction as the scheduling priority when the number of input transactions enqueued and waiting to be processed is less than the limit count value. The value can be a number from 0 through 14. The default is 1.

This priority also controls the priority of messages created by this transaction and sent to a destination in a remote system.

When a transaction is processed exclusively by a batch message program (BMP), code the normal priority as 0.

When a transaction is processed exclusively by a batch message program (BMP), define the limit priority as 0. If the application program specified by PGM() is defined with a program type of batch, the current priority is forced to be 0. However, a batch message processing region (BMP) can process transactions with current scheduling priorities other than 0.

For remote transactions, the priority used to send the transaction to the processing system, which is termed *the MSC link message priority*. The three MSC link message priority groups are:
- Low
- Medium
- High

The low priority group consists of primary requests in the input terminal system. This group is assigned remote transaction priorities from 0 to 6. The medium group consists of secondary requests, responses, primary requests in an intermediate system, and primary requests in the input terminal system. This group is assigned a remote transaction priority of 7. The high group consists of primary requests in the input terminal system. Messages in this group are assigned remote transaction priorities from 8 to 14. Within each group, messages have a priority based on the current priority value of the transaction or remote transaction in the input terminal system for primary requests, and on the latest processing system for secondary requests and responses.

In an MSC configuration, the transaction priority determines the priority used to send messages inserted by this transaction across an MSC link. If the transaction inserts multiple messages to the same destination (for example, pages to a printer) and these messages must be sent in the order inserted, the normal and limit priority values should be the same. If the normal and limit priority values are not identical, messages inserted at a higher priority than previously inserted messages could arrive at their destination first. (This restriction does not apply to multiple segments of the same message.)

The normal priority value is ignored for a transaction processed by a BMP.

PARLIM

Specifies the parallel processing limit count. This is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. This is the threshold value to be used when the associated application is defined with a scheduling type of parallel. An additional region is scheduled whenever the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.

The value can be a number from 0 to 32767 or 65535. PARLIM(0) indicates that any input message can cause a new region to be scheduled because the scheduling condition is always met (the number of messages is greater than zero). If you specify PARLIM(0), you should specify a MAXRGN value to limit the number of regions that can be scheduled to process a particular transaction. PARLIM(65535) means that parallel processing is disabled and IMS allows the transaction to be scheduled in only one region at a time.

The value specified for PARLIM applies to message processing programs (MPPs) only; it is not supported for batch message processing programs (BMPs).

If you define the application as SERIAL or the scheduling type as SERIAL, define PARLIM(65535).

In a shared-queues environment (when the scheduling type is PARALLEL), any PARLIM value other than 65535 causes a new region to be scheduled whenever the successful consecutive GU count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction. For shared-queues environments, the successful consecutive GU count is used instead of the queue count. New regions continue to be scheduled up to the maximum number of regions specified on MAXRGN.

Note: In a shared-queues environment, the PARLIM value behaves differently than it does in a non-shared-queues environment. In a non-shared-queues environment, the queue depth (the number of messages that are currently enqueued) for the transaction is used as the value that is compared with the PARLIM value to determine when to schedule another region. IMS responds to a growing queue of input transactions by scheduling more regions as the queue grows.

In a shared-queues environment, each individual IMS does not know the depth of the queue, because the queue is in the shared-queues coupling facility structure that is managed by Common Queue Server (CQS). The transaction queue might be added to by many different IMS systems. IMS

is notified only when the first message is put in a queue (that is, when the queue becomes *not empty*). IMS is not notified for every subsequent message that is placed on the queue after that first message. In a shared-queues environment, the PARLIM comparison is done against a counter that each IMS keeps of the number of successful consecutive GU calls for the transaction by that IMS, rather than queue depth. IMS schedules more regions when it consistently gets messages from CQS when it asks for them. Thus, in a shared-queues environment, IMS infers the depth of the queue of messages based on processing activity, but it does not know the actual depth of the queue.

A PARLIM value of 0 in a shared-queues environment is the most responsive setting. PARLIM(0) ensures that message regions are scheduled until all messages are processed from the transaction queue, or until the maximum region value (MAXRGN) limit is reached. PARLIM(0) might, however, result in many unnecessary schedules (or *false schedules*). A false schedule occurs when a message region is scheduled and finds no more messages on the queue. This occurs particularly with PARLIM(0) because after each successful get unique (GU), IMS must always schedule an additional region to try to read the queue to see if there are more messages. This process continues for each successful GU until the queue becomes empty, at which time the successive GU count is reset to 0. This is a consequence of IMS not knowing how many messages are queued on the transaction queue.

Setting the PARLIM to a value greater than 0 can reduce the number of false schedules, because IMS then schedules a new message region only after a number of messages have been obtained consecutively without the queue becoming empty. Setting the PARLIM to a value of 2 or greater is useful for reducing false schedules for transactions that are low-volume and that run relatively quickly (such that the queue depth is typically 1), because it avoids scheduling a second region until the first region has obtained at least two messages in a row. However, be aware that while a PARLIM value greater than 0 can reduce unnecessary schedules, it is also less responsive. If a transaction is long running, or if its processing is delayed (for example, because of locking contention), the consecutive GU count does not change while the transaction is executing, and no additional message regions are scheduled. This can result in delayed processing of other messages for this same transaction until a currently-scheduled message completes. This delay can occur even if message regions are available to process the transaction.

Keyword combination rules include the following:
- MAXRGN>0 and PARLIM(65535) are mutually exclusive.
- SERIAL(Y) and PARLIM between 0 and 32767 are mutually exclusive.

PGM

Specifies the name of the application program associated with the transaction. The program must exist unless the transaction is defined as REMOTE(Y).

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There

must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

PLCT

Specifies the processing limit count. This is maximum number of messages sent to the application program by the IMS for processing without reloading the application program. The value must be a number from 0 through 65535. PLCT(0) means the maximum number of messages sent to the application is one and the application program is reloaded before receiving a subsequent message. PLCT(65535) means that no limit is to be placed upon the number of messages processed at a single program load. Values 1 through 65535 are eligible for quick reschedule processing.

The value is used to determine how many messages an application program is allowed to process in a single scheduling cycle. When the application program requests, and receives, the number of messages indicated, any subsequent requests result in one of two things.

1. IMS indicates "no more messages exist" if any of the following conditions is true.
   - The region is not an MPP.
   - The currently scheduled mode is not CMTMODE(SNGL).
   - Equal or higher priority transactions are enqueued for the region.

   IMS might, in fact, have other messages enqueued for the application program. It is the responsibility of the application program to terminate when it receives an indicator that no more messages are available. Termination of the application program makes the region it occupied available for rescheduling. This feature makes it possible for IMS to enable scheduling of higher priority transactions that entered the system while the previous transactions were in process. In addition, if any equal-priority transactions are enqueued, they become eligible for scheduling on a first-in, first-out (FIFO) basis.

2. The region goes through quick reschedule and returns the next message to the application of all of the following conditions are true.
   - The region is an MPP.
   - The transaction is CMTMODE(SNGL).
   - No equal or higher transactions are enqueued.
   - Messages are still enqueued for the application.

PLCTTIME

Specifies the processing limit count time. This is the amount of time (in hundredths of seconds) allowable to process a single transaction (or message). The number specifies the maximum CPU time allowed for each message to be processed in the message processing region.

Batch Message Programs (BMPs) are not affected by this setting.

The value can be a number, in hundredths of seconds, that can range from 1 to 6553500. A value of 6553500 means no time limit is placed on the application program.

If Fast Path is used, this keyword specifies, for a given transaction name, the amount of time (in hundredths of seconds) the program is allowed to process a single transaction message. The time represents real time that elapses during transaction processing (not accumulated task time). Real

time is used because the input terminal is in response mode and cannot enter another transaction until the response is sent. In this case PLCT() is ignored.

The value controls application program looping. You are not required to optimize the value for program-transaction execution time. However, the time value assigned should not be less than the expected per-transaction execution time. If the scheduled application program exceeds the product of PLCTTIME() and PLCT(), the application program ends abnormally. If an IMS STIMER value of 2 is specified on the DFSMPR procedure, the region does not abend until completion of the DL/I call.

Important: The application program must not use STIMER timer services. IMS uses STIMER timer services to time the execution of transactions. If an application program issues an MVS STIMER macro, it cancels the STIMER timer services set by IMS. Use the STIMERM macro instead for application program timer requests.

RECOVER
Specifies the recovery option.

N The transaction should not be recovered.

Y The transaction should be recovered during an IMS emergency or normal restart.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- CONV(Y) and RECOVER(N) are mutually exclusive.
- FP(E) and RECOVER(N) are mutually exclusive.
- FP(P) and RECOVER(N) are mutually exclusive.
- INQ(N) and RECOVER(N) are mutually exclusive.

REMOTE
Specifies the remote option.

N The transaction is not remote. The transaction is local and runs on the local system.

Y The transaction is remote. The transaction runs on a remote system.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Local transactions must have the SIDR value set equal to the SIDL value. When updating a transaction from remote to local, the SIDR and SIDL keywords must be specified, along with REMOTE(N), to set the remote SYSID equal to the local SYSID.

Keyword combination rules include the following:

- REMOTE(Y) requires MSNAME or SIDR and SIDL.

RESP

Specifies the response mode option.

N   The transaction is not response mode. For terminals specifying or accepting a default of OPTIONS=TRANRESP, input should not stop after this transaction is entered.

Y   The transaction is response mode. The terminal from which the transaction is entered is held and prevents further input until a response is received. For terminals specifying or accepting a default of OPTIONS=TRANRESP, no additional messages are to be allowed after this transaction is entered until this transaction sends a response message back to the terminal. Response mode can be forced or negated by individual terminal definition. RESP(Y) is ignored during online processing for all terminals that do not operate in response mode.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- FP(E) and RESP(N) are mutually exclusive.
- FP(P) and RESP(N) are mutually exclusive.

SEGNO

Specifies the segment number. This is the maximum number of application program output segments that are allowed into the message queues per Get Unique (GU) call from the application program. The value can be a number from 0 through 65535. If SEGNO(0) is defined, the number of segments is not checked by the online system at execution time.

SEGNO cannot be specified for CPI Communications driven transactions.

SEGSZ

Specifies the segment size. This is the maximum number of bytes allowed in any one output segment. The value can be a number from 0 through 65535. If SEGSZ(0) is defined, the segment size is not checked by the online system at execution time.

The maximum output message segment to an LU 6.2 device is 32767. If a transaction is expected to send output to an LU 6.2 device, the SEGSIZE parameter should be no greater than 32767. However, this is not enforced during processing of the command, because IMS cannot determine the device type for the message destination until output time.

SEGSZ cannot be specified for CPI Communications driven transactions.

SERIAL

Specifies the serial option.

N   Messages for the transaction are not processed serially. Message processing can be processed in parallel. Messages are placed on the suspend queue after a U3303 pseudoabend. Scheduling continues until repeated failures result in the transaction being stopped with a USTOP.

Y   Messages for the transaction are processed serially. U3303

pseudoabends do not cause the message to be placed on the suspend queue but rather on the front of the transaction message queue, and the transaction is stopped with a USTOP. The USTOP of the transaction is removed when the transaction or the class is started with a /START or UPD TRAN command.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- MAXRGN>0 and SERIAL(Y) are mutually exclusive.
- PARLIM value 0 - 32767 and SERIAL(Y) are mutually exclusive.

SIDL

Specifies the system identification (SYSID) of the local system in a multiple-IMS system (MSC) configuration. The local system is the originating system to which responses are returned. The value can be a number from 1 to 2036, if MSC is enabled, or 0, if MSC is not enabled. The local SYSID can be defined in any or all of the MSNAMEs or transactions.

The SIDL parameter is independent of the link type (CTC, MTM, TCP/IP, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- FP(E) and SIDL are mutually exclusive, unless SIDL and SIDR are specified as a pair and are equal to the local system ID of this IMS.
- MSNAME and SIDL are mutually exclusive.
- SIDL value must be defined to this IMS.

SIDR

Specifies the system identification (SYSID) of the remote system in a multiple-IMS system (MSC) configuration. The remote system is the system on which the application program executes. The value can be a number from 1 to 2036, if MSC is enabled, or 0, if MSC is not enabled. The remote SYSID specified must also be defined for an MSNAME.

