



# **Stop Worrying and Learn to Love DB2 Performance Management:**

## **IBM DB2 Query Monitor for z/OS Version 2 Release 1 Best Practices**

**Release: 01.02  
Date: 30 September 2005**

**Authors: Mike Bracey  
Martin Horgan  
Thomas Hubbard  
Gareth Jones  
Mary Petras  
Steve Speller**

# Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

CICS®	Distributed Relational Database Architecture™	DB2 Connect™
DB2 Universal Database™	DB2®	DRDA®
IBM®	ibm.com®	MVS™
OS/390®	PR/SM™	RACF®
S/390®	VTAM®	WebSphere®
z/Architecture™	z/OS®	zSeries®

The following terms are trademarks of other companies:

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 Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

# Table of Contents

Table of Contents.....	4
Table of Figures.....	6
Chapter 1: Introduction .....	8
Chapter 2: Installation.....	12
Overview .....	12
Naming conventions.....	15
SMP/E .....	17
DB2PARMS control file .....	21
Setting Up the Consolidation and Analysis Engine (CAE) Agent.....	23
Installing the CAE Windows components .....	24
GUI Client User Administration.....	28
Catalog Indexes .....	28
Chapter 3: Customizing Query Monitor ISPF Data Views .....	30
Chapter 4: Data Collection.....	37
Setting up the Query Monitor Data Collector .....	38
Workloads .....	39
Summary data compared to exceptions and alerts.....	40
CQMPARMS .....	42
Monitoring profiles .....	48
Chapter 5: GUI Client Usage .....	57
Using the GUI Client to obtain SQL performance metrics to define thresholds for exceptions and alerts.....	57
Setting Exceptions and Alerts.....	60
DB2 Monitoring.....	62
The Alert Browser.....	63
Alerts .....	65
Chapter 6: The DB2 Query Monitor Performance Database .....	69
Overview .....	69
Implementation.....	69
Database Tables .....	70
Table structure .....	70
Sample User Defined Function for DB2 Version 7.....	75
Load Process - Operational Considerations .....	78
Sample Queries.....	80
Chapter 7: Frequently Asked Questions .....	86

Bibliography .....	99
Query Monitor manuals .....	99
DB2 Manuals.....	99
z/OS Manuals.....	99

## Table of Figures

Figure 1: Query Monitor Architecture .....	10
Figure 2: Installation steps summary.....	14
Figure 3: Query Monitor FMIDs .....	17
Figure 4: CAE Agent STDENV specification .....	18
Figure 5: Sample 'D UNI,ALL' output.....	19
Figure 6: Sample Conversion Services JCL.....	20
Figure 7: Sample 'TSO NETSTAT' output .....	23
Figure 8: CAE Windows executables .....	24
Figure 9: The CAE Agent Console .....	25
Figure 10: Selecting 'Profiles & Configurations' from the CAE GUI main window.....	26
Figure 11: The 'Profiles & Configuration' window .....	27
Figure 12: Administrator account details .....	27
Figure 13: CAE user roles and authorities.....	28
Figure 14: Query Monitor column customization primary commands .....	31
Figure 15: The 'Display Exceptions' panel.....	32
Figure 16: Defining fixed columns (1).....	32
Figure 17: Defining fixed columns (2).....	33
Figure 18: The new 'Display Exceptions' panel .....	33
Figure 19: The CQMCLIST CLIST .....	34
Figure 20: Displaying current data set allocations using ISRDDN .....	35
Figure 21: Query Monitor ISPTLIB members .....	35
Figure 22: Data Collection Overview .....	37
Figure 23: Sample CQMPARMS .....	43
Figure 24: The 'Update Profile Line' panel .....	49
Figure 25: The 'Update Monitoring Profile' panel (1) .....	54
Figure 26: The 'Update Monitoring Profile' panel (2) .....	55
Figure 27: The GUI Client Activity Summaries (Operational) panel (1).....	57
Figure 28: The GUI Client Activity Summaries (Operational) panel (2).....	58
Figure 29: The GUI Client Activity Summaries (Operational) panel (3).....	59
Figure 30: The GUI Client Activity Summaries (Operational) panel (4).....	59
Figure 31: The GUI Client Profiles & Configuration panel (1) .....	61
Figure 33: The GUI Client Profiles & Configuration panel (2) .....	63
Figure 34: Configuring Message Boards.....	64
Figure 35: Viewing alerts on the Message Board.....	65
Figure 36: The Message Details window.....	66
Figure 37: Root cause analysis details on the Alert Browser .....	66
Figure 38: Root cause analysis in the Message Details window.....	67
Figure 39: SQL Statement text in the Message Details window.....	68

Figure 40: Query Monitor Performance Database tables .....	70
Figure 41: The summary data tables .....	71
Figure 42: The exception tables .....	72
Figure 43: The SQLCODE tables .....	73
Figure 44: The FTP command stream to download the V7 UDF .....	76
Figure 45: Extracting the UDF load library .....	76
Figure 46: DDL to create the V7 UDF.....	77
Figure 47: Error and Discard DD cards for the performance database load process .....	79
Figure 48: Selecting summary metrics data with SQL text (dynamic SQL).....	80
Figure 49: Selecting summary metrics data and associated SQL text from SYSPACKSTMT (static SQL) .....	80
Figure 50: Selecting summary metrics data and associated DECLARE CURSOR SQL text from SYSPACKSTMT (static SQL) .....	81
Figure 51: Selecting exception metrics and associated SQL text (dynamic SQL) .....	81
Figure 52: Selecting SQLCODE detail data and associated SQL text (dynamic SQL) .....	81
Figure 53: Selecting summary metrics data with associated statement type description .....	81
Figure 54: Summary metrics for all statements in a collection .....	82
Figure 55: Summary metrics including object access for a single statement.....	82
Figure 56: Summary metrics for all statements that access a single table .....	83
Figure 57: View Definition to change all timestamps to decimal in the summary metrics table .....	85
Figure 58: Interval processing events.....	89
Figure 59: GUI Client user administration .....	91
Figure 60: The Exceptions display for negative SQL codes.....	92
Figure 61: Call-level statistics for negative SQL codes .....	93
Figure 62: Displaying SQL Statement Text .....	93
Figure 63: Lock-related statistics .....	94
Figure 64: The SQL Code Summary panel .....	95
Figure 65: The SQL Code Detail Display .....	95
Figure 66: The Display Exceptions panel .....	96
Figure 67: Buffer Pool Statistics .....	97

# Chapter 1: Introduction

DB2 Performance Management is often a source of anxiety for DB2 for z/OS specialists, as dynamic SQL, application changes, changes to the database design, changes in data demographics, or DB2 maintenance can all cause SQL query performance to be less than optimal. As some of this behaviour is unpredictable, what is needed is a tool that can help to identify those SQL queries whose performance levels are outside the bounds of acceptability. DB2 Query Monitor for z/OS is such a tool, as it focuses on performance at the SQL query level rather than at the application level. This white paper is intended to help you stop worrying about how you're going to manage SQL query performance, and put controls in place to identify and analyse troublesome queries. Actually, we can't guarantee you'll learn to love performance management, but we are confident that, if you use Query Monitor, together with the information in this white paper, you'll feel more in control of your DB2 applications.

Any information presented here is intended to supplement that found in the *IBM DB2 Query Monitor for z/OS User's Guide*, SC18-9202 and the *Program Directory for IBM DB2 Query Monitor for z/OS*, GI10-8587. You don't need to have read these manuals before using the information in this white paper, but they are very useful to have at hand.

Apart from some very dodgy philosophising (and there's not much of that), this paper contains information on:

- Query Monitor Installation, including:
  - SMP/E background information
  - Naming conventions
  - Java prerequisites
  - Conversion services considerations
  - Hardware considerations for the CAE Server
  - The DB2PARMS control file
  - Setting up the CAE Agent
  - Installing the CAE Windows components
  - CAE User Administration
  - Recommended user indexes on the DB2 catalog
- Customization of the Query Monitor ISPF data views, and making those views available to all users.
- Data Collection, including:
  - Initial configuration of the Data Collector
  - Considerations for the number of Data Collector tasks
  - Summary data compared to exceptions and alerts
  - Detailed discussion of many of the most significant Query Monitor configuration parameters specified via the CQMPARMS data set

- An explanation of what monitoring profiles are and how to set them up.
- GUI Client usage, including:
  - Obtaining SQL performance metrics to define exception and alert thresholds
  - Setting up exceptions and alerts
  - Administration of the DB2 Monitoring configuration, including monitoring profiles
  - The Alert Browser
- The Query Monitor Performance Database, including:
  - The database structure
  - Defining the database
  - Installing the sample User Defined Function
  - Operational Considerations for the load process
  - Sample queries
- A Frequently Asked Questions (FAQ) section
- Bibliography.

Throughout the document, the terms 'DB2 Query Monitor', 'Query Monitor', 'DB2 QM' and 'QM' refer to IBM DB2 Query Monitor for z/OS Version 2 Release 1.

As described in the *IBM DB2 Query Monitor for z/OS User's Guide*, SC18-9202, Query Monitor provides current and historical views of query activity throughout your DB2 subsystems, enabling you to identify SQL requests that are preventing critical applications from completing within agreed service level agreements. There are two interfaces provided, an ISPF interface and a GUI Client (also called the CAE client). The diagram below shows a sample configuration where Query Monitor is implemented on two LPARs, each with a single DB2 subsystem (possibly in a data sharing group). The Query Monitor subsystem and the Monitoring Agents (one for each monitored DB2 subsystem) run in the same address space, known as the Data Collector.



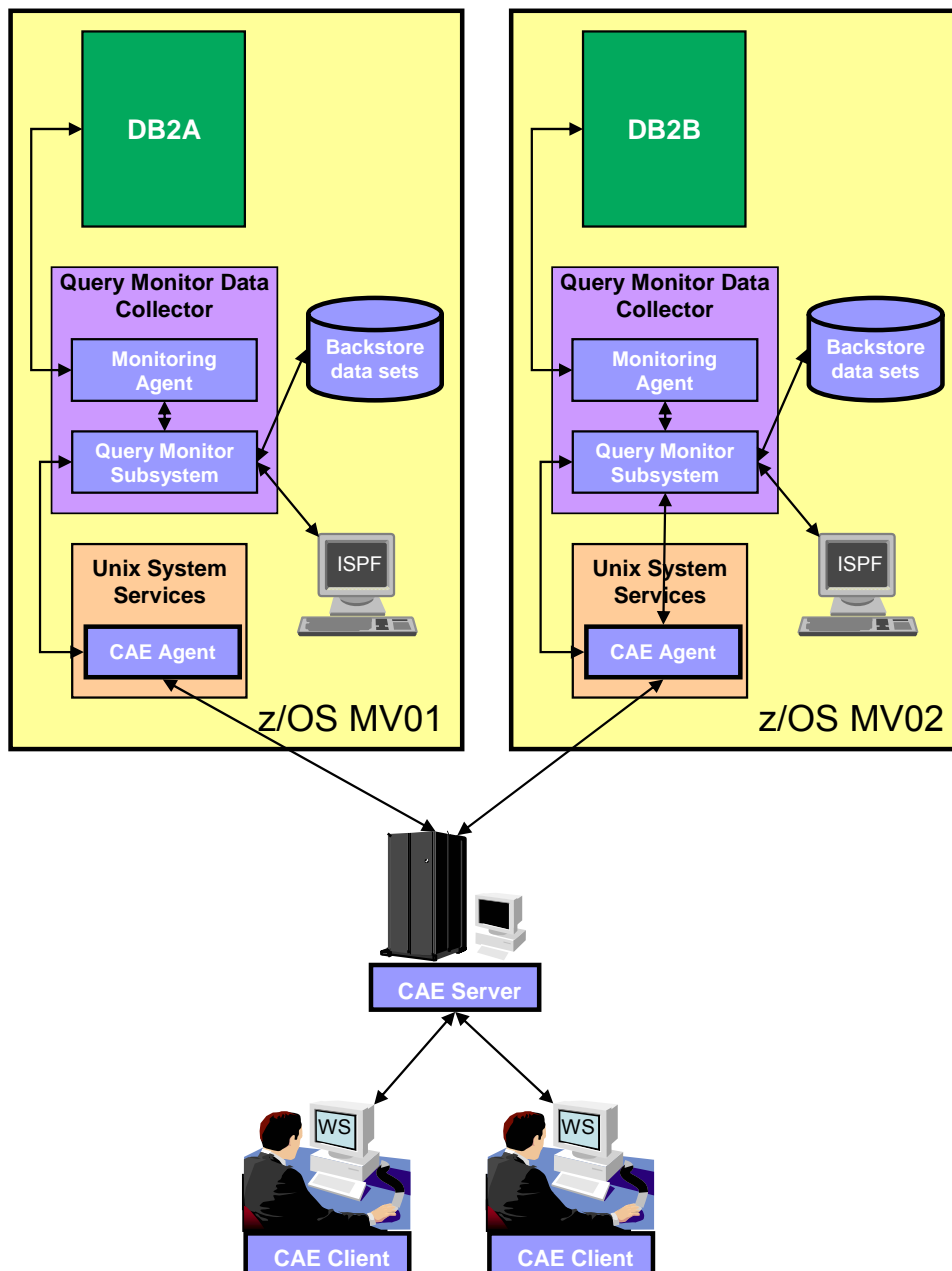


Figure 1: Query Monitor Architecture

The IBM professionals who wrote this book are:

**Mike Bracey**, who works as an IBM EMEA level DB2 Tools on z/OS pre-sales specialist. Mike is based at the Hursley laboratory in the UK and can be contacted at [mike\\_bracey@uk.ibm.com](mailto:mike_bracey@uk.ibm.com).

**Martin Horgan**, who is an IBM zSeries Software Technical Sales Information Management Specialist in the UK. Martin lives very close to the beautiful Derbyshire Dales and can be contacted by email at [HORGANH@uk.ibm.com](mailto:HORGANH@uk.ibm.com).

**Thomas Hubbard**, who works for Rocket Software (an IBM Business Partner) and is DB2 Tools product specialist. Tom is based in Houston, Texas, and can be contacted at [thomas.hubbard@rocketsoftware.com](mailto:thomas.hubbard@rocketsoftware.com).

**Gareth Jones**, who works for the IBM Software Group Product Introduction Centre, is responsible for managing beta programs for DB2 Tools for z/OS and for the DB2 engine. He can be contacted at [jonesgth@uk.ibm.com](mailto:jonesgth@uk.ibm.com).

**Mary Petras**, who is based in Connecticut in the USA, is a DB2 Tools for z/OS Technical Support Specialist and Advocate. Mary can be contacted at [marypetr@us.ibm.com](mailto:marypetr@us.ibm.com).