The SIDR parameter is independent of the link type (CTC, MTM, TCP/IP, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement.

Local transactions must have the SIDR value set equal to the SIDL value. When updating a transaction from remote to local, the remote SYSID must be set to the local SYSID.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a

message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- FP(E) and SIDR are mutually exclusive, unless SIDL and SIDR are specified as a pair and are equal to the local system ID of this IMS.
- MSNAME and SIDR are mutually exclusive.
- SIDR value must be defined to this IMS.

SPASZ

Specifies the scratchpad area (SPA) size, in bytes, for a conversational transaction. The value can be a number from 16 and 32767.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- CONV(N) and SPASZ are mutually exclusive.
- FP(E) and SPASZ are mutually exclusive.

SPATRUNC

Specifies the scratchpad area (SPA) truncation option of a conversational transaction. This defines whether the SPA data should be truncated or preserved across a program switch to a transaction that is defined with a smaller SPA.

When a conversation initially starts, and when a program is switched, the SPATRUNC option is checked and set or reset as specified. When the option is set, it remains set for the life of the conversation, or until a program switch occurs to a transaction that specifies the option is to be Reset.

When a program switch occurs, the truncated data option for the new transaction is first checked, and that specification is set for the conversation and is used for the SPA inserted into the output message. If the option is not specified for the new transaction, the option currently in effect for the conversation is used.

S   IMS preserves all of the data in the SPA, even when a program switch is made to a transaction that is defined with a smaller SPA. The transaction with the smaller SPA does not see the truncated data, but when the transaction switches to a transaction with a larger SPA, the truncated data is used.

R   The truncated data is not preserved.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- CONV(N) and SPATRUNC are mutually exclusive.
- FP(E) and SPATRUNC are mutually exclusive.

TRANSTAT

Specifies whether transaction level statistics should be logged for message driven programs. If Y is specified, transaction level statistics are written to the log in a X'56FA' log record.

N Transaction level statistics should not be logged.

Y Transaction level statistics should be logged.

The TRANSTAT keyword on the UPDATE TRAN command gives the user the ability to override the system default or the current value of the TRANSTAT parameter. If the TRANSTAT keyword is omitted on the UPDATE TRAN command, the current transaction level statistics setting is unchanged.

WFI

Specifies the wait-for input option. This attribute does not apply to Fast Path transactions, which always behave as wait-for-input transactions.

N This is not a wait-for-input transaction.

Y This is a wait-for-input transaction. A message processing or batch processing application program that processes WFI transactions are scheduled and invoked normally. If the transaction to be processed is defined as WFI, the program is allowed to remain in main storage after it has processed the available input messages. The QC status code (no more messages) is returned to the program if the PROCLIM count (PLCT) is reached; a command is entered to change the status of the scheduled transaction, database, program, or class; commands relating to the databases used by the transaction are entered, or IMS is terminated with a checkpoint shutdown.

Updating this attribute stops messages from being queued to the transaction while the command is in progress. Any attempt to queue a message to the transaction is rejected. Updating this attribute also stops the transaction from being scheduled while the command is in progress. There must be no work in progress and no messages queued (non-shared queues) for the command to complete successfully.

Keyword combination rules include the following:
- MODE(MULT) and WFI(Y) are mutually exclusive.

START()

Specifies the attributes to be started.

Q Updates local and global status to start queuing transactions. This change applies to transactions that are entered either from a network or terminal, or using the QUEUE TRAN command, to DB/DC or DCCTL environments.

SCHD

Updates local and global status to start transactions from being scheduled. This change applies to transactions scheduled in DB/DC or DCCTL environments.

In a shared-queues environment, the UPD TRAN START(SCHD) command will result in IMS registering interest for the transaction, which indicates that the transaction can be scheduled at that IMS. An UPDATE TRAN NAME(*) START(SCHD) command does not register transactions that are already registered to CQS.

SUSPEND
   If the transaction has messages on the suspend queue, that suspend queue is automatically transferred to the ready queue.

TRACE
   Starts the transaction trace, which captures the DL/I portion of Data Communications (DC) for the specified transaction. The information is written as a 6701 log record to the IMS log.

STOP()
Specifies the attributes to be stopped.

Q Updates local and global status to stop transactions from being queued. This change applies to transactions that are entered either from a network or terminal, or using the QUEUE TRAN command, to DB/DC or DCCTL environments.

SCHD
   Updates local and global status to stop transactions from being scheduled. This change applies to transactions scheduled in DB/DC or DCCTL environments.

   The UPDATE TRAN STOP(SCHD) command stops the scheduling of transactions; however, the transactions will continue to be processed until the limit count is reached. If the limit count is large, the processing interval will be long.

   In a shared-queues environment, the UPD TRAN STOP(SCHD) command results in IMS deregistering interest for the transaction, which indicates that the transaction cannot be scheduled at the IMS.

TRACE
   Stops the transaction trace.

Usage notes

Resources exist for the life of the IMS unless they are deleted using a DELETE command. Resources are recoverable across an IMS warm start or emergency restart. Resources are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up.

The UPDATE TRAN command is recoverable across an IMS warm start or emergency restart, except for START(TRACE) and STOP(TRACE). Updates to the definitional attributes of a transaction are lost if IMS is cold started, unless the transaction definitions are exported to an RDDS and then imported from the RDDS at the cold start.

The UPDATE TRAN command can be issued only through the OM API. This command applies to DB/DC and DCCTL systems.

This command is not valid on the XRF alternate, RSR tracker, or FDBR region.

The UPDATE TRAN command changes a MODBLKS transaction to dynamic, if the AOCMD, CMTMODE, CONV, DCLWA, DIRROUTE, EDITRTN, EDITUC,

EMHBSZ, FP, INQ, MSGTYPE, MSNAME, PLCTTIME, PGM, RECOVER, REMOTE, RESP, SERIAL, SPASZ, SPATRUNC, TRANSTAT, or WFI value is changed.

Each transaction is updated individually, unlike the online change process where either all transactions are updated or no transactions are updated. Some runtime resource definition values for a transaction can be updated only if the transaction is not in use.

You can update the status of a transaction (LOCK, START, STOP) while the transaction is in use.

The following transaction attributes cannot be updated if online change for MODBLKS is enabled: AOCMD, CMTMODE, CONV, DCLWA, DEFAULT, DIRROUTE, EDITRTN, EDITUC, EMHBSZ, FP, INQ, MSGTYPE, MSNAME, PGM, PLCTTIME, RECOVER, REMOTE, RESP, SERIAL, SIDL, SIDR, SPASZ, SPATRUNC, WFI.

If all the attributes specified by the UPDATE command are already defined for the resource, no update is actually made, no resources are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

In a shared message queues environment, the UPDATE TRAN command updates the transaction even if there are messages on the shared messages queues for the transaction. If the messages were placed on the shared message queues using conversation, Fast Path, response mode, or serial transaction attributes that differ from the transaction being updated on this IMS, this IMS will not be able to access the messages on the shared message queues.

The UPDATE TRAN NAME(tranname) SET(REMOTE(N),SIDR(localsysid),SIDL(localsysid)) command sets the remote system ID to be the same as the local system ID. It is equivalent to the command /MSASSIGN TRAN tranname TO LOCAL. The UPDATE TRAN NAME(tranname) SET(REMOTE(N)) command specified without SIDR/SIDL or MSNAME does not change the local and remote system IDs, therefore it is not equivalent to the /MSASSIGN TRAN tranname TO LOCAL command.

Equivalent IMS type-1 commands

The following table shows variations of the UPDATE TRAN command and the type-1 IMS commands that perform similar functions.

Table 545. Type-1 equivalents for the UPDATE TRAN command

UPDATE command	Similar IMS type-1 command
UPDATE TRAN(name) START(Q) STOP(SCHD)	/PSTOP TRAN *name*
UPDATE TRAN NAME(name) START(SCHD) STOP(Q)	/PURGE TRAN *name*
UPDATE TRAN NAME(name) START(Q,SCHD,SUSPEND)	/START TRAN *name*
UPDATE TRAN NAME(name) STOP(Q,SCHD)	/STOP TRAN *name*

Table 545. Type-1 equivalents for the UPDATE TRAN command (continued)

UPDATE command	Similar IMS type-1 command
UPDATE TRAN NAME(name) START(TRACE)	/TRACE SET ON TRAN *name*
UPDATE TRAN NAME(name) STOP(TRACE)	/TRACE SET OFF TRAN *name*
UPDATE TRAN NAME(name) SET(CLASS(*new_class_number*))	/ASSIGN TRAN *name* TO CLS *new_class_number*
UPDATE TRAN NAME(name) SET(CPRI(*new_current_priority*))	/ASSIGN CPRI *new_current_priority* TO TRAN *name*
UPDATE TRAN NAME(name) SET(LCT(*new_limit_count*))	/ASSIGN LCT *new_lmct_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(LPRI(*new_limit_priority*))	/ASSIGN LPRI *new_lpri_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(NPRI(*new_normal_priority*))	/ASSIGN NPRI *new_npri_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(PARLIM(*new_parallel_limit*))	/ASSIGN PARLIM *new_parmlim_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(PLCT(*new_processing_limit*))	/ASSIGN PLCT *new_plmct_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(SEGNO(*new_segment_number*))	/ASSIGN SEGNO *new_segno_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(SEGSZ(*new_segment_size*))	/ASSIGN SEGSZ *new_segsize_number* TO TRAN *name*
UPDATE TRAN NAME(name) SET(MAXRG(*new_max_regions*))	/CHANGE TRAN *name* MAXRGN *new_maxrgn_number*

Output fields

The following table shows the UPDATE TRAN output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies the keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 546. Output fields for the UPDATE TRAN command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the nonzero completion code.

Table 546. Output fields for the UPDATE TRAN command (continued)

Short label	Keyword	Meaning
CONVID	TRAN	Conversation ID of conversation associated with transaction that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a nonzero completion code and contains information that further explains the completion code.
GBL	SCOPE(ALL)	Indicates that the response line is for the global update.
LU	TRAN	APPC LU name associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.
MBR	N/A	The IMSplex member that built the output line. The IMS identifier of the IMS for which the transaction information is displayed. The IMS identifier is always returned.
NODE	TRAN	Node name of static node associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.
TMEM	TRAN	OTMA tmember name associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.
TPIP	TRAN	OTMA tpipe name associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.
TRAN	N/A	The transaction name. The transaction name is always displayed.
USER	TRAN	User name of dynamic user associated with the transaction conversation that caused the update to fail with a completion code of C'1A'. This information can be used to exit the conversation, before attempting the update again.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 547. Return and reason codes for the UPDATE TRAN command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See Table 548 on page 1354 for details.
X'00000004'	X'00002008'	Invalid number of keywords. Either a SET, START, or STOP keyword is required.
X'00000008'	X'0000200C''	No resources were found to be updated. The resource names specified might be invalid or there were no resources that match the filter specified. Confirm that the UPDATE TRAN command is issued with valid resources.
X'00000008'	X'00002040'	More than one filter value is specified on the UPDATE TRAN command. Confirm that only one of SET or START	STOP filters are specified on the command.
X'00000008'	X'00002044'	The UPDATE TRAN command is not processed because the same attribute value was specified for the START and STOP filters. The attribute "Q,SCHD" can be specified only on START or STOP but not both. For example, UPDATE TRAN START(Q) STOP(Q) is not valid but UPDATE TRAN START(Q) STOP(SCHD) is valid. Confirm that only one START	STOP attribute is specified on the command.
X'00000008'	X'00002048'	The UPDATE TRAN command is not processed because an invalid SET attribute is specified. Confirm that the correct SET attribute is specified on the command.
X'00000008'	X'0000204C'	The UPDATE TRAN command is not processed because a CLASS value specified is invalid. Confirm that the correct CLASS value is specified on the command.
X'00000008'	X'00002050'	The UPDATE TRAN command is not processed because the CPRI value specified is invalid. Confirm that the correct CPRI value is specified on the command.
X'00000008'	X'00002054'	The UPDATE TRAN command is not processed because the LCT (limit count) value specified is invalid. Confirm that the correct LCT value is specified on the command.
X'00000008'	X'00002058'	The UPDATE TRAN command is not processed because the LPRI value specified is invalid. Confirm that the correct LPRI value is specified on the command.
X'00000008'	X'0000205C'	The UPDATE TRAN command is not processed because the MAXGN value specified is invalid. Confirm the correct MAXRGN value is specified on the command.
X'00000008'	X'00002060'	The UPDATE TRAN command is not processed because the NPRI value specified is invalid. Confirm that the correct NPRI value is specified on the command.