**Steve Speller**, who, like Mike, works as an IBM EMEA level DB2 Tools on z/OS pre-sales specialist. Steve is also based at the Hursley laboratory in the UK, and he can be contacted at [steve\\_speller@uk.ibm.com](mailto:steve_speller@uk.ibm.com).

## Chapter 2: Installation

This chapter discusses in some detail information pertinent to the installation process. It is not meant to describe the entire process, but rather to provide additional information to assist you during installation and customization.

Use the following manuals for primary installation details, along with the information in this chapter:

- *IBM DB2 Query Monitor for z/OS User's Guide*, SC18-9202 (referred to here as the *Query Monitor User's Guide*)
- *Program Directory for IBM DB2 Query Monitor for z/OS*, G110-8587

### Overview

After a discussion on the order in which installation tasks should be performed, this chapter considers various stages of the installation process, summarized below. These can usefully be divided into three groupings: the mandatory tasks; the tasks required to use the Consolidation and Analysis Engine (CAE) features; and an optional task, which can improve Query Monitor performance when analysing performance data.

- **The Mandatory Tasks**  
This section lists basic prerequisite tasks for Query Monitor and the installation may fail if these prerequisites are not carefully checked.
  - **Naming conventions**  
Deciding on a naming convention is a planning activity and affects many other tasks.
  - **SMP/E**  
This part of the installation is often performed by an SMP/E specialist or z/OS Systems Programmer, but can also be performed by the DB2 specialist responsible for implementing Query Monitor.
  - **QM prerequisites**  
In this section we discuss the z/OS actions you need to perform to be able to install and configure Query Monitor correctly.
  - **DB2PARMS control file**  
The DB2PARMS control file is used by many of the DB2 Tools to store information about the DB2 subsystems where the Tools are installed. Here we discuss the options available to set up DB2PARMS.
- **CAE Required Tasks**  
If you want to use any of the features of the CAE, or even if you simply want to use the GUI Client, you need to perform the following tasks.
  - **Setting up the CAE Agent**  
The CAE Agent runs under z/OS Unix System Services and acts as a communication channel between the Query Monitor Data

Collector and the CAE Server, which runs on Windows and provides access to performance data for the GUI Client.

- **Installing the CAE Windows components**  
This section discusses the Windows installation of the CAE Server and the GUI Client.
- **GUI Client User Administration**  
GUI Client users connect to the CAE Server using an id and password maintained separately from z/OS or Windows user ids, so some advice is given on GUI Client User Administration.
- **Optional Performance Task**
  - **Recommended Catalog Indexes**  
Query Monitor uses SQL queries to obtain information from the DB2 catalog about SQL statement text. To make access to this data as efficient as possible, you can define user indexes on DB2 catalog tables.

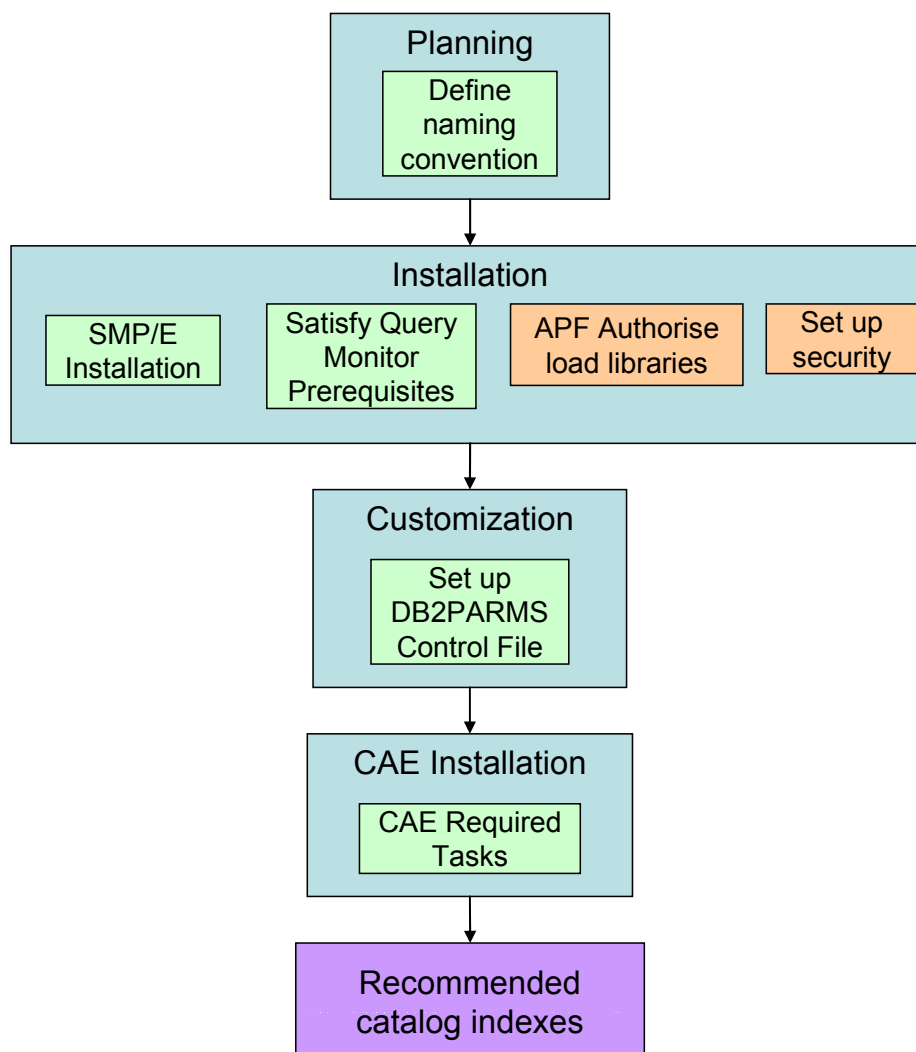
## The Installation Sequence

Before discussing these tasks in detail, a few general comments on product installation are in order. The key to a successful and (hopefully) speedy installation is careful planning. It is important that you understand how the installation tasks are related to each other, and what dependencies there are. This enables you to draw up a plan where delay in the process is minimized and the installation and customization process becomes more efficient. Remember, we only highlight some of the tasks which you have to perform to install and configure Query Monitor, and do not outline the complete process. The information here supplements that in the Program Directory and the User Guide, which should be your primary source of information. Now, onto the discussion of the sequence of the installation tasks.

- As establishing a naming convention is part of the planning process, you should complete this, along with the rest of your installation and customization planning, before any of the other tasks are started.
- The next two tasks which need to be completed are the SMP/E installation and making sure your systems meets all the Query Monitor pre-requisites. The order in which you accomplish these two tasks really doesn't matter, and you may decide to perform them in parallel. However, you cannot proceed to any other tasks until these two have been completed. Other actions that you can usefully do at this stage are APF-authorisation of the load libraries, and setting up the authorisations needed to customize and use Query Monitor. These are discussed in Chapter Three, Customization, of the *Query Monitor User's Guide*.
- The next task is to create the DB2PARMS control file (unless one already exists), followed by the remaining customization tasks listed in Chapter Three, Customization, of the *Query Monitor User's Guide*, except those required for the CAE. Unless indicated in the User Guide chapter as optional, all these tasks are mandatory.

- If using the CAE, you should then proceed to follow the steps listed in Chapter Three, Customization, of the *Query Monitor User's Guide*, in conjunction with the CAE Required Tasks discussed in this document.
- The final task is to define user indexes on the DB2 catalog. This task is unusual in that, because it is independent of the other tasks, can be done at any time in the process. However, most benefit can be obtained by performing the task after installation, customization and configuration.

The installation process is summarized in the diagram below:



Legend:

- Main reference: *IBM DB2 Query Monitor for z/OS User's Guide* , SC18-9202
- Additional references: this chapter
- Sole reference: *IBM DB2 Query Monitor for z/OS User's Guide* , SC18-9202
- Sole references: this chapter

Figure 2: Installation steps summary

## Naming conventions

Establish good naming conventions for the environments where QM is to be deployed:

- Establish a convention for the run-time libraries, bearing in mind that it might be important to be able to identify the tool based on the library name.
  - If you have one set of run-time libraries per QM Data Collector, you may wish to include the LPAR name (where you have no more than one data collector per LPAR) and/or Query Monitor data collector name (where you have, or have the potential for, more than one data collector per subsystem) as data set name qualifiers.
  - Using the LPAR name as a data set name qualifier could, however, become problematic if you ever need to move the data collector to another LPAR.
- Establish a convention for the SMP/E target and distribution libraries.
- Establish a naming convention for the subsystem name of the QM data collector. You should take into consideration:
  - How many QM data collectors you are likely to run on any given LPAR
  - The likelihood of moving workloads and therefore the QM data collector from one LPAR to another
  - Whether multiple DB2 subsystems could be monitored by a single QM data collector, or each DB2 subsystem will be monitored by a dedicated QM data collector.
- From the QM data collector name, derive the names or high level qualifiers of the QM VSAM back-store data sets.

Choose these names carefully and consider specific site standards as well as usability.

Determining the naming convention at the planning stage will lay the foundations for a solid, unambiguous naming convention which will work well in your environment. You may decide to use different names for production and test. You may also choose to have separate libraries for each DB2 tool or one set of libraries for all of the tools. The unique three character designation for DB2 Query Monitor is *CQM*. Consider using these unique three characters in the dataset name, and optionally the version and release number.

A sample dataset naming convention is illustrated below as a guideline for developing your own. In all of these, there are two high level qualifiers (DMSYS and DB2TOOLS) which identify the library owners and the library subset, and the low-level qualifier is the library or data set name as defined by IBM.

In the first example, the three character product identifier (CQM) and the LPAR name (MV50) are used as data set name qualifiers. Here, there is a

single data collector for the LPAR, and there is little or no likelihood of moving it to another LPAR:

```
DMSYS . DB2TOOLS . CQM . MV50 . SCQMDBRM
DMSYS . DB2TOOLS . CQM . MV50 . HFS
DMSYS . DB2TOOLS . CQM . MV50 . SCQMLoad
DMSYS . DB2TOOLS . CQM . MV50 . SCQMMENU
DMSYS . DB2TOOLS . CQM . MV50 . SCQMPENU
DMSYS . DB2TOOLS . CQM . MV50 . SCQMSAMP
DMSYS . DB2TOOLS . CQM . MV50 . SCQMTRAN
```

In the second example, as well as the three character product identifier and LPAR name, the name of a DB2 subsystem (DBDM) is used as an additional qualifier. You might use this method where you to associate a data collector with a specific DB2 subsystem on a specific LPAR, but be aware that this may cause administrative complications if you ever need to move the DB2 subsystem to another LPAR or need to monitor multiple DB2 subsystems with a single Query Monitor data collector:

```
DMSYS . DB2TOOLS . CQM . MV50 . DBDM . SCQMDBRM
DMSYS . DB2TOOLS . CQM . MV50 . DBDM . HFS
DMSYS . DB2TOOLS . CQM . MV50 . DBDM . SCQMLoad
DMSYS . DB2TOOLS . CQM . MV50 . DBDM . SCQMMENU
DMSYS . DB2TOOLS . CQM . MV50 . DBDM . SCQMPENU
DMSYS . DB2TOOLS . CQM . MV50 . DBDM . SCQMSAMP
DMSYS . DB2TOOLS . CQM . MV50 . DBDM . SCQMTRAN
```

Make sure, of course, that none of the data set names exceed the allowed limit of 44 characters, including periods.

You should also consider naming standards for your SMP/E target and distribution libraries. If you use SMP/E target libraries for your run-time libraries, then you can derive the names for the distribution libraries from these. If you keep your SMP/E libraries separate from your run-time libraries, then a simplified version of the recommendations made above, without reference to LPAR names or Query Monitor subsystem names, is appropriate. The following example illustrates a possible naming convention for the target libraries. Here, the three character product identifier (CQM) and the version number (V210) make up the third-level qualifier for the data set names:

```
DMSYS . DB2TOOLS . CQMV210 . SCQMDBRM
DMSYS . DB2TOOLS . CQMV210 . HFS
DMSYS . DB2TOOLS . CQMV210 . SCQMLoad
DMSYS . DB2TOOLS . CQMV210 . SCQMMENU
DMSYS . DB2TOOLS . CQMV210 . SCQMPENU
DMSYS . DB2TOOLS . CQMV210 . SCQMSAMP
DMSYS . DB2TOOLS . COMV210 . SCOMTRAN
```

## SMP/E

SMP/E is the basic tool for installing and maintaining software in z/OS systems. If you do not already understand SMP/E, we recommend you ask an SMP/E expert to perform this part of the installation for you. Whoever carries out this task should consult the Program Directory shipped with the product, *Program Directory for IBM DB2 Query Monitor for z/OS*, G110-8587.

### Maintenance Recommendations

The FMIDs can be found in the Query Monitor program directory; at the time of writing there are two FMIDs supplied with the product, one for DB2 Query Monitor and the other for FEC common code (this code is used by multiple DB2 Tools products).

DB2 Query Monitor consists of the following FMIDs:

Product	FMID
DB2 Query Monitor V2R1	H238210
FEC Common Code	H25F132

Figure 3: Query Monitor FMIDs

NOTE: FMID H25F132 contains common code and is shared among multiple DB2 Data Management Tools, e.g. DB2 Table Editor, DB2 Automation Tool, etc. FMID H25F132 is delivered with all products that use the common code. You should only install the FMID once, and make it available to all products. For this reason, you may find it advisable to install all DB2 Tools products into one SMP/E CSI.

Important: When installing QM V2R1 ensure that all PTFs for the Query Monitor FMIDs are applied via SMPE. This applies to the FMIDs for the QM product and the FEC Common Code.

### QM prerequisites

QM requires minimum software and hardware configurations. It is recommended you examine the prerequisites and review them before starting the installation.

### Software and hardware prerequisites

These prerequisites for QM are listed in Chapter 2, Planning and Deployment, of the *Query Monitor User's Guide*.

Rather than repeat all of the documented requirements here, we only make note of some additional considerations for the following items (please note, this is *not* a comprehensive list):



- Java
- Conversion Services
- Language Environment
- QM server host
- Network considerations

## Java

The current requirement for QM is that Java build cm1411sr2a-20040515 or higher be installed on z/OS or OS/390. This is the Java for z/OS 1.4.1 SDK with service refresh 2a.

**Note:** Java build cm1411sr2a-20040515 is available at:  
<http://www.ibm.com/servers/eserver/zseries/software/java/aboutj14.html>

You may have an earlier version of Java installed and in use by existing applications. You also need to install the Java build required by Query Monitor; it can be installed into another directory without affecting existing applications.

Java is required by the Consolidation and Analysis Engine (CAE) agent, and you can specify the location of Java 1.4.1 through the STDENV file in the CAE agent JCL:

```
//STDENV DD *
JAVA_HOME=/usr/lpp/java141
CQM_LOGS=/var/cqm/logs
```

**Figure 4: CAE Agent STDENV specification**

This information was recently added to the QM Program Directory.

## Conversion Services

Conversion Services is an installed component of z/OS V1R2 and higher and is optionally installed as an extra step for OS/390 V2R10. It performs conversion between one coded character set identifier (CCSID) and another. These could be ASCII, EBCDIC or Unicode CCSIDs.

DB2 Query Monitor V2 requires that Conversion Services be installed and activated on any LPAR where the QM Data Collector is to be run. At startup of the QM ISPF monitor, QM V2 verifies translations are available from CCSIDs 500 and 1208 and the user's terminal CCSID. The reason for this is that QM V2 stores captured SQL in Unicode. If the conversion for a user's terminal is not defined to Conversion Services, the following message is issued to the user:

```
CQM153E - RETURN CODE 08 REASON CODE 0003 WAS ENCOUNTERED DURING
TRANSLATION. SOURCE CCSID = 500 TARGET CCSID = 285
```

The message above indicates that the user's terminal is running codepage 285 and there is no translation page for CCSID 500 to CCSID 285.

You will also need to ensure that translations to and from all the above CCSIDs to those used by the DB2 subsystem are available (these are specified in the DSNHDECP initialization parameter load module).

Use the MVS DISPLAY command "D UNI,ALL " to check that Conversion Services has been activated, and to see the conversions installed on this LPAR.

Note: If Conversion Services is not active on an LPAR, an IPL is required to enable it. Implementation of Conversion Services will normally be done by your MVS Systems Programmer.

Here is some sample output from the DISPLAY command:

```
-D UNI,ALL
CUN3000I 13.21.49 UNI DISPLAY 601
ENVIRONMENT:  CREATED      12/30/2004 AT 14.31.14
              MODIFIED     02/09/2005 AT 11.35.28
              IMAGE CREATED 02/09/2005 AT 11.31.09
SERVICE:    CHARACTER     NORMALIZATION CASE
STORAGE:    ACTIVE        620 PAGES
              LIMIT        2000 PAGES
CASECONV:   NORMAL
NORMSERV:   DISABLED
CONVERSION: 00850-01047-ER      01047-00850-ER
              00037-01200(13488)-ER 01200(13488)-00037-ER
              00500-01200(13488)-ER 01200(13488)-00500-ER
              01047-01200(13488)-ER 01200(13488)-01047-ER
              01208-01200-ER        01200-01208-ER
              01383-01200-ER        01200-01383-ER
              00932-01200-ER        01200-00932-ER
              00939-01200-ER        01200-00939-ER
```

Figure 5: Sample 'D UNI,ALL' output

Below is some sample JCL and control cards for adding the conversion tables to allow conversion between codepages 037, 1208 and 1388. It is recommended that you add the parameter ER, to indicate the conversion technique search order, when defining the conversions. See the *DB2 for z/OS Installation Guide*, or *z/OS Support for Unicode Using Conversion Services*, SA22-7649, for more information on what these parameters mean.

```

//CUNMIUTL EXEC PGM=CUNMIUTL
//SYSPRINT DD SYSOUT=*
//TABIN DD DISP=SHR.DSN=SYS1.SCUNTBLL
//SYSIMG DD DSN=SYS1.IMAGES(CUNIMG00),DISP=SHR
//SYSIN DD *
CASE NORMAL;
CONVERSION 37,1388,ER;
CONVERSION 1388,37,ER;
CONVERSION 1208,1388,ER;
CONVERSION 1388,1208,ER;
CONVERSION 37,1208,ER;
CONVERSION 1208,37,ER;

```

**Figure 6: Sample Conversion Services JCL**

Note that conversion tables are always specified as code page pairings; to allow conversion between code page 1208 and 1388, 1208 and 037, and 037 and 1388, all those conversions must be configured. Note also that, for each pairing, you need to specify conversion both ways. For example, from 1208 to 1388 and from 1388 to 1208.

z/OS does not require an IPL to add conversion tables to Conversion Services, but your site rules might dictate that they can only be added via an IPL.

**Note:** For OS/390 V2R10 installations, Unicode support code must be downloaded from the IBM Software download page , [www.software.ibm.com/download](http://www.software.ibm.com/download).

The installation of z/OS support for Unicode with SMP/E is described in *z/OS Planning for Installation, GA22-7504*.

## Language Environment

Ensure that the CEE.SCEERUN2 library is APF authorized and in the LNKLIST. Otherwise, you may receive an U4093 abend with return code 3EB for module CEEBINIT when the CAE agent running as a daemon is started in USS.

## QM CAE Server

This task is required only if you are planning to use the Consolidation and Analysis Engine (CAE).

A dedicated server machine in the customer's network is necessary for the QM CAE Server component. This server needs to run Windows 2000, Windows XP or Windows 2003.

If you cannot get a dedicated machine for the CAE Server, you can still install QM on the host and revisit the CAE Server and GUI client installs at a later date. Alternatively, if a dedicated machine is not available, you can use the same personal workstation as the client component.

However, if the server is down there are three important points to keep in mind:

- All alerts go to the server. If the server is down, these alerts will only be kept in a queue maintained by the CAE Agent until the ALERT\_LIMIT value is exceeded, and then any subsequent alerts will be lost. See the chapter on Data Collection for more details on this COMPARMS setting.
- If the server is on a workstation and that workstation is down, no one will be able to logon using their GUI client.
- Much of the processing power of the personal workstation where the server is installed will be used servicing requests from GUI clients and processing alert data. This is probably undesirable from the point of view of the user of that workstation.

As a result, installing the server on a personal workstation is not recommended in a production environment. Ideally the server should be installed on a dedicated machine.

### Network protocols

The CAE uses TCP/IP as the communications vehicle. TCP/IP connectivity must be available between the CAE Server and all other components. The CAE server runs on a Windows host. The CAE agent runs as a USS daemon under z/OS.

NOTE: DB2 Connect is NOT required for QM.
---

### DB2PARMS control file

The DB2PARMS control file is a VSAM KSDS dataset used by some of the DB2 tools products that contains information about each DB2 subsystem ID, and specific details about each corresponding DB2 subsystem.

### Sharing of DB2PARMS

If the DB2PARMS control file has been already set up for other DB2 Tools products, you can also use that file for the QM product. Sharing of the DB2PARMS control file possible among different DB2 Tools products is possible because of the way the products use MVS facilities to access the file (using ENQ and DEQ macros). Several DB2 tools can share the same DB2PARMS control file because each tool uses a different key value to access the VSAM KSDS. The key value contains both a product identifier and a DB2 subsystem identifier. Therefore, if another product already has the DB2PARMS control file defined, you do not need to define a different DB2PARMS control file for the product you are currently configuring. However, all of the DB2 Tools products require the same set information for each DB2 subsystem. Therefore, if you have already defined information stored in the DB2PARMS control file for, say, subsystem DSN for DB2 Log

Analysis tool, you still need to define the parameters for QM for this same DB2 subsystem, DSN.

If the DB2PARMS control file does not exist or you do not wish to share it across multiple DB2 products, then use member CQMCNTFL in dataset SCQMSAMP to create a new DB2PARMS control file. Edit this member according to the embedded instructions in the JCL and run it to allocate the VSAM dataset to be used as the DB2PARMS control file for QM.

## Setup DB2PARMS

Once the DB2PARMS control file is created, you use option S (the 'Setup' function) from the main QM panel to provide information for each DB2 subsystem you plan to monitor with a single QM Data Collector. This step actually occurs quite late in the customisation process as you need to have started the Query Monitor data collector to be able to use the 'Setup' option (see Step 15 in the customization process as outlined in the *Query Monitor User's Guide*). The information QM needs is:

- DB2 DSNZPARM member name
- BSDS dataset names
- DB2 load library names
- QM plan name

## Considerations

You do not need multiple control datasets for multiple DB2 subsystems. Indeed, only one DB2PARMS control file is necessary for monitoring multiple DB2 subsystems on a single LPAR. If you are monitoring members of a data sharing group on a single LPAR with one QM started task, then we recommend that you identify all of these members within the same DB2PARMS control file.

You can share the same DB2PARMS control file among multiple subsystems. For example, one DB2PARMS control file can be shared by all products, or you can have one DB2PARMS control file for each product. Whichever you decide, update the profile information for each DB2 subsystem within the different products using the setup option.

Our recommendation is to have a single, shared DB2PARMS control file. The only reasons for having multiple DB2PARMS control files is either if you have multiple subsystems on different LPARS without shared DASD, or if you want to support multiple versions of the products in the same DB2 subsystem.

## Authorization for the DB2PARMS control file

It is recommended that the installer define the proper RACF controls on the DB2PARMS control file to allow read and update access to appropriate ids. In this way, the DB2PARMS control file has proper security to prevent access by unauthorized users.

## Setting Up the Consolidation and Analysis Engine (CAE) Agent

The CAE agent uses various TCP and UDP ports for communications. According to the list in the User's Guide, use of port 3444 is required for the product's messaging. This port, in particular, is set in the product, so to ensure successful installation you should reserve this port for the CAE agent. You can check to see if this port is already reserved for another application by use of the NETSTAT command.

In TSO (option 6) issue the command NETSTAT PORTList. Look for port 3444 to see if it has been reserved. The following screenshot shows the output of the NETSTAT PORTList command and port 3444 that has been reserved for OMVS.

EZZ2350I	MVS	TCP/IP	NETSTAT	CS	V1R4	TCPIP Name:	TCPIP
EZZ2795I	Port#	Prot	User	Flags	Range	IP Address	
EZZ2796I	-----	----	----	-----	-----	-----	-----
EZZ2797I	03444	TCP	OMVS	DA			
EZZ2797I	04600	TCP	DBM6DIST	DA			
EZZ2797I	04601	TCP	DBM6DIST	DA			
EZZ2797I	00580	UDP	NCPROUT	DA			
EZZ2797I	00750	UDP	MVSKERB	DA			
EZZ2797I	00751	UDP	ADM@SRV	DA			
EZZ2797I	03444	UDP	OMVS	DA			

Figure 7: Sample 'TSO NETSTAT' output

This only shows that port 3444 has been reserved. You should issue a TSO NETSTAT CONN command to see if anything is actually using the port because even if it has not been reserved, an application may be using it. Another option is to browse the TCPPARMS dataset and look for port 3444.

If port 3444 has already been reserved for another application, then you have 3 choices:

1. Change the application that has reserved port 3444 to use another port.
2. Wait for the availability of a planned PTF which will allow you to specify the port number. At the time of writing the PTF number is unknown.
3. Contact IBM and ask *if* they can provide a means of changing the CAE agent so that it uses another port.

If you are intending to reserve port 3444 for the CAE agent, it is recommended that you use a name that lets you identify the agent in a NETSTAT display. For example:

3444	TCP	QMCAE	;	DB2	Query	Monitor	Agent
3444	UDP	QMCAE	;	DB2	Query	Monitor	Agent

There can only be 1 CAE agent per LPAR. The CAE agent will automatically

discover the QM Data Collector(s) and become aware of the subsystems they are monitoring. It is possible to have multiple QM Data Collectors in any given LPAR, each with one or more different target DB2 subsystems. However, all defined alerts from those DB2 subsystems will be processed through a single CAE agent.

## Installing the CAE Windows components

The CAE Windows components are available in the SCQMTRAN target library.

The following table lists the different members in this library and suggested executable names. You may use any name for the executable names; the table below is only a guideline.

Member name	Executable name	Description
COMCAEAL	CAESetup.exe	Installs software for both the CAE Server and the GUI Client
COMCAEGU	GUI-client-setup.exe	Installs software for the GUI Client ONLY
COMCAEPT	CAESetup2.exe	Installs the PTFs for the CAE Server and the GUI Client
COMCAEWB	CAESetupW2003.exe	Installs the PTF for the CAE Server and the GUI Client in a Windows 2003 environment.

**Figure 8: CAE Windows executables**

Notice that there are two install executables for both the QM CAE server and the GUI Client. Each should be transferred in binary mode to the CAE Server Windows machine and the GUI Client Windows machine as executable files, preferably using FTP rather than 3270 emulator facilities.

To install the CAE Server and the GUI client, you must run CAESetup.exe first, followed by CAESetup2.exe, otherwise the level of the CAE code will not reflect the level of the CAE Agent running on z/OS, and the GUI Client may not function correctly. For the GUI Client alone, you may alternatively use GUI-client-setup.exe on the client machine. Remember to follow the install with the CAESetup2.exe as it contains the fixes for the GUI Client. There is no option to install the server component on its own, i.e. without the GUI Client.

Note: CAESetupW2003.exe was delivered via APAR PQ91162 and is intended for a Windows 2003 environment only. To run this executable, you need first to run CAESetup.exe or GUI-client-setup.exe.

For the server machine, you need to run CAESetup.exe followed by CAESetupW2003.exe and CAESetup2.exe in any order.

For the client machine, you need to run GUI-client-setup.exe followed by CAESetupW2003.exe and CAESetup2.exe in any order.

After the installation of the CAE Server and GUI Client, you should verify the installation. You will find the QM applications in a folder called "DB2 Query Monitor" on your Windows start menu.

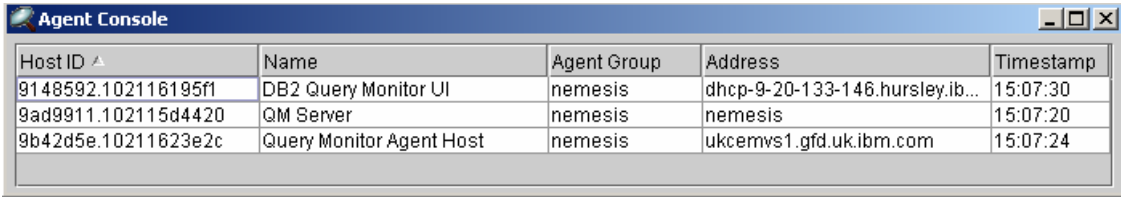
This is best done by first starting the **Agent Console**. The Agent Console is used to show CAE connections to the CAE server. It is recommended that you start the Agent Console first before starting the CAE server so that you can verify that the server starts successfully. The agent console also allows you to see the successful connection of the CAE agent(s) from the host(s) and also the CAE GUI clients. This may be important if you need to shutdown the CAE server for some reason. It is recommended that the CAE agent console remain open permanently.

Starting the Agent Console on the workstation where a CAE GUI is running, (i.e. not the CAE server), has little value since all you will see is the information about the CAE server that you connected to and your CAE GUI session. You will not see information about the CAE Agent(s) or other CAE GUI users.

Next step is to start the **DB2 Query Monitor Server**. This will open and immediately close a window on the CAE Server machine, and to check that the CAE Server is started you should go to the Agent Console.