Table 547. Return and reason codes for the UPDATE TRAN command (continued)

Return code	Reason code	Meaning
X'00000008'	X'00002064'	The UPDATE TRAN command is not processed because the PARLIM value is invalid. Confirm that the PARLIM value is specified on the command.
X'00000008'	X'00002068'	The UPDATE TRAN command is not processed because the PLCT value is invalid. Confirm that the correct PLCT value is specified on the command.
X'00000008'	X'00002100'	CMTMODE(MULT) mutually exclusive with WFI(Y).
X'00000008'	X'00002101'	CONV(Y) mutually exclusive with CMTMODE(MULT).
X'00000008'	X'00002103'	CONV(N) mutually exclusive with SPASZ>0 and SPATRUNC.
X'00000008'	X'00002104'	CONV(Y) mutually exclusive with RECOVER(N).
X'00000008'	X'00002105'	CONV(Y) requires SPASZ and SPATRUNC.
X'00000008'	X'00002108'	Invalid EDITRTN name.
X'00000008'	X'0000210A'	Invalid EMHBSZ. One of the following situations occurred: • The EMHB size specified is either greater than the maximum size, which is 30720 • The EMHB size specified plus the length of the X'5901' log record prefix is greater than the log buffer size
X'00000008'	X'0000210C'	FP(E) mutually exclusive with EDITRTN.
X'00000008'	X'0000210E'	FP(E) or FP(P) mutually exclusive with MSC keyword MSNAME or SIDR and SIDL.
X'00000008'	X'0000210F'	FP(E) or FP(P) mutually exclusive with MSGTYPE(MULTSEG).
X'00000008'	X'00002110'	FP(N) mutually exclusive with EMHBSZ > 0.
X'00000008'	X'00002111'	FP(E) or FP(P) mutually exclusive with RECOVER(N).
X'00000008'	X'00002112'	FP(E) or FP(P) mutually exclusive with RESP(N).
X'00000008'	X'00002116'	INQ(N) mutually exclusive with RECOVER(N).
X'00000008'	X'00002119'	MSC keyword MSNAME or SIDL/SIDR mutually exclusive with application program defined as Fast Path exclusive (FP(E)) associated with this transaction.
X'00000008'	X'0000211A'	Invalid MSNAME name.
X'00000008'	X'0000211B'	MSNAME mutually exclusive with SIDL and SIDR.
X'00000008'	X'0000211D'	MAXRGN>0 mutually exclusive with PARLIM(65535).
X'00000008'	X'0000211E'	MAXRGN>0 mutually exclusive with SERIAL(Y).
X'00000008'	X'00002121'	PARLIM value mutually exclusive with SERIAL(Y).
X'00000008'	X'00002125'	If REMOTE(Y) is specified, the SIDR value must be a remote SYSID and the SIDL value must be a local SYSID. If neither the MSNAME keyword nor the SIDR and SIDL keywords are specified explicitly, the SIDR and SIDL values from the existing transaction definition are used. If REMOTE(N) is specified, the SIDR value must be equal to the SIDL value. If the SIDR and SIDL values are not specified explicitly, the values from the existing transaction definition are used. The MSNAME keyword cannot be specified with REMOTE(N).

Table 547. Return and reason codes for the UPDATE TRAN command (continued)

Return code	Reason code	Meaning
X'00000008'	X'00002126'	Invalid SIDL value.
X'00000008'	X'00002127'	SIDL/SIDR must be specified as a pair. Either SIDL was specified alone or SIDR was specified alone.
X'00000008'	X'00002128'	Invalid SIDR value.
X'0000000C'	X'00003000'	Command was successful for some of the resources. The command output contains a line for each resource, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), output lines are only returned for resources with nonzero completion codes. See Table 548 for details.
X'0000000C'	X'00003004'	Command was not successful for any of the resources. The command output contains a line for each resource, accompanied by its completion code. See Table 548 for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.
X'00000010'	X'00004024'	No Fast Path defined, FP(E), FP(P), or EMHBSZ >0 invalid.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004310'	Storage could not be obtained for the Transaction Input edit routine table. A cold start is required to fix this error.
X'00000010'	X'00004314'	The Transaction Input edit routine could not be loaded.
X'00000010'	X'00004318'	A new Transaction Input edit routine could not be added. The maximum of 255 routines has already been reached.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.
X'00000014'	X'00005010'	Latch could not be obtained.
X'00000014'	X'000050FF'	The UPDATE TRAN or UPDATE TRANDESC command processing terminated because of an internal error.

Errors unique to the processing of the UPDATE TRAN command are returned as a completion code. A completion code is returned for each action against an individual resource. The completion codes in the following table might be returned on the UPDATE TRAN command.

Table 548. Completion codes for the UPDATE TRAN command

Completion code	Completion code text	Meaning
0		Command completed successfully for transaction.

Table 548. Completion codes for the UPDATE TRAN command (continued)

Completion code	Completion code text	Meaning
10	NO RESOURCES FOUND	Transaction name is invalid, or the wildcard parameter specified does not match any resource names.
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this transaction. This could also mean this command, if the resource is specified by more than one specific or wildcard parameter. Or, the transaction is updating the program name and another command is in progress for that program.
19	CMTMODE=MULT/ WFI=Y CONFLICT	Transaction update failed because commit mode multiple CMTMODE(MULT) option conflicts with the wait-for-input WFI(Y) option.
1A	IN CONVERSATION	Transaction is in conversation. The conversation ID and terminal in conversation are returned separately. The terminal can be a static node, node and user, dynamic user, APPC luname, or OTMA tmember and tpipe.

Suggested actions: Terminate the conversation. |
1B	CONV=Y/ CMTMODE=MULT CONFLICT	Transaction update failed because the conversation CONV(Y) option conflicts with the commit mode multiple CMTMODE(MULT) option.
1E	CONV=N/SPASZ/ SPATRUNC CONFLICT	Transaction update failed because the non-coversation CONV(N) option conflicts with the SPA size or SPA truncation option.
1F	CONV=Y/RECOVER=N CONFLICT	Transaction update failed because the conversation CONV(Y) option conflicts with the nonrecoverable RECOVER(N) option.
2F	FP=E/BMPTYPE=Y CONFLICT	Transaction update failed because Fast Path exclusive FP(E) option conflicts with program defined as batch BMPTYPE(Y).
30	QUEUE-ONLY TRANSACTION	The UPDATE TRAN command is invalid for the resource because the transaction is a queue-only transaction.
34	NOT ALLOWED FOR A CPIC TRAN	The UPDATE TRAN command is invalid for the resource because the transaction is a CPIC transaction.
35	FP=E/EDITRTN CONFLICT	Transaction update failed because Fast Path exclusive FP(E) option conflicts with the edit routine EDITRTN.
36	FP=E/FP=N PGM CONFLICT	Transaction update failed because Fast Path exclusive FP(E) option conflicts with program defined as non-Fast Path FP(N).
38	NOT ALLOWED FOR A BMP	The UPDATE TRAN command is invalid for the resource because the PSB associated with the transaction is a BMP.

Table 548. Completion codes for the UPDATE TRAN command (continued)

Completion code	Completion code text	Meaning
3A	FP=E OR P/MSC KEYWORD CONFLICT	Transaction update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with the MSC MSNAME, SIDR/SIDL option.
3B	FP/ MSGTYPE=MULTSEG CONFLICT	Transaction update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with message type multiple segment MSGTYPE(MULTSEG) option.
3C	MAXRGN/ PARLIM=65535 CONFLICT	Transaction update failed because maximum region count MAXRGN value conflicts with the parallel limit count PARLIM value 65535, which means parallel scheduling is disabled. MAXRGN > 0 is not allowed with PARLIM(65535).
3E	FP=N/FP=E PGM CONFLICT	Transaction update failed because non-Fast Path FP(N) option conflicts with program defined as Fast Path exclusive FP(E).
3F	FP=P/BMPTYPE=Y CONFLICT	Transaction update failed because Fast Path potential program FP(P) conflicts with the program defined as batch BMPTYPE(Y).
40	PARLIM/ SCHDTYPE=SERIAL CONFLICT	The PARLIM cannot be changed for the resource because the PSB associated with the transaction is defined as does not have parallel scheduling.
41	FP=E OR P/RECOVER=N CONFLICT	Transaction update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with nonrecoverable RECOVER(N) option.
42	FP=E OR P/RESP=N CONFLICT	Transaction update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with response mode RESP(N) option.
44	TRANSACTION BUSY	The UPDATE TRAN command cannot be processed for the resource because the transaction is currently being scheduled.
45	INVALID SIDR VALUE	The UPDATE TRAN command could not be completed for the resource because the SID number is invalid.
49	INQ=N/RECOVER=N CONFLICT	Transaction update failed because non-inquiry INQ(N) option conflicts with nonrecoverable RECOVER(N) option.
4A	IN USE	Transaction is in use. Queuing is in progress, either terminal input or a program-to-program switch.
4F	INVALID MAXRGN VALUE	Maximum region MAXRGN value is invalid.

Table 548. Completion codes for the UPDATE TRAN command (continued)

Completion code	Completion code text	Meaning
5E	MAXRGN GT 0/SERIAL=Y CONFLICT	Transaction update failed because non-zero maximum region value conflicts with serial SERIAL(Y) option.
61	DFSBCB STORAGE ERROR.	DFSBCB storage could not be obtained.
6A	FP=P/FP=E PGM CONFLICT	Transaction to be updated as Fast Path potential FP(P) conflicts with program already defined as Fast Path exclusive FP(E).
6B	PARLIM/SERIAL=Y CONFLICT	Transaction update failed because the parallel limit PARLIM value conflicts with the serial SERIAL(Y) option.
6D	INVALID PROGRAM NAME	Program name specified is invalid.
73	PROGRAM SCHEDULED	Program is scheduled.
79	REMOTE/SIDR/SIDL/ MSNAME CONFLICT	Transaction update failed because there is a conflict between the REMOTE value and the MSNAME keyword or the SIDR and SIDL values.
85	SUSPENDED	Transaction is on the suspend queue.
87	TRAN QUEUEING	Transaction has messages queued (non-shared-queues environment).
89	TRAN SCHEDULED	Transaction is scheduled.
90		UPD TRAN command did not complete because of an internal error.
99	NOT INITIALIZED	Transaction update failed because the transaction was not successfully initialized. QUERY TRAN STATUS(NOTINIT) displays the reason the transaction is not initialized, for example the program does not exist. Correct this problem and issue an UPDATE TRAN START(SCHD,Q) command to initialize the transaction.
9B	FASTPATH TRAN NOT SUPPORTED	Transaction update failed because the transaction is Fast Path exclusive FP(E).
B3	TRAN ELIGIBLE FOR SCHEDULING	Transaction is eligible for scheduling and cannot be updated. You might need to stop the transaction with the UPDATE TRAN STOP(Q,SCHD) command before attempting the UPDATE again.
B5	ROUTING CODE ALREADY EXISTS	Transaction update failed because Fast Path exclusive FP(E) option conflicts with routing code that already exists by that transaction name.
BF	FP=E/CONV KEYWORD CONFLICT	Fast Path exclusive FP(E) is mutually exclusive with any conversation keyword, including CONV(Y), SPASZ, and SPATRUNC. An FP exclusive transaction cannot be defined as conversational.

Table 548. Completion codes for the UPDATE TRAN command (continued)

Completion code	Completion code text	Meaning
145	MESSAGE IN PROGRESS ACROSS LINK	Transaction update failed because a message for the transaction is in progress across the MSC link.
1D7	IMPORT CHANGE LIST ERROR	The resource is in the process of being imported from the change list in the IMSRSC repository or was not successfully imported from the change list. The resource cannot be updated until it is successfully imported from the repository.