The CAE Agent Console window below depicts a successful trio of QM components :

- **DB2 Query Monitor UI**  
This represents a single GUI Client; you may see more than one.
- **QM Server**  
This represents the CAE Server; at most one CAE Server will appear.
- **Query Monitor Agent Host**  
This represents the CAE Agent; multiple CAE Agents are possible.



Host ID ^	Name	Agent Group	Address	Timestamp
9148592.102116195f1	DB2 Query Monitor UI	nemesis	dhcp-9-20-133-146.hursley.ib...	15:07:30
9ad9911.102115d4420	QM Server	nemesis	nemesis	15:07:20
9b42d5e.10211623e2c	Query Monitor Agent Host	nemesis	ukcemvs1.gfd.uk.ibm.com	15:07:24

Figure 9: The CAE Agent Console

Now that the QM CAE Server is successfully started you should start the **DB2 Query Monitor Client**. As a general rule there is no need to start the CAE GUI on the server other than for initial testing. If you intend to access more than one CAE server (not at the same time), then it is recommended that you set up an alias for the IP address of the CAE server(s) in the HOSTS file. The HOSTS file typically is found in the path c:\windows\system32\drivers\etc\. The reason to do this is that the CAE GUI does not remember the name of all the CAE servers that you have connected to other sessions. It only



remembers the last one that you used. Consequently, you have to specify an agent portal address when you want to change to another CAE server. If you set up an alias for all of the CAE servers that you will access, then you just need to specify the alias name in the agent portal address box. The alias name can be any name that is meaningful to you.

The built-in administration user id/password that comes with the CAE server is administrator/password. You can not remove (delete) the administrator userid, but you can and should change the password at your earliest opportunity (see the section on the 'Users configuration editor' in Chapter 15, Administrator tools and procedures, in the QM User's Guide). The process for changing the password is shown below:

Select **Tools** on the taskbar at the top of the client window, and then select **Profiles & Configurations**

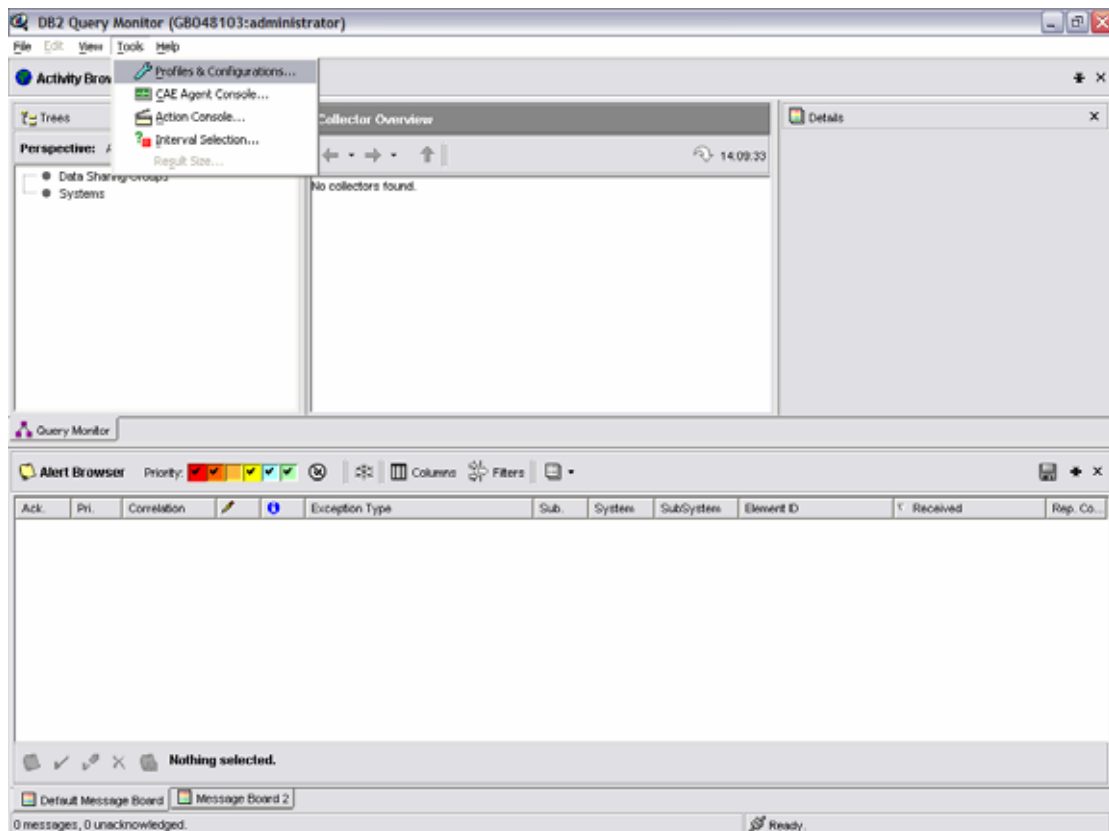


Figure 10: Selecting 'Profiles & Configurations' from the CAE GUI main window.

This opens a new window, from which you should select the **Users** tab at the bottom left hand side of the window to present the screen below:

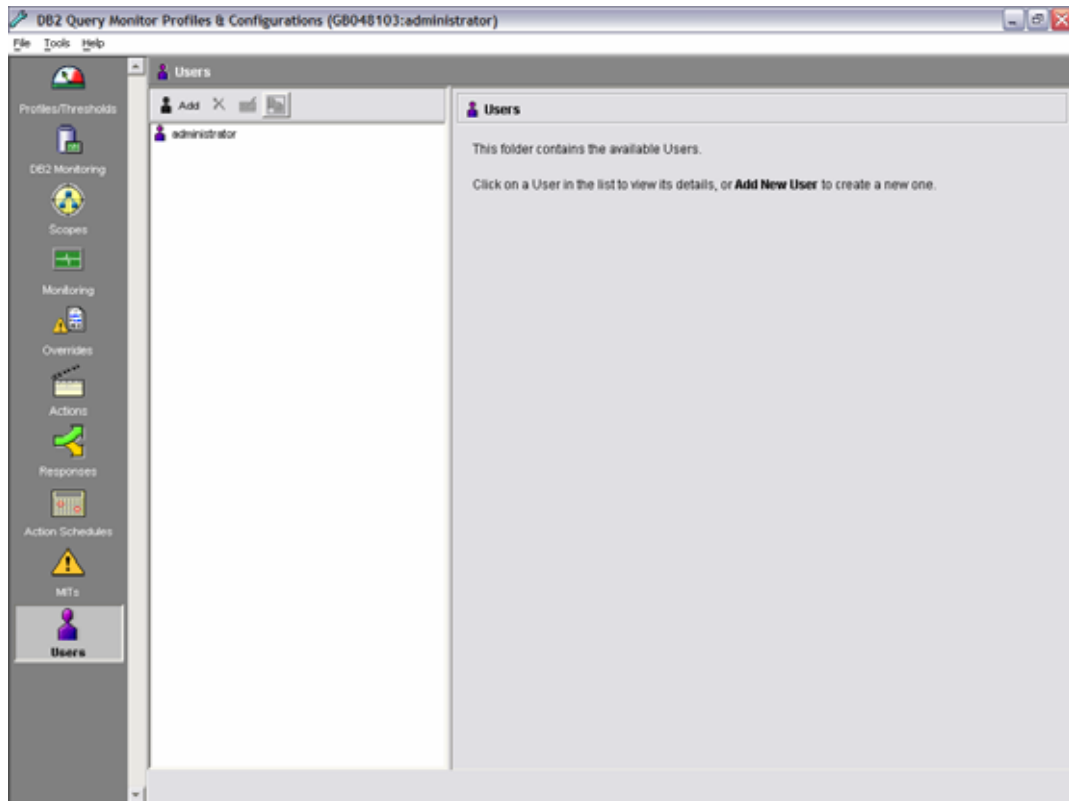


Figure 11: The 'Profiles & Configuration' window

Now select the **administrator** account by moving the cursor over it and right clicking. The account details appear on the right hand side of the window.

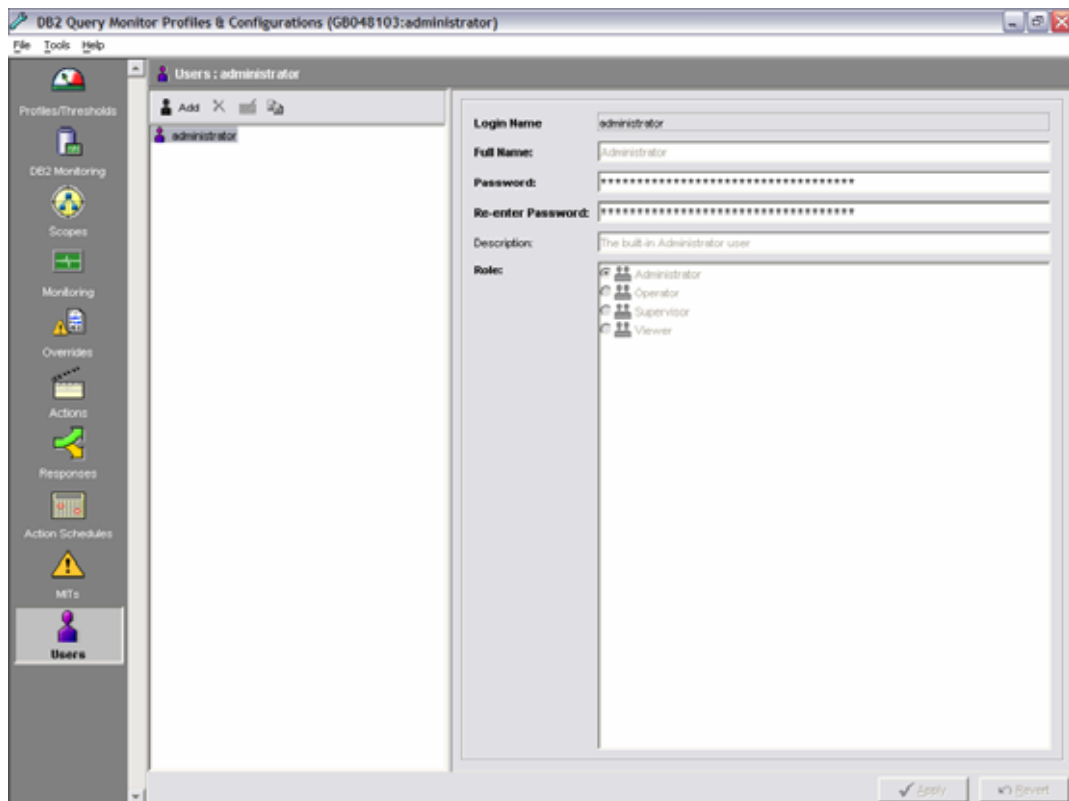


Figure 12: Administrator account details

You see from this window that only the password can be changed. Enter a new password in both the password and re-enter password fields, and press the **Apply** button at the bottom of the window.

## GUI Client User Administration

It is recommended that every user who intends to access the CAE server via the CAE GUI should be assigned a user id and role. This also applies to any users who are administrators. There are 4 possible roles (in order of ascending relative authority):

Role	Authority
Viewer	Can not acknowledge alerts or access "Tools" panel
Operator	As for Viewer but can also acknowledge alerts
Supervisor	As for Operator but also has access to the "Tools" panel, except for "users" administration, but not cancel threads
Administrator	As For Supervisor but can also administer userid and passwords and cancel threads

Figure 13: CAE user roles and authorities

User ids and passwords are assigned by a user id with a role of Administrator. There is currently no synergy with the Windows operating system for administering user ids and passwords.

Generally the CAE Administrator will have to assign and maintain passwords for users. This may compromise the security policy at your installation.

Since the CAE agent passes alerts from all monitored DB2 subsystems in any given LPAR, there is no possibility of limiting a user's view of alerts to a subset of those DB2 subsystems.

## Catalog Indexes

To ensure the optimal performance accessing static SQL text from the DB2 catalog, it is recommended that you define the following indexes:

On SYSIBM.SYSPACKSTMT:

```

LOCATION
COLLID
NAME
CONTOKEN
SECTNO
STMTNO
SEQNO

```

This is very similar to DSNKSX01 but with extra columns SECTNO and STMTNO added before SEQNO. This is because some of the ordering requested by Query Monitor includes SECTNO and STMTNO.

On SYSIBM.SYSDBRM:

PLNAME  
NAME  
TIMESTAMP

There are no default indexes on SYSDBRM, but all of the QM SYSDBRM queries use these three columns as predicates. You may wish to exclude TIMESTAMP from the index, as there is always data page access when obtaining the statement text.

On SYSIBM.SYSSTMT:

PLNAME  
NAME  
SECTNO  
STMTNO  
SEQNO

Again, there are no default indexes on SYSSTMT, but these are the columns used by Query Monitor as predicates and for ordering.

These indexes were identified by using Query Monitor to identify the SQL text used in its own calls, and then determining appropriate indexes to use. You may wish to perform this process yourself, to validate the indexes identified are appropriate for your environment. As maintenance may change the SQL statements used by Query Monitor, you should also validate these indexes again when maintenance is applied which hits members of SCQMDBRM.

# Chapter 3: Customizing Query Monitor ISPF Data Views

## Overview

When QM is initially installed, there is a specific pattern to the various data displays associated with each panel where performance data is displayed. That is, the data on these panels is presented in rows and columns. The columns represent items of performance data, such as plan name, elapsed time, or DB2 object name. These columns are displayed in a pre-defined order (left to right), in pre-defined column widths, and with a pre-defined column used to order the rows in the display. Together, these characteristics (column ordering, column width) make up a *data view*. These data views are referred to as the QM product default view. A user may build a customized view which is referred to as a user-defined data view. This is because it can be useful to present the data in a different way to the default, so that all or most of the data that is most important to the user is presented on the panel when it is first displayed, to minimize sideways scrolling. It is also possible to create a customized view and share that view with all QM users. This view can be referred to as a site default view.

This chapter shows you how to build a customized view of the various QM panel data displays. A user has the ability to change such things as the order of columns, width of the display, and column sort order. Once a user has conveniently set up his / her user-defined preferences, you have the ability to allow all other users to share the same settings yet allow them the flexibility to make their own changes. This chapter describes this process in detail, which is comprised of the following steps:

- How to create a user-defined customized view
- Changing the "Display Exceptions" view
- Where is the user-defined customization information stored?
- How to make the user-defined customized view available to all users

## How to create a user-defined customized view

The following table summarizes the different primary commands and a brief description of its function used to customize a data panel view. Refer to the *IBM DB2 Query Monitor for z/OS User's Guide* for an in depth discussion of these primary commands.