Examples

The following are examples of the UPDATE TRAN command:

Example 1 for UPDATE TRAN command

TSO SPOC input:
```
UPDATE TRAN NAME(BADNAME,AOBMP,APOL17,CPI%,BAD*) SET(WFI(Y))
```

TSO SPOC output:
```
Response for: UPDATE TRAN NAME(BADNAME,AOBMP,APOL17,CPI%,BAD*)
Trancode MbrName    CC CCText
AOBMP    IMS1        0
APOL17   IMS1       19 CMTMODE=MULT/WFI=Y CONFLICT
BAD*     IMS1       10 NO RESOURCES FOUND
BADNAME  IMS1       10 NO RESOURCES FOUND
CPI1     IMS1        0
CPI2     IMS1        0
CPI3     IMS1        0
CPI4     IMS1        0
```

OM API input:
```
CMD(UPDATE TRAN NAME(BADNAME,AOBMP,APOL17,CPI%,BAD*) SET(WFI(Y)))
```

OM API output:
```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.312 00:07:49.667460</statime>
<stotime>2006.312 00:07:49.668362</stotime>
<staseq>BFAC164BEAE846C0</staseq>
<stoseq>BFAC164BEB20A400</stoseq>
<rqsttkn1>USRT011 10160749</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS     </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
```

```
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1   </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>TRAN          </kwd>
<input>UPDATE TRAN NAME(BADNAME,AOBMP,APOL17,CPI%,BAD*,AOBMP)
SET(WFI(Y)) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="TRAN" llbl="Trancode" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="GBL" llbl="Global" scope="GBL" sort="d" key="2" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="y" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="CONVID" llbl="ConvID" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="NODE" llbl="NodeName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="USER" llbl="UserName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="LU" llbl="LUName" scope="LCL" sort="n" key="0"
 scroll="yes" len="24" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMEM" llbl="TMember" scope="LCL" sort="n" key="0"
 scroll="yes" len="16" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TPIP" llbl="TPipe" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>TRAN(BADNAME ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>TRAN(AOBMP   ) MBR(IMS1) CC(   0) </rsp>
<rsp>TRAN(APOL17  ) MBR(IMS1) CC(  19) CCTXT(CMTMODE=MULT/WFI=Y CONFLICT) </rsp>
<rsp>TRAN(CPI1    ) MBR(IMS1) CC(   0) </rsp>
<rsp>TRAN(BAD*    ) MBR(IMS1) CC(  10) CCTXT(NO RESOURCES FOUND) </rsp>
<rsp>TRAN(CPI2    ) MBR(IMS1) CC(   0) </rsp>
<rsp>TRAN(CPI3    ) MBR(IMS1) CC(   0) </rsp>
<rsp>TRAN(CPI4    ) MBR(IMS1) CC(   0) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: Update several transactions to be wait-for-input. The update
succeeded for several transactions, as shown by the completion code 0. The update
failed for transaction APOL17 with completion code 19, since the WFI(Y) attribute
conflicts with the CMTMODE(MULT) attribute already defined for APOL17. The
update fails for transaction BADNAME and for parameter BAD* with completion
code 10, since transaction BADNAME does not exist and no transaction name
starts with BAD.

Example 2 for UPDATE TRAN command

TSO SPOC input:
```
UPDATE TRAN NAME(APOL12) START(SCHD) OPTION(AFFIN)
```

TSO SPOC output:

```
Trancode MbrName   CC
APOL12   IMS1        0
```

Explanation: Transaction APOL12 is started with affinity.

Related concepts:

How to interpret CSL request return and reason codes (System Programming APIs)

Related tasks:

Updating runtime transaction resource and descriptor definitions with the UPDATE command (System Definition)

Related reference:

/LOCK command (Commands)

Command keywords and their synonyms (Commands)

Commit Verify exit routine (Exit Routines)

UPDATE TRANDESC command

Use the UPDATE TRANDESC command to update transaction descriptors. A descriptor is a model that can be used to create descriptors or resources.

Updating a descriptor changes only those attributes explicitly specified on the UPDATE command. Attributes not specified retain their existing values. Any transaction resource or descriptor can be created using this descriptor as a model, by specifying the CREATE LIKE(DESC(*descriptor_name*)) command. Any descriptor or resource that was already created using this descriptor is not updated.

Subsections:

Environment

The following table lists the environments (DB/BC, DBCTL, and DCCTL) in which you can use the commands and keywords.

Table 549. Valid environments for the UPDATE TRANDESC command and keywords

Command / Keywords	DB/DC	DBCTL	DCCTL
UPDATE TRANDESC	X		X

Command / Keywords	DB/DC	DBCTL	DCCTL
NAME	X		X
OPTION	X		X
SET	X		X

Syntax
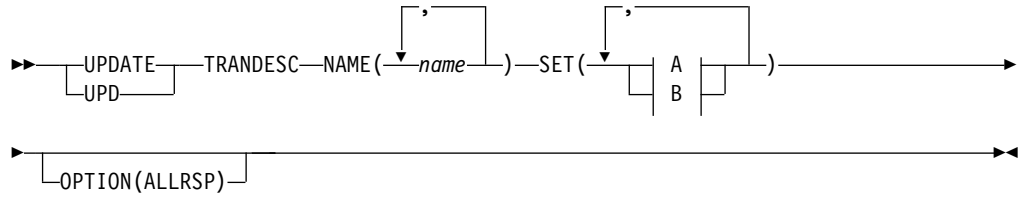
A:

```
|--+-- AOCMD(--+--CMD---+--)--+-----------------------------------------------|
   |           |--N-----|     |
   |           |--TRAN--|     |
   |           '--Y-----'     |
   |                          |
   |-- CLASS(class)-----------|
   |-- CMTMODE(--+--MULT--+--)-|
   |             '--SNGL--'    |
   |-- CONV(--+--N--+--)-------|
   |          '--Y--'          |
   |-- DCLWA(--+--N--+--)------|
   |           '--Y--'         |
   |-- DEFAULT(Y)-------------|
   |-- DIRROUTE(--+--N--+--)---|
   |              '--Y--'      |
   |-- EDITRTN(name)----------|
   |-- EDITUC(--+--N--+--)-----|
   |            '--Y--'        |
   |-- EMHBSZ(size)-----------|
   |-- EXPRTIME(seconds)------|
   |-- FP(--+--E--+--)--------|
   |        |--N--|           |
   |        '--P--'           |
   |-- INQ(--+--N--+--)-------|
   |         '--Y--'          |
   |-- LCT(value)------------|
   |-- LPRI(value)-----------|
   |-- MAXRGN(number)--------|
   |-- MSGTYPE(--+--MULTSEG--+--)-|
   |             '--SNGLSEG--'    |
   |-- MSNAME(name)----------|
   |-- NPRI(value)-----------|
   |-- PARLIM(value)---------|
   |-- PGM(name)------------|
   |-- PLCT(value)-----------|
   |-- PLCTTIME(hundredths of seconds)--|
   '-- RECOVER(--+--N--+--)---'
                 '--Y--'
```

B:

```
|--+-- REMOTE(--+--N--+--)---+--------------------------------------------------|
   |            '--Y--'      |
   |-- RESP(--+--N--+--)-----|
   |          '--Y--'        |
   |-- SEGNO(number)--------|
   |-- SEGSZ(size)----------|
   |-- SERIAL(--+--N--+--)---|
   |            '--Y--'      |
   |-- SIDL(localsysid)-----|
   |-- SIDR(remotesysid)----|
   |-- SPASZ(size)----------|
   |-- SPATRUNC(--+--R--+--)-|
   |              '--S--'    |
   |-- TRANSTAT(--+--N--+--)-|
   |              '--Y--'    |
   '-- WFI(--+--N--+--)------'
             '--Y--'
```

Keywords

The following keywords are valid for the UPDATE TRANDESC command:

NAME
> Specifies the 1-8 character name of the transaction descriptor. Wildcards can be specified in the name. The name is a repeatable parameter. If the NAME parameter specified is a specific or wildcard name, command responses are returned for all the descriptor names that are processed. For NAME(*), command responses are returned only for the descriptor names that resulted in an error. OPTION(ALLRSP) can be specified with NAME(*) to obtain the command responses for all the descriptor names that are processed.

OPTION
> Specifies additional functions to be performed along with the command.

> **ALLRSP**
>> Indicates that the response lines are to be returned for all descriptors that are processed on the command. The default action is to return response lines only for the descriptors that resulted in an error. It is only valid with NAME(*). ALLRSP is ignored for other NAME values.

SET
> Specifies the attribute values to be changed. A transaction must have the same characteristics in all systems where it is defined when it is shared. These characteristics include:
> - Nonconversational or conversational
> - SPA size if conversational
> - Single-segment or multi-segment messages
> - Non-inquiry or inquiry
> - Recoverable or nonrecoverable

> **AOCMD**
>> Specifies that the AOI option indicating whether the transaction can issue the type-1 AOI CMD call or the type-2 AOI ICMD call. If AOCMD is defined as CMD, TRAN, or Y, and the AOI1 execute parameter is defined as AOI1=N, no authorization checking is done, and the transaction is permitted to issue CMD and ICMD calls.

>> **CMD**
>>> Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, authorization checking is based on which transactions can issue a particular command. In this case, the commands (or first three characters of the commands) need to be defined to RACF or equivalent product as a user. The type-1 AOI transactions must be defined as profiles under the TIMS class, and for each transaction, the commands it can issue must be specified. Defining AOCMD(CMD) requires you to create fewer user IDs than you need to create for the AOCMD(TRAN) definition. However, defining AOCMD(CMD) requires you to create or modify a larger number of resource profiles.

>> **N**
>>> Indicates that the transaction is not permitted to issue AOI type-1 CMD calls. The transaction is permitted to issue AOI type-2 ICMD calls.

>> **TRAN**
>>> Indicates that the transaction is permitted to issue type-1 AOI CMD

calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the transaction code is used for authorization. The first authorization check results in the accessor environment element (ACEE) being built. This environment is kept for use by future authorization checks. The type-1 AOI transaction needs to be defined to RACF or equivalent product as a user. The transactions will then be specified on RACF PERMIT statements for each command they are allowed to issue from a type-1 AOI transaction. Specifying AOI transactions as users to RACF might conflict with the name of a user already defined to RACF. If this occurs, then either the transaction name or the existing user name needs to be changed.

Y   Indicates that the transaction is permitted to issue type-1 AOI CMD calls and type-2 AOI ICMD calls. If the AOI1 execute parameter is defined as C, R, or A, the user ID or program name is used for authorization. For some environments, if a Get Unique call has not yet happened, then the program name is used for authorization.

CLASS

Specifies the transaction class, which is an attribute used to select a transaction for scheduling. A transaction can be scheduled if there is a message processing region available for that class. The value can be a number from 1 to 999. This value must not exceed the value given (by specification or default) on the MAXCLAS= keyword of the IMSCTRL macro.

Define CPI-C transactions with a different message class from that used for non-CPI-C transactions. IMS handles all CPI-C transactions as priority zero within the transaction class.

CMTMODE

Specifies when database updates and non-express output messages are committed. This operand affects emergency restart.

MULT
    Database updates and non-express output messages are committed only when the application program terminates normally, when the processing limit count has been reached, or, in the case of a pseudo-WFI dependent region, when there are no more messages on the queue. For example, if five transactions are processed during a single schedule of a program, all five are committed only when the fifth one is completed and the program terminates. Until a transaction has been committed, locks for updated database records are not released and non-express output messages are not queued for output. If an application ends abnormally before committing its messages, emergency restart requeues all the messages that were processed within the commit scope and makes them available for reprocessing.

    If the transaction results in the application calling an external subsystem, such as DB2, the Commit Verify exit provided by the external subsystem can determine whether CMTMODE(MULT) is supported. See documentation under the Commit Verify exit routine in *IMS Version 14 Exit Routines*.

SNGL
    Database updates and non-express output messages are committed when the application program completes processing each transaction. IMS invokes commit processing either when the application program

requests the next message (issues a GU to the IO-PCB), or when the
application program terminates. If an application ends abnormally
before committing its messages, emergency restart requeues all the
messages that were processed within the commit scope and makes
them available for reprocessing. If an application ends abnormally
before committing its message, emergency restart requeues the message
that was in process before the abend and makes it available for
reprocessing.

Keyword combination rules include the following:
- CONV(Y) and CMTMODE(MULT) are mutually exclusive.
- WFI(Y) and CMTMODE(MULT) are mutually exclusive.

CONV
Specifies the conversation option.

N   The transaction is not conversational.

Y   The transaction is conversational. The transaction message is destined
        for a conversational program. A conversational program processes
        transactions made up of several steps. A conversational program
        receives a message from a terminal, replies to the terminal, but saves
        the data from the transaction in a scratchpad area (SPA). When the
        person at the terminal enters more data, the program has the data it
        saved from the last message in the SPA, so it can continue processing
        the request without the person at the terminal having to enter the data
        again.

Keyword combination rules include the following:
- CONV(Y) requires SPASZ and SPATRUNC.
- CMTMODE(MULT) and CONV(Y) are mutually exclusive.
- INQ(Y) and CONV(Y) are mutually exclusive.
- RECOVER(N) and CONV(Y) are mutually exclusive.
- SPASZ and CONV(N) are mutually exclusive.
- SPATRUNC and CONV(N) are mutually exclusive.

DCLWA
Specifies the log write-ahead option.

N   If N is specified, IMS does not perform log write-ahead. Specify N if
        input message integrity and the consistency of output messages with
        associated database updates is not required. DCLWA does not apply to
        response mode or Fast Path input processing, and is ignored during
        IMS execution.

Y   If Y is specified, IMS performs log write-ahead for recoverable,
        nonresponse input messages and transaction output messages. This
        ensures the following:

   - A nonresponse input transaction is made recoverable across IMS
     failures, before IMS acknowledges receipt of the input.
   - Database changes are made recoverable before IMS sends associated
     output reply messages.
   - Information in the log buffers is written to the IMS log, before the
     associated input acknowledgment or output reply is sent to the
     terminal.

Define DCLWA(Y) for all VTAM terminal types.

DEFAULT(Y)
Specifies this descriptor as the default, which resets the existing default descriptor to DEFAULT(N). When a descriptor is created without the LIKE keyword, any attribute not specified on the CREATE command takes the value defined in the default descriptor. Only one descriptor can be defined as the default for a resource type. IMS defines transaction descriptor called DFSDSTR1, where all attributes are defined with the default value. Defining a user-defined descriptor to be the default overrides the current default descriptor. Since only one transaction descriptor can be the default at one time, only one transaction descriptor name can be specified with DEFAULT(Y).

DIRROUTE
Specifies the MSC directed routing option.

N   The application program processing a transaction is not informed of the system which originated the transaction. The name of the originating LTERM is placed in the I/O PCB.

Y   The application program processing a transaction is informed of the system which originated the transaction, if MSC directed routing is used in a multiple IMS system configuration. An MSNAME corresponding to a logical path back to the originating system is placed in the I/O PCB.

EDITRTN

Specifies the 1- to 8-character name of your transaction input edit routine that edits messages before the program receives the message. This name must be alphanumeric (A through Z, 0 through 9, #, $, and @). It must begin with an alphabetic character (A through Z, #, $, @). It cannot include a blank, comma, period, hyphen, or equal. It cannot include wildcard characters * or %.

The specified edit routine can either be an edit routine defined during the system definition process with the EDIT= parameter on the TRANSACT macro or it can be a new routine. If the routine is a new routine, the routine must reside in one of the RESLIB concatenated data sets. A maximum of 255 input edit routines are supported.

If the edit routine specified on the command is new to IMS, IMS attempts to load the routine from RESLIB. If the load fails, the command is rejected. If the edit routine specified is already defined to IMS, a decision is made whether to load a new copy of the routine or to use the existing copy. If the routine was generated into the system, but there are no transactions referencing the routine, IMS first attempts to load a new copy of the module from RESLIB. If the load is successful, the dynamic copy of the edit routine is used. The generated copy cannot be used again unless IMS is cold started. If the load of the dynamic routine fails, the generated copy is used. If the edit routine was generated into the system and other transactions reference it, the generated copy of the module is used.

EDITRTN is used for a Fast Path potential transaction when the transaction is routed to IMS.

For input from LU 6.2 devices, the user edit exit routine DFSLUEE0 is called instead of the transaction input edit routine specified in EDITRTN.

FP(E) and EDITRTN are mutually exclusive.

A zero value can be specified for the EDITRTN() parm (EDITRTN(0)) to remove an edit routine name from a transaction definition.

EDITUC
Specifies the edit to uppercase option.

N The input data is not translated to uppercase. It can consist of uppercase and lowercase characters as entered from the terminal.

Y The input data is to be translated to uppercase before it is presented to the processing program. If FP(E) or FP(P), the transaction is to be translated to uppercase before being presented to the edit/routing exit routine.

Specifying EDITUC(Y) for VTAM terminals prevents the transmission of embedded device control characters.

EMHBSZ

Specifies the EMH buffer size required to run the Fast Path transaction. This overrides the EMHL execution parameter. If EMHBSZ is not specified, the EMHL execution parameter value is used. The value can be a number from 12 to 30720.

Keyword combination rules include the following:
• EMHBSZ > 0 requires Fast Path to be defined.
• FP(N) and EMHBSZ>0 are mutually exclusive.

EXPRTIME
Specifies the elapsed time in seconds that IMS can use to cancel the input transaction. After a transaction is submitted to IMS, the transaction could be delayed for processing because of a stopped transaction or a potential system slow down. In that case, the user or client application could time out before the transaction is processed. When IMS eventually schedules and processes the transaction, the response message is no longer wanted. With the elapsed time specified for the transaction, IMS can flag the input transaction as expired so that the system does not waste CPU cycles to process the unwanted transaction.

The value can be a number, in seconds, which can range from 0 to 65535. The default is 0, which means that no expiration time is set for this transaction. The transaction expiration attribute is supported by all of the IMS TM interfaces.

Restriction: The transaction expiration checking is not performed at the GU time for Fast Path transactions, IMS conversational transactions, and program-to-program switch transactions.

FP Specifies the Fast Path option.

E The transaction is processed exclusively as Fast Path. The program must be defined as Fast Path exclusive.

N The transaction is not a candidate for Fast Path processing. The program must be defined as not Fast Path. When FP(N) is specified, any attempt to use Fast Path resources or commands might yield unpredictable results.

P The transaction is a potential candidate for Fast Path processing. Fast Path-potential transactions must be able to run under two applications: a Fast Path exclusive application and a non-Fast Path application. A Fast Path exclusive application should be defined to which this

transaction can be routed. Fast Path–potential transactions must be processed by a user edit/routing exit to determine whether the transaction is actually to be processed by IMS Fast Path. If it is to be processed by IMS Fast Path, the edit/routing exit routine associates the transaction with a routing code. This routing code identifies which Fast Path application program is to process the transaction.

The program defined by PGM() must not be defined as Fast Path exclusive.

In order to update a transaction from FP(E) to FP(N) or from FP(N) to FP(E) you must also update the transaction to point to a program that has the same FP() attribute. If you do not update the program attribute, the command fails because of a program conflict.

Keyword combination rules include the following:
- EDITRTN and FP(E) are mutually exclusive.
- EMHBSZ>0 and FP(N) are mutually exclusive.
- FP(E) and FP(P) require Fast Path to be defined.
- MSGTYPE(MULTSEG) and FP(E) are mutually exclusive.
- MSGTYPE(MULTSEG) and FP(P) are mutually exclusive.
- MSNAME and FP(E) are mutually exclusive.
- RECOVER(N) and FP(E) are mutually exclusive.
- RECOVER(N) and FP(P) are mutually exclusive.
- RESP(N) and FP(E) are mutually exclusive.
- RESP(N) and FP(P) are mutually exclusive.
- SIDL and FP(E) are mutually exclusive.
- SIDR and FP(E) are mutually exclusive.

INQ

Specifies the inquiry option.

N This is not an inquiry transaction.

Y This is an inquiry transaction. If INQ(Y) is specified, you can also specify whether this transaction should be recovered during an IMS emergency or normal restart using the RECOVER() parameter.

This option should be specified only for those transactions that, when entered, do not cause a change in any database. Programs are prohibited from issuing ISRT, DLET, or REPL calls to a database when scheduled to process a transaction defined as INQ(Y).