You may use any of these commands to control the visual display of any QM panel where column data is formatted. Examples of QM panels which can be controlled by a user include: "Current SQL Activity", "Display Exceptions", "Operational Summaries", and "Structural Summaries". In fact, all panels where performance data is displayed can be customized (because of the large variety of panels, and because these panels may be added to, changed or removed by product maintenance, we do not list all the customizable panels). For each such type of QM panel, a user may decide to increase or decrease the width of a column, change the order of columns, omit certain columns from

the display, define a descending or ascending sort order to particular columns, and actually fix a column so that if you scroll left or right, these columns remain fixed within the data view.

Primary command	Description
CSETUP	Display the SETUP primary option menu
CFIX	Fix a column in a data view
CORDER	Change the order of columns in a data view
CSIZE	Change the width of a column
CSORT	Change the sort order of a column
CRESET	Restore the settings to "as shipped"
CREMOVE	Remove all customizations for a data view Sets all column sizes to their maximum values
PVIEW	Toggle between a temporary or a permanent view

**Figure 14: Query Monitor column customization primary commands**

As an example, at least one change to the QM product default view is recommended when displaying exceptions. Since there can be different criteria that signals an exception, it is extremely useful to display the column containing the name of the SQL workload for which the profile line applies; this column is under the heading labelled WORKLOAD. Currently the product-supplied default data view has WORKLOAD defined as column #39; so in order to see the column heading, you need to scroll to the right 5 pages until this column appears. It certainly would be extremely beneficial, especially in an environment where many different exception profile lines are defined, to know which exception rule triggered the particular exception, and to immediately see that exception rule without having to scroll.

NOTE: To see recommendations for defining a WORKLOAD name for an exception profile line, refer to the chapter on Data Collection.

Important: The principles used to control the visual display of the data panels is common across some of the tools: DB2 Automation Tool, DB2 Table Editor, DB2 Change Accumulation, etc.

## Changing the "Display Exceptions" view

Here we discuss one method illustrating how to configure a user session to display the exceptions where we fix the column WORKLOAD.

First, choose option E from the main QM panel, CQM\$MAIN to select the "Display Exceptions" panel, CQM\$EXCA, in the default layout. There are eight columns that appear on the first page; the first two columns CMD and SSID are fixed; if you scroll to the right or left, these two columns always appear on the panel.

```

CQM$EXCA 1 16:31:34 ----- Display Exceptions ----- Row 117 of 177
DB2 QM Subsystem: QMV2          Interval Start Date: 02/11/2005  Time: 08:00:04
Filters Enabled: N              Interval End   Date: CURRENT      Time: CURRENT
C:A-Analyze,B-Buffers,C-Calls,D-Delays,L-Locks,O-Objects,S-SQL Text
----- -->
CMD  SSID  Plan      Program      DB2 CPU Time      DB2 Elapsed      GETPAGES  SQLCODE
----- --
-   -      -          -            -      -              -          -
-   DBDM  DSNREXX  DSNREXX      0.000810      0.301390         1,552     0
-   DBDM  DSNREXX  DSNREXX      0.000862      0.298584         1,586     0
-   DBDM  DSNREXX  DSNREXX      0.000817      0.268174         1,595     0
-   DBDM  DSNREXX  DSNREXX      0.000896      0.365430         1,876     0
-   DBDM  DSNREXX  DSNREXX      0.000814      0.327019         1,825     0
-   DBDM  DSNREXX  DSNREXX      0.000749      0.290350         1,647     0
-   DBDM  DSNREXX  PRJADD2     1.012589      1.757802        24,300     0
-   DBDM  DSNREXX  PRJADD2     0.190767      0.303641         1,728     0
-   DBDM  DISTSERV RAARDBX3    0.004212      0.078469         23        -204
-   DBDM  DISTSERV RAARDBX3    0.005714      2:15.517431      55         0
-   DBDM  DISTSERV SYSSH200 0.107373      2.667997         2,717    +100
-   DBDM  DSNESPCS DSNESM68    0.005459      3:24.587419      38        -471
-   DBDM  PLNABCD  PGMABCD     0.011275      0.069121         171       -443
-   DBDM  QMF720  DSQDDYSQ    0.004266      0.006411         82        -551
-   DBDM  QMF720  DSQDDYSQ    0.001689      0.003372         2         -204

Option  ==>>                               Scroll ==>> PAGE

```

Figure 15: The 'Display Exceptions' panel

In order to fix a new column in the display, issue the CFIX command or choose the CFIX option from the CSETUP panel. Either method brings you to the "Define Fixed Columns" panel, FECFIX which is illustrated below.

```

FECCFX ----- Define Fixed Columns ----- 2005/02/11 16:54:19
Option  ==>>                               Scroll ==>> PAGE
----- -->
                                         ROW 1 OF 31

Column Function ==>> 1 (1-Fix/Unfix, 2-Order, 3-Size, 4-Sort)
Permanent View  ==>> Y (Y-Perm, N-Temp)  Reset View ==>> N (Y,N)

Device_Width   : 80
Old_Fixed_Width: 9      Old_Unfixed_Width: 71
New_Fixed_Width:      New_Unfixed_Width:

-----

Cmd  New  Old  Len  Column_Name
P   P   P    4   CMD
P   P   P    5   SSID
-   -   -    9   PLAN
-   -   -    9   PROGRAM
-   -   -   15   DB2_CPU_TIME
-   -   -   15   DB2_ELAPSED_TIME

Enter: Process selections; PF3: Exit and save; CAN: Exit without save
Line Cmds: F Fix U Unfix

```

Figure 16: Defining fixed columns (1)

We want to choose the column labeled WORKLOAD. Scroll down to that column and use the line command F to fix that column in the display. This is illustrated in the next panel.

```

FECFIX ----- Define Fixed Columns ----- 2005/02/11 16:56:26
Option ==>                                     Scroll ==> PAGE
-----+-----
                                                ROW 25 OF 31

Column Function ==> 1 (1-Fix/Unfix, 2-Order, 3-Size, 4-Sort)
Permanent View ==> Y (Y-Perm, N-Temp)  Reset View ==> N (Y,N)

Device_Width   : 80
Old_Fixed_Width: 9           Old_Unfixed_Width: 71
New_Fixed_Width:           New_Unfixed_Width:
-----+-----

Cmd New Old Len Column_Name
-      9 LUNAME
-      8 SECTION
-     11 EXCEPTIONS
-     11 ALERTS
f     33 WORKLOAD
-     27 START_TIME

Enter: Process selections; PF3: Exit and save; CAN: Exit without save
Line Cnds: F Fix U Unfix

```

Figure 17: Defining fixed columns (2)

Once you save the change, and return to the "Display Exceptions" panel, you'll notice that the first three columns are fixed in the display: CMD, SSID, and WORKLOAD. This is illustrated in the next panel.

```

QM$EXCA 1 17:00:36 ----- Display Exceptions ----- Row 117 of 177
DB2 QM Subsystem: QMV2           Interval Start Date: 02/11/2005 Time: 08:00:04
Filters Enabled: N               Interval End Date: CURRENT Time: CURRENT
C:A-Analyze,B-Buffers,C-Calls,D-Delays,L-Locks,O-Objects,S-SQL Text
-----+----- -->

CMD SSID WORKLOAD                      Plan      Program    DB2 CPU Time
-----+-----
- DBDM Generic Bucket                  DSNREXX   DSNREXX    0.000810
- DBDM Generic Bucket                  DSNREXX   DSNREXX    0.000862
- DBDM Generic Bucket                  DSNREXX   DSNREXX    0.000817
- DBDM CPU > 2s Elap > 1m GP > 1500   DSNREXX   DSNREXX    0.000896
- DBDM CPU > 2s Elap > 1m GP > 1500   DSNREXX   DSNREXX    0.000814
- DBDM CPU > 2s Elap > 1m GP > 1500   DSNREXX   DSNREXX    0.000749
- DBDM CPU > 2s Elap > 1m GP > 1500   DSNREXX   PRJADD2    1.012589
- DBDM CPU > 2s Elap > 1m GP > 1500   DSNREXX   PRJADD2    0.190767
- DBDM CPU > 2s Elap > 1m GP > 1500   DISTSERV  RAARDBX3  0.004212
- DBDM CPU > 2s Elap > 1m GP > 1500   DISTSERV  RAARDBX3  0.005714
- DBDM CPU > 2s Elap > 1m GP > 1500   DISTSERV  SYSSH200  0.107373
- DBDM CPU > 2s Elap > 1m GP > 1500   DSNESPCS  DSNESM68  0.005459
- DBDM CPU > 2s Elap > 1m GP > 1500   PLNABCD   PGMABCD   0.011275
- DBDM CPU > 2s Elap > 1m GP > 1500   QMF720   DSQDDYSQ  0.004266
- DBDM CPU > 2s Elap > 1m GP > 1500   QMF720   DSQDDYSQ  0.001689

Option ==>                                     Scroll ==> PAGE

```

Figure 18: The new 'Display Exceptions' panel







NOTE: If you change and save the user preferences for any panel, it will be saved in this dataset, and the member name is precisely the same as the panel name. In addition, the member is in an ISPF dialog manager table format.

### How to make the user-defined customized view available to all users

Now, your customer may want to globally save the same user preferences for all users as a site default. The best way to do this is to alter the CQMCLIST and change the following line:

```
ISPEXEC LIBDEF ISPTLIB DATASET ID('&TL..CQM0210.ISPTLIB') UNCOND
```

In order to include a site-supplied user preference library, concatenate it after '&TL..CQM0210.ISPTLIB' in the CQMCLIST.

As a result, using the library, 'DMMPET.CQM0210.ISPTLIB' as the site-supplied user preference library, the ISPEXEC command above in the CQMCLIST becomes:

```
ISPEXEC LIBDEF ISPTLIB DATASET +  
ID('&TL..CQM0210.ISPTLIB' 'DMMPET.CQM0210.ISPTLIB') UNCOND
```

TIP: If a user modifies the user preferences for a panel, and issues a CRESET, then the user preferences reverts back to the QM product default and not the site's default. If a user wants to reset the user preferences back to the site's defaults, simply have that user delete the associated member name in their own &TL..CQM0210.ISPTLIB' dataset.

Similarly, if a user wants to revert back to the QM product default for a particular panel, merely remove the corresponding member name from the user's &TL..CQM0210.ISPTLIB' dataset. This assumes that a site default dataset has been added to the ISPTLIB concatenation.

TIP: If you are using DB2 V8, consider using the CREMOVE command to set every column to its maximum width. However, remember that because of long names (up to 128 characters for some columns such as NAME and CREATOR in SYSIBM.SYSTABLES), this can create some very wide display columns in the Query Monitor object display panels.

## Chapter 4: Data Collection

The objective of this chapter is to enable the reader to learn how to configure the QM data collector task (or tasks) so that QM will use a minimal amount of resources for the collection process yet still provide users access to the data they need for tuning their SQL. It supplements the information provided in the *IBM DB2 Query Monitor for z/OS User's Guide*.

We discuss the following topics:

- Setting up the Query Monitor data collector so that you can start to collect the performance data necessary for analysis.
- Conceptual ideas such as SQL Workloads, Summary data, Exceptions and Alerts, to provide a background for some of the more detailed discussions later on in the chapter.
- A detailed discussion of CQMPARMS, the parameters that Query Monitor uses to determine what data to collect and in what level of detail.
- A detailed discussion on setting up monitoring profiles, including advice on setting exception and alert thresholds, excluding negative SQL codes from analysis, and on building profile lines.

First, however, we provide a quick overview of data collection, using the following diagram:

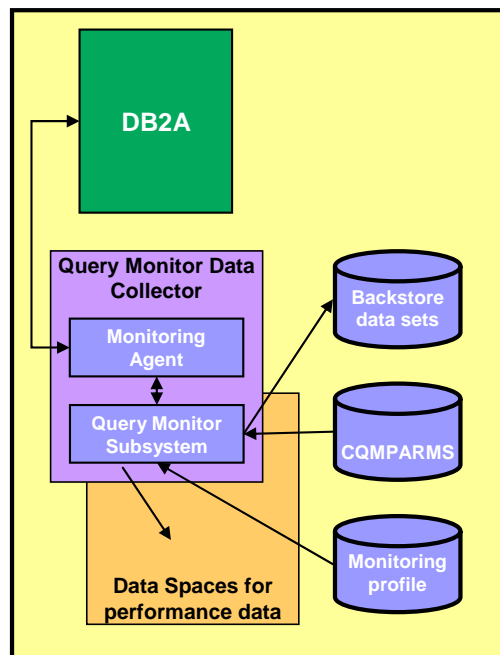


Figure 22: Data Collection Overview

Query Monitor uses the CQMPARMS data set and the monitoring profile to determine what performance data to collect from any given DB2 subsystem. Performance data is collected from the monitored DB2 subsystem by a

monitoring agent, and is for the most part stored in z/OS data spaces until a Query Monitor recording interval is completed, when the data is written to the VSAM backstore data sets.

## Setting up the Query Monitor Data Collector

The process for setting up the data collector involves two steps. Firstly, to gather performance data to use as a base to setup monitoring profiles with reasonable exception and alert thresholds. Secondly, to create a monitoring profile with workload definitions appropriate to the work being monitored based on the data gathered in step one.

If you already have existing SQL metrics such as average elapsed time, average CPU time, number of SQL calls, etc, these can be used to determine an intelligent first cut at a monitoring profile workload. Such metrics are typically obtained from SMF data, often using a tool like DB2 Performance Expert for z/OS. The base profile you derive from this data can be in place the first time the QM data collector is started.

On the other hand, if there is not an existing base of SQL metrics, then QM can be used to gather information about SQL volume, average CPU use, average elapsed time, and negative SQLCODEs from your systems. In this case, you will need at least two half-days to properly configure QM:

- Allocate at least half a day to spend customizing QM and getting the data collector task running. The initial setup and configuration should have the QM data collector monitoring one or more DB2 subsystems without specifying a monitoring profile. In this configuration the QM data collector task is gathering summary data only. There will not be any data in exceptions, alerts, or current activity. Set MAX\_SQLCODES and MAX\_SQLCODE\_DETAIL in CQMPARMS to 250 and between 50 and 100 as recommended in the section on CQMPARMS below, ensuring that data about negative SQLCODEs is collected. The QM Data Collector will issue the following message (where 'ssid' is the DB2 subsystem name), which can be ignored:

```
CQM3302I **WARNING** MONITORING AGENT FOR ssid WILL NOT COLLECT EXCEPTION DATA OR  
CURRENT ACTIVITY
```

- Once QM is configured to collect summary data, let it run for 24 hours, if possible.
- By the following day, there should be plenty of data available for initial analysis in the Activity Summaries and SQLCODE sections of the QM user interfaces. This data can then be reviewed to determine appropriate thresholds for workload definition lines when building the monitoring profile.