An application program cannot do an SQL INSERT, DELETE, or UPDATE when the IMS transaction is defined with INQ(Y).

Keyword combination rules include the following:
- CONV(Y) and INQ(Y) are mutually exclusive.
- RECOVER(N) and INQ(N) are mutually exclusive.

LCT

Specifies the limit count. This is the number that, when compared to the number of input transactions queued and waiting to be processed, determines whether the normal or limit priority value is assigned to this transaction. The value can be a number from 1 to 65 535. The default is 65 535.

The limit count value is ignored for a transaction processed by a BMP.

The limit count value is ignored in a shared queues environment.

LPRI

Specifies the limit priority. This is the scheduling priority to which this transaction is raised when the number of input transactions enqueued and waiting to be processed is equal to or greater than the limit count value. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The value can be a number from 0 through 14.

When the limit priority is used and the scheduling priority is raised to the limit priority, the priority is not reduced to the normal priority until all messages enqueued for this transaction name are processed. If you do not want the limit priority for this transaction, define equal values for the normal priority and limit priority, and a limit count of 65535.

When a transaction is processed exclusively by a batch message program (BMP), define the limit priority as 0. If the program specified by PGM() is defined with a program type of batch, the current priority is forced to be 0. However, a batch message processing region (BMP) can process transactions with current scheduling priorities other than 0.

This priority also controls the priority of messages created by this transaction and sent to a destination in a remote system. See also the discussion on MSC priorities under the NPRI definition.

The limit priority value is ignored for a transaction processed by a BMP.

The limit priority value is ignored in a shared-queues environment.

MAXRGN

Specifies the maximum region count. This limits the number of message processing program (MPP) regions that can be concurrently scheduled to process a transaction. When the number of MPP regions is not limited, one transaction might monopolize all available regions. The value can be a number from 0 to the value specified on the MAXPST= control region parameter. MAXRGN(0) means that no limit is imposed.

If you define the application program with a scheduling type of SERIAL, omit the MAXRGN keyword or define it as 0.

The following keyword combinations are mutually exclusive:
- PARLIM(65535) and MAXRGN value greater than 0
- SERIAL(Y) and MAXRGN value greater than 0

MSGTYPE

Specifies the message type (single segment or multiple segment). It specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing.

If MSC-directed routing is used in a multiple IMS system configuration, IMS does not ensure that both the message and the transaction destined to process that message are either single segment or multiple segments.

MULTSEG

The incoming message can be more than one segment in length. It is

not eligible for scheduling to an application program until an end-of-message indication is received, or a complete message is created by MFS.

SNGLSEG

The incoming message is one segment in length. It becomes eligible for scheduling when the terminal operator indicates end-of-segment.

Keyword combination rules include the following:
- FP(E) and MSGTYPE(MULTSEG) are mutually exclusive.
- FP(P) and MSGTYPE(MULTSEG) are mutually exclusive.
- RESP(Y) and MSGTYPE(MULTSEG) are mutually exclusive.

MSNAME

Specifies the one- to eight-character name of the logical link path in a multiple IMS system configuration (MSC). A logical link path is a path between any two IMS systems. The IMS systems are identified by the remote system ID and the local system ID associated with the logical link path. The remote system ID identifies the system in which messages using this path are to be processed. The local system ID identifies this system. For an UPDATE TRAN command that is changing a transaction to a remote transaction, or changing the MSC path, the new MSNAME must already be defined.

Keyword combination rules include the following:
- FP(E) and MSNAME are mutually exclusive.
- SIDL and MSNAME are mutually exclusive.
- SIDR and MSNAME are mutually exclusive.

NPRI

Specifies the normal scheduling priority. The scheduling priority is an attribute used to select a transaction for scheduling. A transaction of higher priority is scheduled before a lower priority one, if they are defined with the same class. The normal priority is assigned to the transaction as the scheduling priority when the number of input transactions enqueued and waiting to be processed is less than the limit count value. The value can be a number from 0 through 14. The default is 1.

This priority also controls the priority of messages created by this transaction and sent to a destination in a remote system.

When a transaction is processed exclusively by a batch message program (BMP), code the normal priority as 0.

When a transaction is processed exclusively by a batch message program (BMP), define the limit priority as 0. If the application program specified by PGM() is defined with a program type of batch, the current priority is forced to be 0. However, a batch message processing region (BMP) can process transactions with current scheduling priorities other than 0.

For remote transactions, the priority used to send the transaction to the processing system, which is termed *the MSC link message priority*. The three MSC link message priority groups are:
- Low
- Medium
- High

The low priority group consists of primary requests in the input terminal system. This group is assigned remote transaction priorities from 0 to 6. The medium group consists of secondary requests, responses, primary requests in an intermediate system, and primary requests in the input terminal system. This group is assigned a remote transaction priority of 7. The high group consists of primary requests in the input terminal system. Messages in this group are assigned remote transaction priorities from 8 to 14. Within each group, messages have a priority based on the current priority value of the transaction or remote transaction in the input terminal system for primary requests, and on the latest processing system for secondary requests and responses.

In an MSC configuration, the transaction priority determines the priority used to send messages inserted by this transaction across an MSC link. If the transaction inserts multiple messages to the same destination (for example, pages to a printer) and these messages must be sent in the order inserted, the normal and limit priority values should be the same. If the normal and limit priority values are not identical, messages inserted at a higher priority than previously inserted messages could arrive at their destination first. (This restriction does not apply to multiple segments of the same message.)

The normal priority value is ignored for a transaction processed by a BMP.

PARLIM

Specifies the parallel processing limit count. This is the maximum number of messages that can currently be queued, but not yet processed, by each active message region currently scheduled for this transaction. This is the threshold value to be used when the associated application is defined with a scheduling type of parallel. An additional region is scheduled whenever the current transaction enqueue count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction.

The value can be a number from 0 to 32767 or 65535. PARLIM(0) indicates that any input message can cause a new region to be scheduled because the scheduling condition is always met (the number of messages is greater than zero). If you specify PARLIM(0), you should specify a MAXRGN value to limit the number of regions that can be scheduled to process a particular transaction. PARLIM(65535) means that parallel processing is disabled and IMS allows the transaction to be scheduled in only one region at a time.

The value specified for PARLIM applies to message processing programs (MPPs) only; it is not supported for batch message processing programs (BMPs).

If you define the application as SERIAL or the scheduling type as SERIAL, define PARLIM(65535).

In a shared-queues environment (when the scheduling type is PARALLEL), any PARLIM value other than 65535 causes a new region to be scheduled whenever the successful consecutive GU count exceeds the PARLIM value multiplied by the number of regions currently scheduled for this transaction. For shared queues environments, the successful consecutive GU count is used instead of the queue count. New regions continue to be scheduled up to the maximum number of regions specified on MAXRGN.

Keyword combination rules include the following:

- MAXRGN>0 and PARLIM(65535) are mutually exclusive.
- SERIAL(Y) and PARLIM value 0 - 32767 are mutually exclusive.

PGM

Specifies the name of the application program associated with the transaction. The program must exist unless the transaction is defined as REMOTE(Y).

PLCT

Specifies the processing limit count. This is maximum number of messages sent to the application program by the IMS for processing without reloading the application program. The value must be a number from 0 through 65535. PLCT(0) means the maximum number of messages sent to the application is one and the application program is reloaded before receiving a subsequent message. PLCT(65535) means that no limit is to be placed upon the number of messages processed at a single program load. Values 1 through 65535 are eligible for quick reschedule processing.

The value is used to determine how many messages an application program is allowed to process in a single scheduling cycle. When the application program requests, and receives, the number of messages indicated, any subsequent requests result in one of two things.

1. IMS indicates "no more messages exist" if any of the following conditions is true
   - The region is not an MPP.
   - The currently scheduled mode is not CMTMODE(SNGL).
   - Equal or higher priority transactions are enqueued for the region.

   IMS might, in fact, have other messages enqueued for the application program. It is the responsibility of the application program to terminate when it receives an indicator that no more messages are available. Termination of the application program makes the region it occupied available for rescheduling. This feature makes it possible for IMS to enable scheduling of higher priority transactions that entered the system while the previous transactions were in process. In addition, if any equal-priority transactions are enqueued, they become eligible for scheduling on a first-in, first-out (FIFO) basis.

2. The region goes through quick reschedule and returns the next message to the application of all of the following conditions are true.
   - The region is an MPP.
   - The transaction is CMTMODE(SNGL).
   - No equal or higher transactions are enqueued.
   - Messages are still enqueued for the application.

PLCTTIME

Specifies the processing limit count time. This is the amount of time (in hundredths of seconds) allowable to process a single transaction (or message). The number specifies the maximum CPU time allowed for each message to be processed in the message processing region.

Batch Message Programs (BMPs) are not affected by this setting.

The value can be a number, in hundredths of seconds, that can range from 1 to 6553500. A value of 6553500 means no time limit is placed on the application program.

If Fast Path is used this specifies, for a given transaction name, the amount of time (in hundredths of seconds) the program is allowed to process a single transaction message. The time represents real time that elapses during transaction processing (not accumulated task time). Real time is used because the input terminal is in response mode and cannot enter another transaction until the response is sent. In this case PLCT() is ignored.

The value controls application program looping. You are not required to optimize the value for program-transaction execution time. However, the time value assigned should not be less than the expected per-transaction execution time. If the scheduled application program exceeds the product of PLCTTIME() and PLCT(), the application program ends abnormally. If an IMS STIMER value of 2 is specified on the DFSMPR procedure, the region does not end abnormally until completion of the DL/I call.

The application program must not use STIMER timer services. IMS uses STIMER timer services to time the execution of transactions. If an application program issues an MVS STIMER macro, it cancels the STIMER timer services set by IMS. Use the STIMERM macro instead for application program timer requests.

RECOVER
Specifies the recovery option.

N   The transaction should not be recovered.

Y   The transaction should be recovered during an IMS emergency or normal restart.

Keyword combination rules include the following:
* CONV(Y) and RECOVER(N) are mutually exclusive.
* FP(E) and RECOVER(N) are mutually exclusive.
* FP(P) and RECOVER(N) are mutually exclusive.
* INQ(N) and RECOVER(N) are mutually exclusive.

REMOTE
Specifies the remote option.

N   The transaction is not remote. The transaction is local and runs on the local system.

Y   The transaction is remote. The transaction runs on a remote system.

REMOTE(Y) requires MSNAME or SIDR and SIDL.

RESP
Specifies the response mode option.

N   The transaction is not response mode. For terminals specifying or accepting a default of OPTIONS=TRANRESP, input should not stop after this transaction is entered.

Y   The transaction is response mode. The terminal from which the transaction is entered is held and prevents further input until a response is received. For terminals specifying or accepting a default of OPTIONS=TRANRESP, no additional messages are to be allowed after this transaction is entered until this transaction sends a response message back to the terminal. Response mode can be forced or negated

by individual terminal definition. RESP(Y) is ignored during online processing for all terminals that do not operate in response mode.

Keyword combination rules include the following:
- FP(E) and RESP(N) are mutually exclusive.
- FP(P) and RESP(N) are mutually exclusive.
- MSGTYPE(MULTSEG) and RESP(Y) are mutually exclusive

SEGNO

Specifies the segment number. This is the maximum number of application program output segments that are allowed into the message queues per Get Unique (GU) call from the application program. The value can be a number from 0 through 65535. If SEGNO(0) is defined, the number of segments is not checked by the online system at execution time.

SEGSZ

Specifies the segment size. This is the maximum number of bytes allowed in any one output segment. The value can be a number from 0 through 65535. If SEGSZ(0) is defined, the segment size is not checked by the online system at execution time.

The maximum output message segment to an LU 6.2 device is 32767. If a transaction is expected to send output to an LU 6.2 device, the SEGSIZE parameter should be no greater than 32767. However, this is not enforced during processing of the command, because IMS cannot determine the device type for the message destination until output time.

SERIAL

Specifies the serial option.

N Messages for the transaction are not processed serially. Message processing can be processed in parallel. Messages are placed on the suspend queue after a U3303 pseudoabend. Scheduling continues until repeated failures result in the transaction being stopped with a USTOP.

Y Messages for the transaction are processed serially. U3303 pseudoabends do not cause the message to be placed on the suspend queue but rather on the front of the transaction message queue, and the transaction is stopped with a USTOP. The USTOP of the transaction is removed when the transaction or the class is started with a /START or UPD TRAN command.

Keyword combination rules include the following:
- MAXRGN>0 and SERIAL(Y) are mutually exclusive.
- PARLIM value 0 - 32767 and SERIAL(Y) are mutually exclusive.

SIDL

Specifies the system identification (SYSID) of the local system in a multiple-IMS system (MSC) configuration. The local system is the originating system to which responses are returned. The value can be a number from 1 to 2036. The local SYSID can be defined in any or all of the MSNAMEs or transactions.

The SIDL parameter is independent of the link type (CTC, MTM, TCP/IP, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement.

Keyword combination rules include the following:

- FP(E) and SIDL are mutually exclusive, unless SIDL and SIDR are specified as a pair and are equal to the local system ID of this IMS.
- MSNAME and SIDL are mutually exclusive.
- SIDL value must be defined to this IMS.

SIDR

Specifies the system identification (SYSID) of the remote system in a multiple-IMS system (MSC) configuration. The remote system is the system on which the application program executes. The value can be a number from 1 to 2036. The remote SYSID specified must also be defined for an MSNAME.

The SIDR parameter is independent of the link type (CTC, MTM, TCP/IP, VTAM) specified on the TYPE= keyword of the MSPLINK macro statement.

Keyword combination rules include the following:
- FP(E) and SIDR are mutually exclusive, unless SIDL and SIDR are specified as a pair and are equal to the local system ID of this IMS.
- MSNAME and SIDR are mutually exclusive.
- SIDR value must be defined to this IMS.

SPASZ

Specifies the scratchpad area (SPA) size, in bytes, for a conversational transaction. The value can be a number from 16 and 32767.

Keyword combination rules include the following:
- CONV(N) and SPASZ are mutually exclusive.

SPATRUNC

Specifies the scratchpad area (SPA) truncation option of a conversational transaction. This defines whether the SPA data should be truncated or preserved across a program switch to a transaction that is defined with a smaller SPA.

When a conversation initially starts, and when a program is switched, the SPATRUNC option is checked and set or reset as specified. When the option is set, it remains set for the life of the conversation, or until a program switch occurs to a transaction that specifies the option is to be Reset.

When a program switch occurs, the truncated data option for the new transaction is first checked, and that specification is set for the conversation and is used for the SPA inserted into the output message. If the option is not specified for the new transaction, the option currently in effect for the conversation is used.

S   IMS preserves all of the data in the SPA, even when a program switch is made to a transaction that is defined with a smaller SPA. The transaction with the smaller SPA does not see the truncated data, but when the transaction switches to a transaction with a larger SPA, the truncated data is used.

R   The truncated data is not preserved.

Keyword combination rules include the following:
- CONV(N) and SPATRUNC are mutually exclusive.

TRANSTAT

Specifies whether transaction level statistics should be logged for message driven programs. If Y is specified, transaction level statistics are written to the log in a X'56FA' log record.

N Transaction level statistics should not be logged.

Y Transaction level statistics should be logged.

The TRANSTAT keyword on the UPDATE TRAN or UPDATE TRANDESC command gives the user the ability to override the system default or the current value of the TRANSTAT parameter. If the TRANSTAT keyword is omitted on the UPDATE TRAN or UPDATE TRANDESC command, the current transaction level statistics setting is unchanged.

WFI

Specifies the wait-for input option. This attribute does not apply to Fast Path transactions, which always behave as wait-for-input transactions.

N This is not a wait-for-input transaction.

Y This is a wait-for-input transaction. A message processing or batch processing application program that processes WFI transactions are scheduled and invoked normally. If the transaction to be processed is defined as WFI, the program is allowed to remain in main storage after it has processed the available input messages. The QC status code (no more messages) is returned to the program if the PROCLIM count (PLCT) is reached; a command is entered to change the status of the scheduled transaction, database, program, or class; commands relating to the databases used by the transaction are entered, or IMS is terminated with a checkpoint shutdown.

Keyword combination rules include the following:
- MODE(MULT) and WFI(Y) are mutually exclusive.

Usage notes

If all the attributes specified by the UPDATE command are already defined for the descriptor, no update is actually made, no descriptors are quiesced, no log record is created, and a completion code of zero is returned. This avoids unnecessary overhead when no action needs to be taken.

Descriptors exist for the life of the IMS unless they are deleted using a DELETE command. The descriptors are recoverable across an IMS warm start or emergency restart. Descriptors are lost if IMS is cold started, unless cold start imports definitions that were exported while IMS was up. Each descriptor is updated individually. Individual updating does not work like online change where either all descriptors are updated or no descriptors are updated.

The UPDATE TRANDESC command can be issued only through the OM API. This command applies to DB/DC and DCCTL systems.

This command is not valid on the XRF alternate, RSR tracker, or FDBR region. The UPDATE TRANDESC commands are not valid if online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).

This command is recoverable.

If the descriptor being updated is the IMS-defined transaction descriptor (DFSDSTR1) the only attribute that can be changed is the DEFAULT attribute.

Output fields

The following table shows the UPDATE TRANDESC output fields. The columns in the table are as follows:

Short label
Contains the short label generated in the XML output.

Keyword
Identifies keyword on the command that caused the field to be generated. N/A appears for output fields that are always returned. *error* appears for output fields that are returned only in case of an error.

Meaning
Provides a brief description of the output field.

Table 550. Output fields for the UPDATE TRANDESC command

Short label	Keyword	Meaning
CC	N/A	Completion code.
CCTXT	*error*	Completion code text that briefly explains the meaning of the non-zero completion code.
DESC	TRANDESC	Transaction descriptor name.
ERRT	*error*	Error text with diagnostic information. Error text can be returned for a non-zero completion code and contains information that further explains the completion code.

Return, reason, and completion codes

The following table includes the return and reason codes and a brief explanation of the codes. The return or reason code returned for the command might also indicate an error from a CSL request.

Table 551. Return and reason codes for the UPDATE TRANDESC command

Return code	Reason code	Meaning
X'00000000'	X'00000000'	Command completed successfully. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), no output lines are returned. See the completion code table for details.
X'00000004'	X'00002008'	Invalid number of keywords. A SET keyword is required.
X'00000008'	X'00002100'	CMTMODE(MULT) mutually exclusive with WFI(Y).
X'00000008'	X'00002101'	CONV(Y) mutually exclusive with CMTMODE(MULT).
X'00000008'	X'00002102'	CONV(Y) mutually exclusive with INQ(Y).
X'00000008'	X'00002103'	CONV(N) mutually exclusive with SPASZ>0 and SPATRUNC.

Table 551. Return and reason codes for the UPDATE TRANDESC command (continued)

Return code	Reason code	Meaning
X'00000008'	X'00002104'	CONV(Y) mutually exclusive with RECOVER(N).
X'00000008'	X'00002105'	CONV(Y) requires SPASZ and SPATRUNC.
X'00000008'	X'00002108'	Invalid EDITRTN name.
X'00000008'	X'0000210C'	FP(E) mutually exclusive with EDITRTN.
X'00000008'	X'0000210E'	FP(E) or FP(P) mutually exclusive with MSC keyword MSNAME or SIDR and SIDL.
X'00000008'	X'0000210F'	FP(E) or FP(P) mutually exclusive with MSGTYPE(MULTSEG).
X'00000008'	X'00002110'	FP(N) mutually exclusive with EMHBSZ > 0.
X'00000008'	X'00002111'	FP(E) or FP(P) mutually exclusive with RECOVER(N).
X'00000008'	X'00002112'	FP(E) or FP(P) mutually exclusive with RESP(N).
X'00000008'	X'00002116'	INQ(N) mutually exclusive with RECOVER(N).
X'00000008'	X'00002119'	MSC keyword MSNAME or SIDL/SIDR mutually exclusive with application program defined as Fast Path exclusive (FP(E)) associated with this transaction.
X'00000008'	X'0000211A'	Invalid MSNAME name.
X'00000008'	X'0000211B'	MSNAME mutually exclusive with SIDL and SIDR.
X'00000008'	X'0000211D'	MAXRGN>0 mutually exclusive with PARLIM(65535).
X'00000008'	X'0000211E'	MAXRGN>0 mutually exclusive with SERIAL(Y).
X'00000008'	X'00002121'	PARLIM value mutually exclusive with SERIAL(Y).
X'00000008'	X'00002125'	REMOTE(Y) requires MSC keyword MSNAME or SIDR and SIDL.
X'00000008'	X'00002126'	Invalid SIDL value.
X'00000008'	X'00002127'	SIDL/SIDR must be specified as a pair. Either SIDL was specified alone or SIDR was specified alone.
X'00000008'	X'00002128'	Invalid SIDL value.
X'00000008'	X'00002133'	Multiple name parameters were specified with DEFAULT(Y). Only one descriptor can be the default at one time.
X'0000000C'	X'00003000'	Command was successful for some of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. If NAME(*) is specified without OPTION(ALLRSP), output lines are only returned for descriptors with non-zero completion codes. See the completion code table for details.
X'0000000C'	X'00003004'	Command was not successful for any of the descriptors. The command output contains a line for each descriptor, accompanied by its completion code. See the completion code table for details.
X'00000010'	X'0000400C'	Command is not valid on the XRF alternate.
X'00000010'	X'00004014'	Command is not valid on the RSR tracker.

Table 551. Return and reason codes for the UPDATE TRANDESC command (continued)

Return code	Reason code	Meaning
X'00000010'	X'00004024'	No Fast Path defined, FP(E), FP(P), or EMHBSZ >0 invalid.
X'00000010'	X'00004120'	Online change phase is in progress.
X'00000010'	X'00004300'	Command is not allowed because online change for MODBLKS is enabled (DFSDFxxx or DFSCGxxx defined with MODBLKS=OLC, or MODBLKS not defined).
X'00000010'	X'00004310'	Storage could not be obtained for the Transaction Input edit routine table. A cold start is required to fix this error.
X'00000010'	X'00004314'	The Transaction Input edit routine could not be loaded.
X'00000010'	X'00004318'	A new Transaction Input edit routine could not be added. The maximum of 255 routines has already been reached.
X'00000014'	X'00005004'	DFSOCMD response buffer could not be obtained.
X'00000014'	X'00005008'	DFSPOOL storage could not be obtained.
X'00000014'	X'0000500C'	AWE could not be obtained.
X'00000014'	X'00005010'	Latch could not be obtained.
X'00000014'	X'000050FF'	The UPDATE TRANDESC command processing terminated because of an internal error.

Errors unique to the processing of this command are returned as completion codes. The following table includes an explanation of the completion codes.

Table 552. Completion codes for the UPDATE TRANDESC command

Completion code	Completion code text	Meaning
0		Command completed successfully for transaction descriptor.
10	NO RESOURCES FOUND	Transaction descriptor name is invalid, or the wildcard parameter specified does not match any descriptor names.
17	ANOTHER CMD IN PROGRESS	Another command (such as DELETE or UPDATE) is in progress for this transaction descriptor. This could also mean this command, if the descriptor is specified by more than one specific or wildcard parameter.
19	CMTMODE=MULT/ WFI=Y CONFLICT	Transaction descriptor update failed because commit mode multiple CMTMODE(MULT) option conflicts with the wait-for-input WFI(Y) option.
1B	CONV=Y/ CMTMODE=MULT CONFLICT	Transaction descriptor update failed because the conversation CONV(Y) option conflicts with the commit mode multiple CMTMODE(MULT) option.
1E	CONV=N/SPASZ/ SPATRUNC CONFLICT	Transaction descriptor update failed because the CONV(N) option conflicts with the SPA size or SPA truncation option.

Table 552. Completion codes for the UPDATE TRANDESC command (continued)

Completion code	Completion code text	Meaning
1F	CONV=Y/ RECOVER=N CONFLICT	Transaction descriptor update failed because the conversation CONV(Y) option conflicts with the nonrecoverable RECOVER(N) option.
27	NOT ALLOWED FOR DEFAULT DESCRIPTOR	Transaction descriptor update failed because the default descriptor cannot update any runtime resource definition attribute.
2F	FP=E/BMPTYPE=Y CONFLICT	Transaction descriptor update failed, because Fast Path exclusive FP(E) option conflicts with program defined as batch BMPTYPE(Y).
35	FP=E/EDITRTN CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) option conflicts with the edit routine EDITRTN.
36	FP=E/FP=N PGM CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) option conflicts with program defined as non-Fast Path FP(N).
38	NOT ALLOWED FOR A BMP	The UPDATE TRAN command is invalid for the resource because the PSB associated with the transaction descriptor is a BMP.
3A	FP=E OR P/MSC KEYWORD CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with the MSC MSNAME, SIDR/SIDL option.
3B	FP/ MSGTYPE=MULTSEG CONFLIC CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with message type multiple segment MSGTYPE(MULTSEG) option.
3C	MAXRGN/ PARLIM=65535 CONFLICT	Transaction descriptor update failed because maximum region count MAXRGN value conflicts with the parallel limit count PARLIM value 65535, which means parallel scheduling is disabled. MAXRGN > 0 is not allowed with PARLIM(65535).
3E	FP=N/FP=E PGM CONFLICT	Transaction descriptor update failed because non-Fast Path FP(N) option conflicts with program defined as Fast Path exclusive FP(E).
3F	FP=P/BMPTYPE=Y CONFLICT	Transaction descriptor update failed because Fast Path potential program FP(P) conflicts with the program defined as batch BMPTYPE(Y).
40	PARLIM/ SCHDTYPE=SERIAL CONFLICT	The PARLIM cannot be changed for the resource because the PSB associated with the transaction is defined as does not have parallel scheduling.
41	FP=E OR P/RECOVER=N CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with nonrecoverable RECOVER(N) option.
42	FP=E OR P/RESP=N CONFLICT	Transaction descriptor update failed because Fast Path exclusive FP(E) or Fast Path potential FP(P) option conflicts with response mode RESP(N) option.

Table 552. Completion codes for the UPDATE TRANDESC command (continued)

Completion code	Completion code text	Meaning
48	NOT ALLOWED FOR IMS RESOURCE	The specified UPDATE command is not allowed for the IMS descriptor or resource. DFSDSTR1 is an example of an IMS descriptor. The only IMS descriptor attribute you can update is DEFAULT(Y).
49	INQ=N/ RECOVER=N CONFLICT	Transaction descriptor update failed because non-inquiry INQ(N) option conflicts with nonrecoverable RECOVER(N) option.
4F	INVALID MAXRGN VALUE	Maximum region MAXRGN value is invalid.
5E	MAXRGN GT 0/SERIAL=Y CONFLICT	Transaction descriptor update failed because non-zero maximum region value conflicts with serial SERIAL(Y) option.
61	DFSBCB STORAGE ERROR.	DFSBCB storage could not be obtained.
6B	PARLIM/SERIAL=Y CONFLICT	Transaction descriptor update failed because the parallel limit PARLIM value conflicts with the serial SERIAL(Y) option.
6D	INVALID PROGRAM NAME	
79	REMOTE/SIDR/ SIDL/MSNAME CONFLICT	Transaction descriptor update failed because remote REMOTE(Y) option requires MSC keyword such as MSNAME or SIDR/SIDL.
8A	WILDCARD PARAMETER NOT SUPPORTED	A wildcard parameter was specified with DEFAULT(Y). Only one descriptor can be the default at one time.
1D7	IMPORT CHANGE LIST ERROR	The descriptor is in the process of being imported from the change list in the IMSRSC repository or was not successfully imported from the change list. The descriptor cannot be updated until it is successfully imported from the repository.

Examples

The following are examples of the UPDATE TRANDESC command:

Example 1 for UPDATE TRANDESC command

TSO SPOC input:

```
UPD TRANDESC NAME(*) SET(FP(E)) OPTION(ALLRSP)
```

TSO SPOC output:

```
Response for: UPD TRANDESC NAME(*) SET(FP(E)) OPTION(ALLRSP)
DescName MbrName    CC CCText
CONVDESC IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
DESC001  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
DESC002  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
DESC003  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
DESC004  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
DESC005  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
DESC101  IMS1        0
```

```
                 DESC102  IMS1        0
                 DESC103  IMS1        0
                 DESC104  IMS1        0
                 DESC105  IMS1        0
|                DESC201  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
|                DESC202  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
|                DESC203  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
|                DESC204  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
|                DESC205  IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
|                DFSDSTR1 IMS1       3B FP/MSGTYPE=MULTSEG CONFLICT
                 FPEDESC  IMS1        0
|                FPPDESC  IMS1       36 FP=E/FP=N PGM CONFLICT
|                MSCDESC  IMS1       3A FP=E OR P/MSC KEYWORD CONFLICT
```

OM API input:

```
CMD(UPDATE TRANDESC NAME(*) SET(OPTION(ALLRSP)))
```

OM API output:

```
<imsout>
<ctl>
<omname>OM1OM   </omname>
<omvsn>1.3.0</omvsn>
<xmlvsn>20  </xmlvsn>
<statime>2006.312 22:26:57.314733</statime>
<stotime>2006.312 22:26:57.315449</stotime>
<staseq>BFAD419D67BADF8C</staseq>
<stoseq>BFAD419D67E79D0C</stoseq>
<rqsttkn1>USRT011 10142657</rqsttkn1>
<rc>0200000C</rc>
<rsn>00003008</rsn>
<rsnmsg>CSLN054I</rsnmsg>
<rsntxt>None of the clients were successful.</rsntxt>
</ctl>
<cmderr>
<mbr name="IMS1    ">
<typ>IMS    </typ>
<styp>DBDC    </styp>
<rc>0000000C</rc>
<rsn>00003000</rsn>
<rsntxt>At least one request successful</rsntxt>
</mbr>
</cmderr>
<cmd>
<master>IMS1    </master>
<userid>USRT011 </userid>
<verb>UPD </verb>
<kwd>TRANDESC        </kwd>
<input>UPD TRANDESC NAME(*) SET(FP(E)) OPTION(ALLRSP) </input>
</cmd>
<cmdrsphdr>
<hdr slbl="DESC" llbl="DescName" scope="LCL" sort="a" key="1"
 scroll="no" len="8" dtype="CHAR" align="left" />
<hdr slbl="MBR" llbl="MbrName" scope="LCL" sort="a" key="2" scroll="no"
 len="8" dtype="CHAR" align="left" />
<hdr slbl="CC" llbl="CC" scope="LCL" sort="n" key="0" scroll="yes"
 len="4" dtype="INT" align="right" skipb="no" />
<hdr slbl="CCTXT" llbl="CCText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="GBL" llbl="Global" scope="GBL" sort="d" key="2" scroll="yes"
 len="1" dtype="CHAR" align="left" skipb="y" />
<hdr slbl="ERRT" llbl="ErrorText" scope="LCL" sort="n" key="0"
 scroll="yes" len="*" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="CONVID" llbl="ConvID" scope="LCL" sort="n" key="0"
 scroll="yes" len="4" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="NODE" llbl="NodeName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
```

```
<hdr slbl="USER" llbl="UserName" scope="LCL" sort="n" key="0"
 scroll="yes" len="8" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="LU" llbl="LUName" scope="LCL" sort="n" key="0"
 scroll="yes" len="24" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TMEM" llbl="TMember" scope="LCL" sort="n" key="0"
scroll="yes" len="16" dtype="CHAR" skipb="yes" align="left" />
<hdr slbl="TPIP" llbl="TPipe" scope="LCL" sort="n" key="0" scroll="yes"
 len="8" dtype="CHAR" skipb="yes" align="left" />
</cmdrsphdr>
<cmdrspdata>
<rsp>DESC(DESC102 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC004 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(DESC205 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(DFSDSTR1) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(DESC201 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(FPPDESC ) MBR(IMS1) CC(  36) CCTXT(FP=E/FP=N PGM CONFLICT)
</rsp>
<rsp>DESC(DESC103 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC005 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(DESC001 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(DESC202 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(DESC104 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(CONVDESC) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(DESC002 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(DESC203 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(MSCDESC ) MBR(IMS1) CC(  3A) CCTXT(FP=E OR P/MSC KEYWORD CONFLICT) </rsp>
<rsp>DESC(DESC105 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC101 ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(FPEDESC ) MBR(IMS1) CC(   0) </rsp>
<rsp>DESC(DESC003 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
<rsp>DESC(DESC204 ) MBR(IMS1) CC(  3B) CCTXT(FP/MSGTYPE=MULTSEG CONFLICT) </rsp>
</cmdrspdata>
</imsout>
```

Explanation: The UPDATE TRANDESC command is issued to update all
transaction descriptors to be Fast Path exclusive. The update succeeded for some
transaction descriptors and failed for others. The update succeeded for some
transaction descriptors as shown by the completion code 0. The update failed for
transaction descriptor FPPDESC with completion code 36, which indicates that the
FP(E) attribute conflicts with the FP(N) attribute defined for the program referred
to by FPPDESC. The update failed for transaction descriptor MSCDESC with
completion code 3A, which indicates that the FP(E) attribute conflicts with the
MSC settings already defined for descriptor MSCDESC. The update failed for some
transaction descriptors with completion code 3B, which indicates that the FP(E)
attribute conflicts with the MSGTYPE(MULTSEG) already defined for the
transaction.

Related concepts:

How to interpret CSL request return and reason codes (System Programming
APIs)

Related reference:

Command keywords and their synonyms (Commands)

Commit Verify exit routine (Exit Routines)

Chapter 31. /VUNLOAD command

The /VUNLOAD AREA command removes the specified areas from the z/OS data space or coupling facility.

All of the updated CIs for the area are written to DASD. All subsequent I/O for the area is from DASD. /VUNLOAD processing occurs concurrently with application processing.

Subsections:
- "Environment"
- "Syntax"
- "Usage notes"
- "Example" on page 1386

Environment

The following table lists the environments (DB/DC, DBCTL, and DCCTL) from which the command and keyword can be issued.

Table 553. Valid environments for the /VUNLOAD command and keyword

Command / Keyword	DB/DC	DBCTL	DCCTL
/VUNLOAD	X	X	
AREA	X	X	

Syntax

```
>>--+--/VUNLOAD--+--AREA--+--<--areaname--+--------------------><
    +--/VUN------+        +--ALL----------+
```
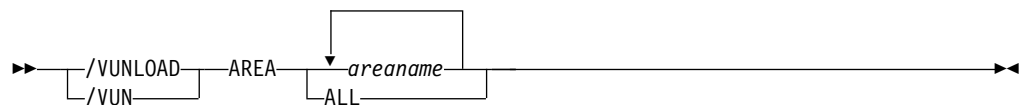
Usage notes

In a data-sharing environment, in order to maintain data integrity, IMS requests IRLM to lock the first control interval (CI) in exclusive mode before IMS unloads the area from the coupling facility. This lock causes those IMS subsystems sharing the area to complete any synchronization point processing. When all sharing subsystems are no longer using the area, IMS requests IRLM to release the CI locks and IMS unloads the area. All subsequent I/O for the area is from DASD.

The /VUNLOAD command does not change any of the VSO options set in the RECON data set. Therefore, at the next IMS restart or /START AREA command, the VSO options again take effect. Any changes to VSO definitions must be made through DBRC commands. For Shared VSO areas however, the /VUNLOAD command is persistent across IMS restarts and can be reset only by a /START AREA command. This is true for shared VSO areas for consistency across all sharing partners. This prevents one system from accessing data from DASD while others are accessing from the coupling facility.

For multiple VSO DEDB areas that share a single coupling facility structure, when the /VUNLOAD AREA command is issued, one of the following actions may occur:

- If there is only one area in the structure (either a single-area structure or a multi-area structure with one area), the area is taken out of VSO and the structure is deleted. The IMS issuing the /VUNLOAD AREA command notifies the other IMS systems sharing the area to disconnect from the structure.
- If there are multiple areas in the structure, when one area is taken out of VSO, the area is disassociated and its CIs are deleted in the structure. The IMS issuing the /VUNLOAD AREA notifies the other IMS systems sharing the area to either disassociate or disconnect from the structure depending on if the IMS is still sharing or not sharing other areas in the structure.

The /VUNLOAD command is not valid for an RSR tracking subsystem.

If the /VUNLOAD command is issued for a local VSO DEDB area during application processing, an FH status code might be returned to the application programs.

This command can be issued to an IMSplex using the Batch SPOC utility.

Example

Entry ET:
```
/VUNLOAD
```

Response ET:
```
NC0000000 FPEC 03070 09:58:03.52 01   00000290  R 14,/VUN AREA DD01AR0
NR8400000 FPEC 03070 09:58:03.52 JOB00116 00000090  IEE600I REPLY TO 14 IS;/VUN AREA DD01AR0
N 8400000 FPEC 03070 09:58:03.53 JOB00116 00000090  DFS058I 09:58:03 VUNLOAD COMMAND IN PROGRESS   SYS3
W 8400000 FPEC 03070 09:58:03.53 JOB00116 00000090  *17 DFS996I *IMS READY*  SYS3
N 8400000 FPEC 03070 09:58:04.42 JOB00116 00000090  DFS2823I AREA DD01AR0
 DISCONNECT FROM STR: DD01AR0STR1 SUCCESSFULS                                      SYS3
N 8400000 FPEC 03070 09:58:04.42 JOB00116 00000090  DFS0488I VUN COMMAND COMPLETED. AREA= DD01AR0  RC= 0 SYS3
```

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those

websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample

programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:
© (your company name) (year).
Portions of this code are derived from IBM Corp. Sample Programs.
© Copyright IBM Corp. _enter the year or years_.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these

publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, See IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www.ibm.com/software/info/product-privacy.

Bibliography

This bibliography lists all of the publications in the IMS 14 library.

Title	Acronym	Order number
IMS Version 14 Application Programming	APG	SC19-4208
IMS Version 14 Application Programming APIs	APR	SC19-4209
IMS Version 14 Commands, Volume 1: IMS Commands A-M	CR1	SC19-4210
IMS Version 14 Commands, Volume 2: IMS Commands N-V	CR2	SC19-4211
IMS Version 14 Commands, Volume 3: IMS Component and z/OS Commands	CR3	SC19-4212
IMS Version 14 Communications and Connections	CCG	SC19-4213
IMS Version 14 Database Administration	DAG	SC19-4214
IMS Version 14 Database Utilities	DUR	SC19-4215
IMS Version 14 Diagnosis	DGR	GC19-4216
IMS Version 14 Exit Routines	ERR	SC19-4217
IMS Version 14 Installation	INS	GC19-4218
IMS Version 14 Licensed Program Specifications	LPS	GC19-4231
IMS Version 14 Messages and Codes, Volume 1: DFS Messages	MC1	GC18-4219
IMS Version 14 Messages and Codes, Volume 2: Non-DFS Messages	MC2	GC18-4220
IMS Version 14 Messages and Codes, Volume 3: IMS Abend Codes	MC3	GC18-4221
IMS Version 14 Messages and Codes, Volume 4: IMS Component Codes	MC4	GC18-4222
IMS Version 14 Operations and Automation	OAG	SC19-4223
IMS Version 14 Release Planning	RPG	GC19-4224
IMS Version 14 System Administration	SAG	SC19-4225
IMS Version 14 System Definition	SDG	GC19-4226
IMS Version 14 System Programming APIs	SPR	SC19-4227
IMS Version 14 System Utilities	SUR	SC19-4228
Program Directory for Information Management System Transaction and Database Servers V14.01.00		GI10-8988
Program Directory for Information Management System Database Value Unit Edition V14.01.00		GI13-4602
Program Directory for Information Management System Transaction Manager Value Unit Edition V14.01.00		GI13-4601

Index

Special characters

V

W

X

IBM ®

Spine information:

IMS   Version 14

Commands, Volume 2: IMS Commands N - V

IBM