## Number of Data Collector Tasks

QM data collectors can monitor up to 64 DB2 subsystems on a single z/OS LPAR. This leads to the question of how many QM data collector tasks does a user need.

In general, there should be a single QM data collector for each z/OS LPAR. Reasons for having more than one collector per LPAR include the following:

1. Differences in startup parameters required for different DB2 workloads.

For example, in some environments, some OPTKEYS may be useful and others may not. Since OPTKEYS is a startup parameter, a different collector task would be required for each different set of OPTKEYS.

A classic example of this would be an ad-hoc, query-based DB2 and an OLTP-based DB2.

In the ad-hoc DB2, OPTKEYS(TEXT) is probably not useful as most of the SQL in the systems will be unique and not reused. OPTKEYS(AUTHID) may be useful in this same system as the number of users is most likely relatively small.

In the OLTP based DB2, if the dynamic SQL is repeated, OPTKEYS(TEXT) may be very useful, whereas OPTKEYS(AUTHID) would most likely not be at all useful. If the number of distinct AUTHIDs is large, this will cause much overhead and minimal summarization. If there is a single AUTHID used for all DB2 SQL then all of the summary will be in a single bucket anyway, and therefore there is no value in using the OPTKEYS.

2. Limiting the QM user's view of DB2 subsystems being monitored.

Access to the QM data from a given data collector can be restricted using an external security system such as RACF. If an installation decides to setup a specific QM data collector for each DB2 subsystem to be monitored, users could be restricted to only being allowed to access QM collectors for DB2 subsystems where the user is also authorized.

Note: Users of the CAE GUI will have access to QM data on all DB2 subsystems on all LPARS which are connected to the CAE Server.

## Workloads

Throughout this chapter, you will references to a concept called a 'workload' or 'SQL workload', so now is a good time to give a brief explanation as to what they are. A workload is a means of identifying a group of applications to Query Monitor so that performance data can be collected for SQL statements executed by those applications. Workloads are defined in the monitoring profile used by Query Monitor to monitor a given DB2 subsystem. Workloads can be said to have three characteristics:

- The workload name, which must be unique and is used purely to identify the workload.
- The workload filters, in which identifiers such as plan name, subsystem name, authorization id and so on are used to identify which applications to include or exclude from data collection.
- The thresholds for collecting exceptions and alerts, such as elapsed time, CPU time or GETPAGES.

## Summary data compared to exceptions and alerts

The QM data collector maintains three basic types of performance data. The data types are summary, exceptions, and alerts.

### Summary Data

Summary data is exactly that, it is performance data summarized for each unique SQL statement executed in a Query Monitor recording interval. The values collected are totals and averages; for example, the elapsed time values are averages. A unique SQL statement is represented by a unique value of:

Plan + Program + Section + Statement number + Statement Type

When QM is first installed, by default, partial summary data will be collected for all SQL executed in the DB2 subsystem(s) being monitored. SQL from specific workloads can be excluded from summary collection by using the monitoring profiles. See the section on monitoring profiles for additional information.

Summary data which is not collected by default is the negative SQLCODE data. This collection must be activated using the appropriate CQMPARMS startup parameters.

QM Summary data can be viewed from both of the QM user interfaces. The areas are:

1. For the ISPF interface:
  - a. From the "View Activity Summaries" Query Monitor primary menu option
  - b. From the "View SQLCODEs" Query Monitor primary menu option
2. For the CAE GUI activity browser:
  - a. Using a perspective of "Structural Summaries"
  - b. Using a perspective of "Operational Summaries"
  - c. Using a perspective of "SQLCODES"

### Exception Data

QM Exception data is data about individual SQL calls that have exceeded user-defined thresholds - for example, elapsed time or CPU time (setting up

these is described later in this chapter). These thresholds are defined via a monitoring profile - with no monitoring profile, no exception data is collected. Exception data can be viewed from both of the QM user interfaces. The areas are:

1. For the ISPF interface:
  - a. From the "View Exceptions" Query Monitor primary menu option
2. For the CAE GUI activity browser:
  - a. Using a perspective of "Exceptions"

## Alert Data

Query Monitor Alerts are SQL events which require immediate attention. As for exceptions, QM Alert data is data about individual SQL calls that have exceeded user-defined thresholds. Alerts can also be classified as exceptions - this is the recommendation, as the alert/exception data is written to the VSAM backstore data sets and is therefore available for later analysis. It is possible, but not recommended, to have Alerts which are only viewable through the CAE GUI user interface, by only defining them as alerts but not as exceptions. Alert specifications can be identical to the corresponding exception specifications, thereby generating alerts and exceptions simultaneously, but this may mean you either generate large numbers of 'immediate attention' alerts or have too narrow a view of what constitutes an exception SQL event. Alert specifications are discussed in detail later in the chapter, but we mention now that when defining thresholds for alerts that the alert thresholds be higher than the exception thresholds. To reiterate, alerts are SQL events which require immediate attention, whereas this is not necessarily the case with exceptions.

Note: Alerts which do not also qualify as exceptions will not be available in the "View Exceptions" area of the ISPF interface or the "Exceptions" perspective of the CAE GUI activity browser. The alerts are not stored in the VSAM back-store data sets, and therefore will not be available in the QM Performance Database, if used. For this reason, every alert should also qualify as an exception.

Alert data, depending on how the monitoring profile is specified, may be available through both of the QM interfaces. The areas to view Alert data are:

1. For the ISPF interface:
  - a. From the "View Exceptions" Query Monitor primary menu option, if the Alert is also an Exception
2. For the CAE GUI Activity Browser:
  - a. Using a perspective of "Exceptions" if the Alert is also an Exception
3. For the CAE GUI Message Board:



- a. Any Alert not specifically excluded by the QM user

## CQMPARMS

### Use a PDS for CQMPARMS

Query Monitor uses a set of startup parameters which define how Query Monitor is implemented, including the Query Monitor subsystem name, the monitored DB2 subsystems, and the length of the recording interval. These parameters are stored in a data set allocated to the CQMPARMS DD statement in the Query Monitor JCL. This data set should be allocated as a partitioned data set (PDS) with a separate member for each set of startup parameter definitions. Each individual QM data collector task will need to have a separate set of startup parameters. Using a PDS allows all of the startup parameters for the various QM collectors to be stored in a single data set. In addition, the individual members can be **edited** while the collector task is active. If a sequential data set is used for each collector, the parameters can only be changed while the collector task is shut down.

## Sample CQMPARMS

An example of the contents of the CQMPARMS file is shown below:

```
AUTHID(DB2USER) -
MONITOR(DB2A,DB2APROF,DB2B,DB2BPROF) -
SUBSYS(I71A) -
INTERVAL(60) -
RETAIN(96) -
OPTKEYS(TEXT) -
ALERT_LIMIT(100) -
MAX_SQLCODES(250) -
MAX_SQLCODE_DETAIL(100) -
STORCLAS(DB2TEMP) -
MGMTCLAS(DB2) -
DATACLAS(VSHAR33) -
EXCPDATA_DSN(CQMHLQ.I71A.EDATA.D&YYMMDD..T&LHR.&LMIN..&INTV.) -
EXCPINDX_DSN(CQMHLQ.I71A.EINDX.D&YYMMDD..T&LHR.&LMIN..&INTV.) -
METRDATA_DSN(CQMHLQ.I71A.METRD.D&YYMMDD..T&LHR.&LMIN..&INTV.) -
OBJSDATA_DSN(CQMHLQ.I71A.OBJSD.D&YYMMDD..T&LHR.&LMIN..&INTV.) -
TEXTDATA_DSN(CQMHLQ.I71A.TEXTD.D&YYMMDD..T&LHR.&LMIN..&INTV.) -
SQLCDATA_DSN(CQMHLQ.I71A.SQLCD.D&YYMMDD..T&LHR.&LMIN..&INTV.) -
DB2CDATA_DSN(CQMHLQ.I71A.DB2CD.D&YYMMDD..T&LHR.&LMIN..&INTV.) -
EXCPDATA_SPACE_UNITS(TRKS) -
EXCPINDX_SPACE_UNITS(TRKS) -
METRDATA_SPACE_UNITS(CYLS) -
OBJSDATA_SPACE_UNITS(CYLS) -
TEXTDATA_SPACE_UNITS(CYLS) -
SQLCDATA_SPACE_UNITS(TRKS) -
DB2CDATA_SPACE_UNITS(TRKS) -
EXCPDATA_PRIMARY(45) -
EXCPINDX_PRIMARY(15) -
METRDATA_PRIMARY(5) -
OBJSDATA_PRIMARY(5) -
TEXTDATA_PRIMARY(3) -
SQLCDATA_PRIMARY(15) -
DB2CDATA PRIMARY(15) -
```

Figure 23: Sample CQMPARMS

## CQMPARMS Parameters

We now move on to a description of some of the most important parameters contained in CQMPARMS.

### AUTHID

The AUTHID parameter is intended for temporary use until the QM data collector task has been properly defined to your security system (e.g. RACF). Once the data collector task has been set up in the security system, a USERID is assigned to the started task associated with the QM Data Collector. Until such time, use the AUTHID parameter to specify the authorization id to use for the data collector. The USERID associated with the data collector task (or the id specified in the AUTHID parameter) needs

authorization to connect to the DB2 subsystems being monitored, and EXECUTE authority on the QM plan on those DB2 subsystems. Once security has been set up properly, the AUTHID parameter should be removed from COMPARMS, or commented out.

## MONITOR

The MONITOR parameter identifies the DB2 subsystems to be monitored by the QM data collector when the task is started. The subsystems being monitored can be changed using the "Work With Monitoring Agents" option of the ISPF interface or the DB2 Monitoring Configuration Editor in the CAE GUI.

The initial monitoring profile for each DB2 subsystem is also specified here. If no monitoring profile is specified at startup time, only summary data will be collected for the subsystem and QM message CQM3302I will be issued for that DB2 subsystem.

Note 1: There must be at least one DB2 subsystem specified on this parameter. This is a required parameter.

Note 2: If monitoring is started on additional DB2 subsystems after the QM data collector is started or the monitoring profile name is changed for a particular DB2, those changes must also be made to the MONITOR startup parameter in order for the change to be made persistent across a restart of the QM data collector.

## INTERVAL and RETAIN

The INTERVAL and RETAIN startup parameters are related to each other. Changing either parameter may necessitate a change to the other parameter.

The INTERVAL parameter defines the length of the recording intervals in minutes. Shorter intervals such as 60 minutes are recommended to allow for the correlation of data collected by QM with other performance products such as DB2 account data, RMF data, OMEGAMON for MVS data, and other performance data sources. Shorter intervals also create visibility to peaks and valleys in the SQL elapsed times because the data is summarized by interval.

The RETAIN parameter defines how many prior INTERVALs are to be retained on DASD after interval processing is complete.

Together these parameters control the time period for which data can be accessed through the QM user interfaces.

For example, a common specification in customer sites of INTERVAL(60) and RETAIN(96) will cause QM to keep 4 days (96 one hour time slices) available to the user interfaces. This is a useful quantity of data because it allows for data to be available to the user interfaces after a holiday weekend. When a QM user arrives on Tuesday after a long weekend, the data for Friday will just be starting to roll off. Also, the data for Saturday will remain available for the whole day on Tuesday. Another reason for using shorter intervals is that each

time interval processing is run the oldest interval worth of data is deleted. Shorter intervals contain less data per interval and therefore less data is deleted by interval processing.

Another example would be, a specification of INTERVAL(30) and RETAIN(192) will also cause QM to keep 4 days (in this case 192 30 minute time slices) available to the user interfaces.

## OPTKEYS

QM collects and summarizes SQL activity using the following basic key:

Plan + Program + Section + Statement number + Statement type

The OPTKEYS parameter allows QM users to specify additional levels of summarization. As these additional levels can significantly increase the volume of data collected by QM and stored in its data spaces, thereby increasing real storage usage, care must be taken when adding OPTKEYS.

As discussed earlier, a classic example of this would be an ad-hoc query-based DB2 and an OLTP based DB2. In the ad-hoc DB2, OPTKEYS(TEXT) is probably not useful, as most of the SQL in the systems will be unique and not reused. OPTKEYS(AUTHID) may be useful in this same system as the number of users is most likely relatively small.

Note: Do not be misled into thinking that omitting TEXT from the OPTKEYS specification means that the text of exception dynamic SQL statements will be lost. The SQL statement text associated with exception events is always recorded.

In the OLTP based DB2 if the dynamic SQL is repeated, OPTKEYS(TEXT) may be very useful, whereas OPTKEYS(AUTHID) is most likely not at all useful. If the number of distinct AUTHIDs is large, this will cause more overhead and minimal summarization. If there is a single AUTHID used for all DB2 SQL then all of the summary will be in a single bucket any way and there is no value in using the OPTKEYS. If there is no dynamic SQL, there is no benefit in using OPTKEYS(TEXT), because the default summarization key is enough to identify each SQL statement being executed.

The number of summary buckets grows quickly and this can be exacerbated by specifying multiple OPTKEYS. For example, let's assume that a system has 1,000 distinct dynamic SQL statements and 1,000 users. Also assume that each user will execute every SQL statement at least once during each interval. Finally add into the assumptions that each SQL statement accesses 3 application objects plus the 12 objects needed to PREPARE a dynamic SQL statement for execution. For this example, the OPTKEYS will affect the summary collection as follows:

- Specifying OPTKEYS(TEXT) - This adds 1,000 summary buckets to the METR data space and 15,000 (1,000\*15) buckets to the OBJS data space.
- Specifying OPTKEYS(AUTHID) - This adds 1,000 summary buckets to the METR data space and 15,000 (1,000\*15) buckets to the OBJS data space.
- Specifying OPTKEYS(TEXT,AUTHID) - This adds 1,000,000 (1,000\*1,000) summary buckets to the METR data space and 15,000,000 (1,000\*1,000\*15) buckets to the OBJS data space.

When determining your OPTKEYS settings, it is important to determine what categories of summarization are meaningful and useful in your environment. For example, with SAP, in contrast to the settings for the OLTP-based DB2 and the Data Warehouse-based DB2 discussed above, it is probably more appropriate to summarize by WSTRAN and TEXT. The reason for this is that there is only one AUTHID used by SAP (usually SAPR3), whereas it is WSTRAN that helps you identify the user. Since SAP only uses dynamic SQL which is subject to repeated execution, TEXT is needed to be able to summarize by SQL statement. Don't forget, however, that these both act as multipliers to the number of summary buckets, so a large number of users may mean that specifying WSTRAN is undesirable.

#### **MAX\_SQLCODES and MAX\_SQLCODE\_DETAIL**

The MAX\_SQLCODES and MAX\_SQLCODE\_DETAIL parameters control the summary data collection for negative SQLCODEs. The summary data is accessed using the "View SQLCODEs" option in the ISPF interface and the "SQLCODES" perspective in the CAE GUI.

The MAX\_SQLCODES parameter sets the limit on the number of unique SQLCODES for which summary information is collected.

The MAX\_SQLCODE\_DETAIL sets the limit on the number of detail records which is collected for each occurrence of a negative SQLCODE.

Note: Because we are dealing with summary information in this area, the detail collected is very limited. The detail collected consists of the SQLCA and the text of the SQL statement, if the statement text is available. There is no performance data or host variable information available in this part of the QM product. Host variables and performance metrics for statements which end with negative SQLCODES are kept with the exception record for the event.

A recommended starting value for MAX\_SQLCODES is 250. This will most likely be larger than the number of distinct negative SQLCODES in a given interval.

A recommended starting value for MAX\_SQLCODE\_DETAIL is between 50 and 100. This will normally allow a QM user to determine which negative SQLCODES are being used as coding techniques by which programs. Once

the codes being used as coding techniques are identified, they can be excluded from exception and alert processing in the monitoring profile. How to do this is discussed in detail in the section dealing with setting up monitoring profiles.

## **ALERT\_LIMIT**

The ALERT\_LIMIT parameter defines the maximum number of alerts which are to be queued for the CAE Agent alert processor. When the specified limit is reached, no more alerts are queued (all subsequent alerts are discarded) until alerts have been removed from the queue by the CAE Agent. For additional details regarding this parameter, see Appendix B of the DB2 Query Monitor User's Guide.

The number of alerts generated by SQL in a given DB2 subsystem should be relatively small. This is because alerts are important events which require an immediate reaction.

Do not confuse ALERT\_LIMIT with Exception Limit, which is specified in monitoring profile lines. The ALERT\_LIMIT parameter operates at the QM Data Collector level, whereas Exception Limit operates at the workload or monitoring profile line level.

## **CATALOG\_OBJECTS**

A new CQMPARMS parameter, CATALOG\_OBJECTS, is being made available via APAR PQ99869, so that the amount of dataspace storage used for DB2 object statistics per interval is minimized for PREPARE SQL calls. Data about catalog objects used when preparing dynamic SQL is not stored if CATALOG\_OBJECTS is set to "NO". The default value is "YES".

## **OBJECTS**

The OBJECTS parameter allows you to control the collection of statistics for all objects. If you specify OBJECTS(N), then no statistics are gathered. The default is OBJECTS(Y). You may find this parameter useful as a means of controlling the size of the Query Monitor data spaces. Be aware, however, that this is an all-or-nothing parameter and affects all DB2 subsystems monitored by the QM data collector.

### **Data Set Sizing**

The allocation specifications for the VSAM back-store data sets are also in the CQMPARMS data set. Typically, once the parameters are set up, they are seldom if ever changed. Refer to Appendix D of the QM User's Guide for complete instructions on calculating the space for these data sets. You will need to know the volume and mix of SQL being executed in order to perform the calculations. This can be determined from several sources such as:

- DB2 PE statistical reports
- Data from an SQL monitor product (such as IBM DB2 Query Monitor)

- Other DB2 capacity planning products.

If the volume and mix of SQL in the workload can not be easily determined, a simple method is to use the default allocations in whole cylinders. Track the number of extents used by the data sets during the first few days or weeks that QM is running. If the data sets take multiple extents during every interval, then increase the primary allocation until the data will fit in a single extent during intervals when the SQL volume is low.

During intervals when SQL volume is highest, the back-stores should always be allowed to take multiple extents. This help to minimize the DASD used for the back-stores because it avoids over-allocation of the back-store data sets for intervals when SQL volumes are low.

SMS management is highly recommended for the VSAM back-store data sets. There are seven unique data sets created for each interval. Using a RETAIN parameter of 96 will result in 672 VSAM back-store data sets being created and retained.

## Monitoring profiles

Monitoring profiles fulfil three basic functions in QM. Those functions are:

1. To define exception and/or alert thresholds for particular SQL workloads
2. To activate the current activity display for particular SQL workloads
3. To Exclude particular SQL workloads from summary data collection

It is a recommended practice to have a separate monitoring profile for each DB2 subsystem being monitored. This facilitates limiting the number of workload definition lines in each of the profiles which may possibly reduce overhead in the profile search process.

Monitoring profiles are made up of one or more workload definition lines. Each workload definition line consists of the following components:

1. Line type (Include or Exclude)
2. Miscellaneous flags
3. Workload definition
4. Exception thresholds
5. Exception limit
6. SQL codes excluded from exceptions
7. Alert thresholds
8. SQLCODEs excluded from alerts

A sample of the "Update Profile Line" panel is shown below

```

----- Update Profile Line for SAMPLE1 -----
Option ==> _____ Scroll ==> PAGE

INCLUDE/EXCLUDE          I      (I=Include, E=Exclude)
Disable Summary Reporting N (Y/N)  Gather Host Variables Y (Y/N)
DB2 Subsystem            * _____ Plan Name      * _____
Program Name             * _____
AUTHID                   * _____ JOBNAME        * _____
Connection ID            * _____ CORRID          * _____
Workstation User         * _____
Workstation Trans        * _____
Workstation Name         * _____
Workload Name            All other work _____
Exception CPU            00 : 00 : 00 . 040000
Exception Elapsed        00 : 00 : 00 . 100000
Exception Getpages       0 _____
Exception SQL Calls      0 _____
Exception Limit          1000 _____
Exclude Exception SQLCODES N (Y/N)
Alert CPU                 00 : 00 : 00 . 100000
Alert Elapsed            00 : 00 : 01 . 000000
Alert Getpages           0 _____
Alert SQL Calls          0 _____
Exclude Alert SQLCODES  N (Y/N)

```

Figure 24: The 'Update Profile Line' panel

### Include/Exclude

The Include/Exclude flag indicates what QM is to do with SQL which matches the workload definition for this profile line. 'Include' indicates that matching SQL should be included in further exception/alert processing. 'Exclude' indicates that matching SQL should be excluded from further exception/alert processing.

Note: Excluding SQL from exception/alert processing will also exclude the matching SQL from the current activity display.

### Miscellaneous Flags

#### Disable Summary Reporting

The "Disable Summary Reporting" flag is only used for profile lines which are defined as exclude lines. Setting this flag to "Y" will exclude matching SQL from summary, exception, alert processing and the current activity display. In other words, QM will not perform any monitoring of any SQL which matches the workload definition in an exclude line with this flag set to "Y".

#### Gather Host Variables

The "Gather Host Variables" flag is used to tell QM that host variables should be gathered when an SQL statement meets the following conditions:



For exceptions:

1. The workload definition for this line is matched by the SQL  
and
2. The exception limit has not be reached for this profile line  
and
3. One or more of the following is true:
  - a. The SQL exceeds one or more of the specified exception thresholds  
or
  - b. The SQL statement completed with a negative SQLCODE which is not in the "Exclude Exception SQLCODEs" list

For alerts:

1. The workload definition for this line is matched by the SQL  
and
2. One or more of the following is true:
  - a. The SQL exceeds one or more of the specified alert thresholds  
or
  - b. The SQL statement completed with a negative SQLCODE which is not in the "Exclude Alert SQLCODEs" list

### **Workload Definition**

The workload definition section of the profile consists of the following fields:

- a. DB2 Subsystem
- b. Plan Name
- c. Program name
- d. AUTHID
- e. JOBNAME
- f. Connection ID
- g. CORRID

- h. Workstation User
- i. Workstation Trans
- j. Workstation Name
- k. Workload Name

Each of these fields is described in the 'Working with profiles' chapter of the DB2 Query Monitor User's Guide.

When creating a workload definition, the attributes of the SQL statement must match all of the values specified in fields a through j listed above. In other words, the fields are "anded" together to evaluate if the SQL statement match the workload definition. It works something like this:

If **DB2 Subsystem** is a match and **Plan Name** is a match and **Program Name** is a match and **AUTHID** and **JOBNAME** is a match and **Connection ID** is a match and **CORRID** is a match and **Workstation User** is a match and **Workstation Trans** is a match and **Workstation Name** is a match, then the SQL statement matches this workload definition. **Once a match is made, no additional workload definition lines are searched.** For all these fields, the asterisk (\*) is a wild-card character.

The Workload Name entry is not used in evaluating whether the SQL matches the workload definition. However, the value in the Workload Name field is included in any exception and/or alert records created for SQL statements matching the workload definition. This field should always be filled in as it helps identify which monitoring profile line matched the characteristics of the SQL statement.

### Exception Thresholds

These are for user defined thresholds used to determine if Query Monitor should consider an SQL statement an exception. The thresholds are:

- a. Exception CPU
- b. Exception Elapsed
- c. Exception Getpages
- d. Exception SQL Calls

These thresholds are evaluated independently. Therefore, if any one of the thresholds is exceeded, the SQL statement is considered an exception. Setting the value of any threshold to zero **turns exception checking off** for the specified threshold.

### Exception Limit

The exception limit defines the maximum number of exceptions which will be captured for SQL matching this lines' workload definition during a specific interval. The exception counters are automatically reset during interval

processing. For example, a value of 100 will limit the number of exceptions captured for this particular workload to 100 for each interval.

Note: If exception limit is set to zero, then no exceptions will be captured for this workload. This is not a 'no limit' setting.

The exception limit is useful for preventing QM from capturing an excessive volume of exceptions which might be caused by some outside influence. In theory, once a certain volume of exceptions is captured for a given workload, the QM user should be able to determine and fix the problem with the available data. Therefore, no additional notification of these exceptions is necessary.

### Exclude Exception SQLCODEs

By default, all SQL statements which complete with a negative SQLCODE are treated as exceptions (and also alerts). If an installation is using some negative SQL codes as coding techniques, those negative SQL codes can be excluded from exception processing.

Note: If negative SQL codes are excluded from exception processing, the same codes should also be excluded from alert processing. This is important as alerts are sent to the CAE Server and **not** retained in the QM VSAM back-store data sets unless, as recommended, the alert also qualifies as an exception.

### Alert Thresholds

There are four user defined thresholds which determine if QM should generate an alert for a SQL statement. The thresholds are:

- a. Alert CPU
- b. Alert Elapsed
- c. Alert Getpages
- d. Alert SQL Calls

These thresholds are evaluated independently. Therefore, if any one of the thresholds is exceeded, an alert is generated and passed to the CAE Agent for the SQL statement. Setting the value of any threshold to zero **turns alert checking off** for the specified threshold.

### Exclude Alert SQLCODEs

By default, alerts are generated for all SQL statements which complete with a negative SQL code. If an installation is using some negative SQL codes as

coding techniques, those negative SQL codes can be excluded from alert processing.

Note: If negative SQL codes are excluded from exception processing, the same codes should also be excluded from alert processing. This is important as alerts are sent to the CAE Server and **not** retained in the QM VSAM back-store data sets unless as recommended the alert also qualifies as an exception.

## Building Profile Lines

In the 'Working with profiles' chapter of the DB2 Query Monitor User's Guide, there is a long description of the workflow for creating monitoring profiles.

Take note that the profile lines are evaluated in the sequence they appear in the monitoring profile. The first matching workload definition line, and only the first matching line, is used to evaluate what to do with the SQL statement. Therefore the sequence of the lines in the monitoring profile is very important.

Only fields in the workload definition section of the profile line are used for matching criteria.

The search is ended once a workload definition line is matched, whether or not the SQL statement qualifies for an exception or alert based on the thresholds in that profile line. In other words an SQL statement will be evaluated against the thresholds on one and only one workload definition line.

The workload definition lines should be placed in the profile with the most frequently matched line first and in order by decreasing frequency of use. There are exceptions to this rule. If there is a generic "catch all" line with an asterisk (\*) for all matching criteria, that line must be placed last in the sequence. If there are lines with more specific criteria, then these should be placed before those with less specific criteria. For example, a profile line that matches on plan name DSNTEP71 should be placed before one that matches on the more generic plan name of DSN\*. Or, a profile line that matches on program name DSN\* and AUTHID FRED should be placed before one that matches only on program name DSN\*. Once a workload definition line in the monitoring profile is matched, the search is over and subsequent lines in the profile will not be checked for the SQL statement.

A sample profile is shown below:

```

2005/02/11 06:50:48 ----- Update Monitoring Profile ----- Row 1 of 8
Option ==> Scroll ==> Prof
Profile Name: SAMPLE2

C:A-After,B-Before,C-Copy,D-Delete,I-Insert,M-Move,R-Repeat,U-Update
----->
CMD  WORKLOAD NAME                INCL\EXCL  SSID  JOBNAME  Plan  Program
-----
-    CICS Transactions             I         *    CICS*   *     *
-    IMS TM Work                   I         *    IMS*   *     *
-    Human Resources batch work    I         *    HR*    *     *
-    Accounts Payable batch work   I         *    AP*    *     *
-    Exclude DB2 Performance Monitor E         *    *      *     DGO*
-    QMF work                       I         *    *      QMF*  *
-    QMF for Windows work          I         *    *      RAA*  *
-    All other work                I         *    *      *     *
***** Bottom of Data *****

```

**Figure 25: The 'Update Monitoring Profile' panel (1)**

Using the monitoring profile shown above the following will be true:

Work coming in with a job name beginning with CICS will use thresholds set in the 1<sup>st</sup> profile line.

Work coming in with a job name beginning with IMS will use thresholds set in the 2<sup>nd</sup> profile line.

Work coming in with a job name beginning with HR will use thresholds set in the 3<sup>rd</sup> profile line.

Work coming in with a job name beginning with AP will use thresholds set in the 4<sup>th</sup> profile line.

Work coming in with a program (package/DBRM) name beginning with DGO and a job name not beginning with (CICS, IMS, HR, and AP) will use thresholds set in the 5<sup>th</sup> profile line.

Work coming in with a PLAN name beginning with QMF and (a job name not beginning with (CICS, IMS, HR, and AP) and a program (package/DBRM) not beginning with DGO) will use thresholds set in the 6<sup>th</sup> profile line.

Work coming in with a PLAN name beginning with RAA and (a job name not beginning with (CICS, IMS, HR, and AP) and a program (package/DBRM) not beginning with DGO) will use thresholds set in the 7<sup>th</sup> profile line.

All other work will use thresholds set in the 8<sup>th</sup> profile line.

Here is another sample profile:

```

2005/02/11 06:56:17 ----- Update Monitoring Profile ----- Row 1 of 4
Option ==>> Scroll ==>> PAGE
Profile Name: SAMPLE3

C:A-After,B-Before,C-Copy,D-Delete,I-Insert,M-Move,R-Repeat,U-Update
----->
CMD  WORKLOAD NAME                INCL\EXCL  SSID  JOBNAME  Plan  Program
-----
-   -----
-   All other work                 I         *    *        *    *
-   Exclude DB2 Performance Monitor E         *    *        *    DGO*
-   QMF work                       I         *    *        QMF*  *
-   QMF for Windows work          I         *    *        RAA*  *
***** Bottom of Data *****

```

Figure 26: The 'Update Monitoring Profile' panel (2)

In the profile above, only the 1<sup>st</sup> line of the profile is used. Because every SQL statement will match the workload definition specified in the 1<sup>st</sup> line, the search will always end with that line.

**How to Build Profile Lines**

When designing and building profile lines, some techniques can reduce the work required to create the individual line items. This is especially true as it relates to the excluding of SQL codes from exception and alert processing.

It is likely that most but probably not all negative SQLCODEs will be excluded from exception and from alert processing. Since only the first matching profile line is used for evaluating the SQLCODEs to be excluded, the list of exclude alert SQLCODEs must be included on every profile line in order to be sure the SQLCODEs are excluded for all workloads.

The simplest way to build the list is as follows:

1. Allow QM to collect summary negative SQLCODE data for about a day.
2. Load the collected SQLCODE data into the QM DB2 Performance database.
3. Query the QM Performance database to get a list of all the distinct negative SQL codes collected.
4. Build the most generic line profile line first. In the example shown above, that means the last line.
  - a. Specify "Y" for "Exclude Exception SQLCODEs" and for "Exclude Alert SQLCODEs".
  - b. Enter the list of SQLCODES from item 3 on the panel headed "Exception SQL Code Exclusion List", and then hit F3.
  - c. Enter the same list on the panel headed "Alert SQL Code Exclusion List".
  - d. Save the workload definition line

Now you have a single line in the profile which has the complete list of SQLCODEs to be excluded from exception and alert processing. Adding

additional line items with the SQLCODE list intact can be accomplished using the following procedure:

1. Replicate the generic line.
2. Edit the new line and change the workload definition, thresholds and exclude SQLCODEs as appropriate. Remember, you started with a complete list of excludes when the line was replicated.
3. Move the new line to the appropriate place in the monitoring profile.

### **Updating Monitoring Profiles**

When you have updated and activated or refreshed a monitoring profile, then it is prudent to cause an interval to be taken otherwise you may not quite see the change you expect.

## Chapter 5: GUI Client Usage

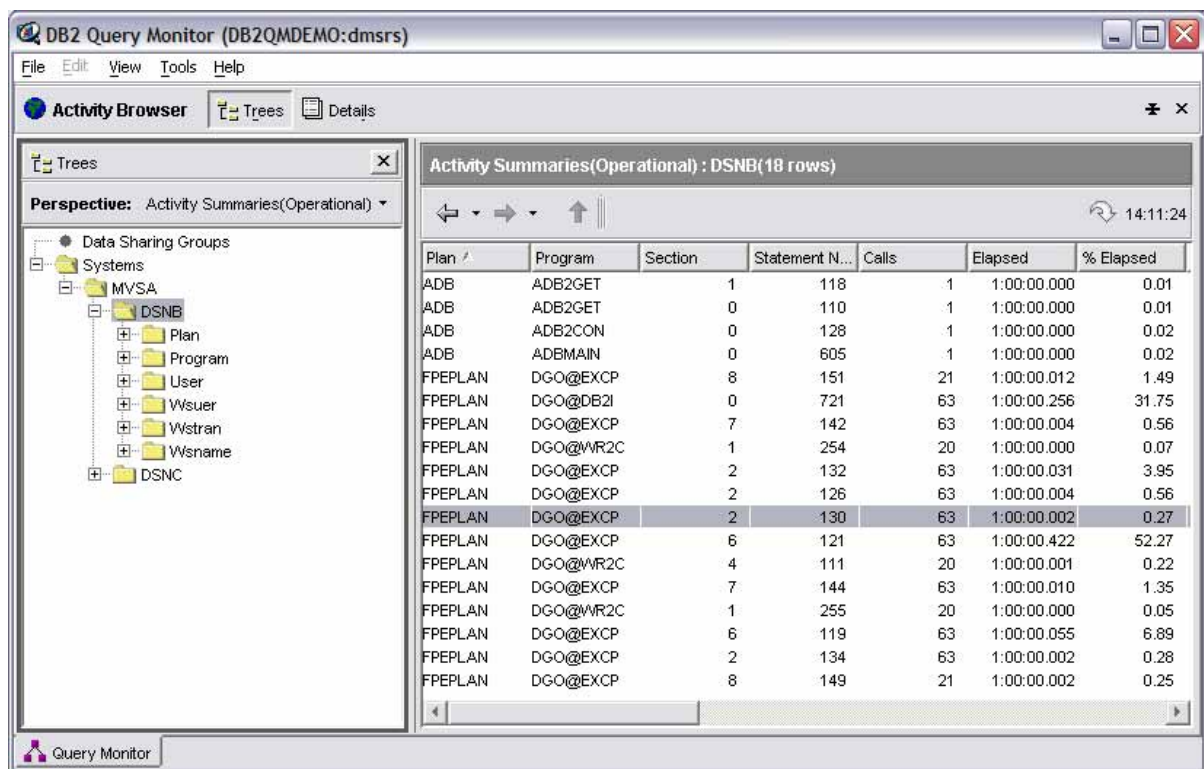
This chapter covers the administration of monitoring profiles, setting up alerts and exceptions, and the use of message boards in the Alert Browser using the QM GUI Client client.

The GUI Client offers several advantages over the ISPF UI.

1. It is the only place that you will see alerts<sup>1</sup>
2. It is the only place that you can set up responses to alerts (e-mail, scripts, etc.)
3. It is a way to view multiple DB2 systems on multiple LPARs, for example members in a DB2 data sharing group

### Using the GUI Client to obtain SQL performance metrics to define thresholds for exceptions and alerts

When you are defining exceptions and alerts, the easiest way to do this is to start by expanding the Activity Browser system tree, using the Activity Summaries(Operational) perspective, and click on the subsystem name.



Plan	Program	Section	Statement N...	Calls	Elapsed	% Elapsed
ADB	ADB2GET	1	118	1	1:00:00.000	0.01
ADB	ADB2GET	0	110	1	1:00:00.000	0.01
ADB	ADB2CON	0	128	1	1:00:00.000	0.02
ADB	ADBMAIN	0	605	1	1:00:00.000	0.02
FPEPLAN	DGO@EXCP	8	151	21	1:00:00.012	1.49
FPEPLAN	DGO@DB2I	0	721	63	1:00:00.256	31.75
FPEPLAN	DGO@EXCP	7	142	63	1:00:00.004	0.56
FPEPLAN	DGO@WR2C	1	254	20	1:00:00.000	0.07
FPEPLAN	DGO@EXCP	2	132	63	1:00:00.031	3.95
FPEPLAN	DGO@EXCP	2	126	63	1:00:00.004	0.56
FPEPLAN	DGO@EXCP	2	130	63	1:00:00.002	0.27
FPEPLAN	DGO@EXCP	6	121	63	1:00:00.422	52.27
FPEPLAN	DGO@WR2C	4	111	20	1:00:00.001	0.22
FPEPLAN	DGO@EXCP	7	144	63	1:00:00.010	1.35
FPEPLAN	DGO@WR2C	1	255	20	1:00:00.000	0.05
FPEPLAN	DGO@EXCP	6	119	63	1:00:00.055	6.89
FPEPLAN	DGO@EXCP	2	134	63	1:00:00.002	0.28
FPEPLAN	DGO@EXCP	8	149	21	1:00:00.002	0.25

Figure 27: The GUI Client Activity Summaries (Operational) panel (1)

<sup>1</sup> In fact you will see alerts in the ISPF UI where those alerts are also defined as exceptions; the fields where alert thresholds have been exceeded are highlighted in red, whereas those fields where only exception thresholds have been exceeded are highlighted in yellow.



This displays the SQL statement type and statistics, but not the SQL statement text. The reason for this is one of network performance. It is not desirable to have thousands of statements coming down to the server every time the operational summaries are displayed, especially as each statement can be up to 2MB in length.

However, you can get the individual SQL text by clicking on the statement line and requesting that the Details pane be displayed by clicking on the Details button in the Activity Browser's menu bar. Click on the "Text" tab in the "Details" pane.

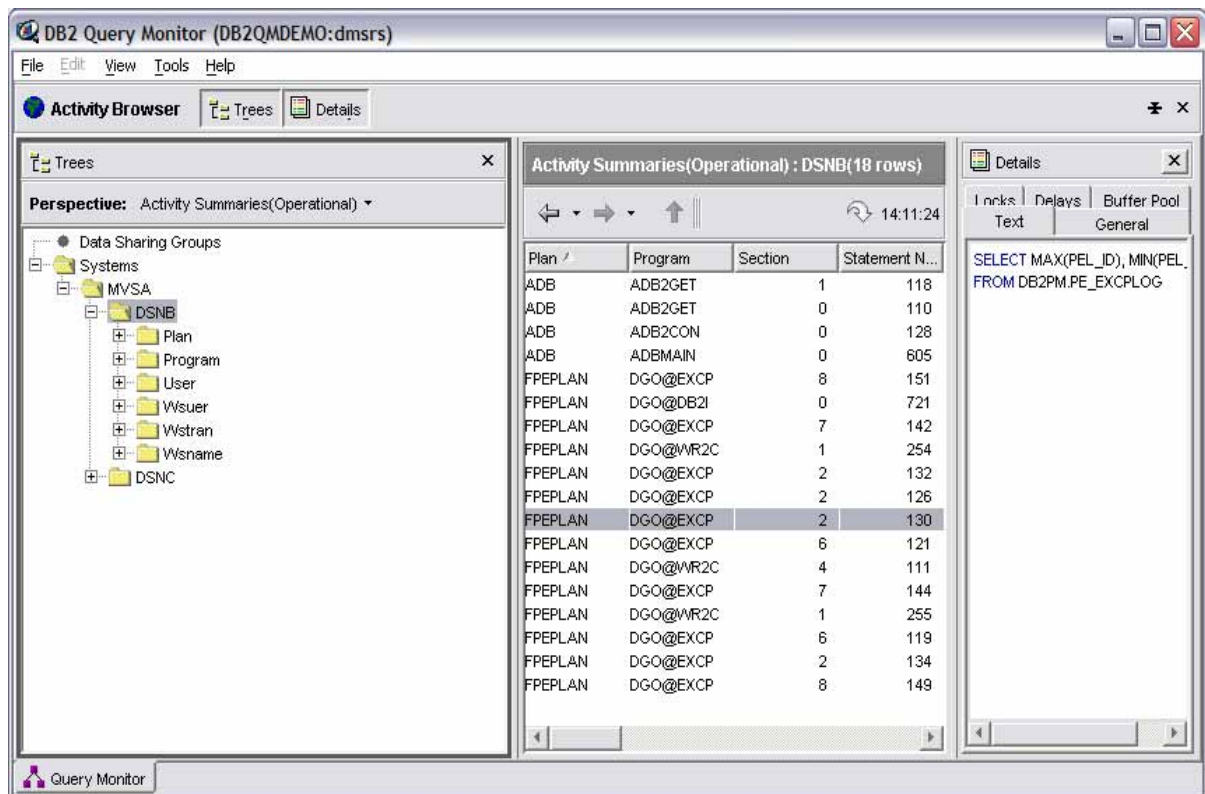


Figure 28: The GUI Client Activity Summaries (Operational) panel (2)

Returning to the Activity Summaries(Operational) pane; sort by average elapsed time (descending) by clicking on the column header until a down arrow is displayed, so that you can see the number of SQL calls and the average elapsed time per call by program. From this you can work out what your elapsed time threshold should be in your monitoring profile for each Plan or program.

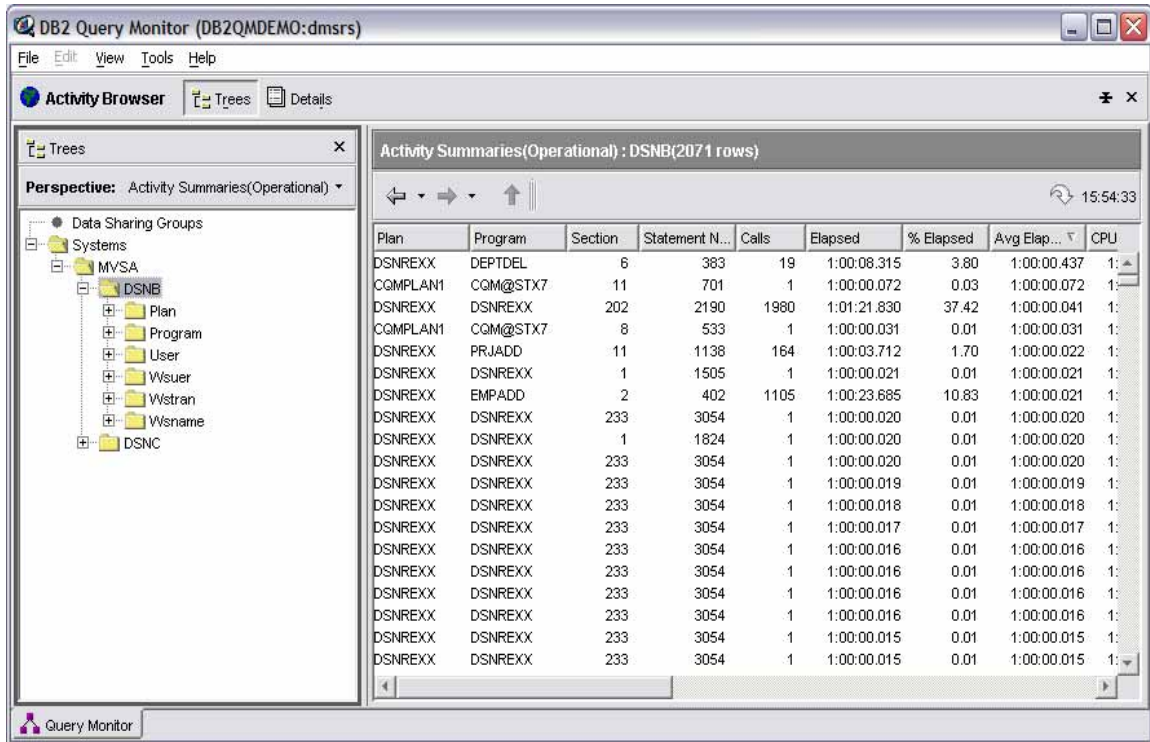


Figure 29: The GUI Client Activity Summaries (Operational) panel (3)

You can repeat the exercise for CPU times. Probably an easier way to achieve this is to expand the tree and highlight the Programs folder. This displays a list of programs that you can sort by average CPU time (per SQL call) descending.

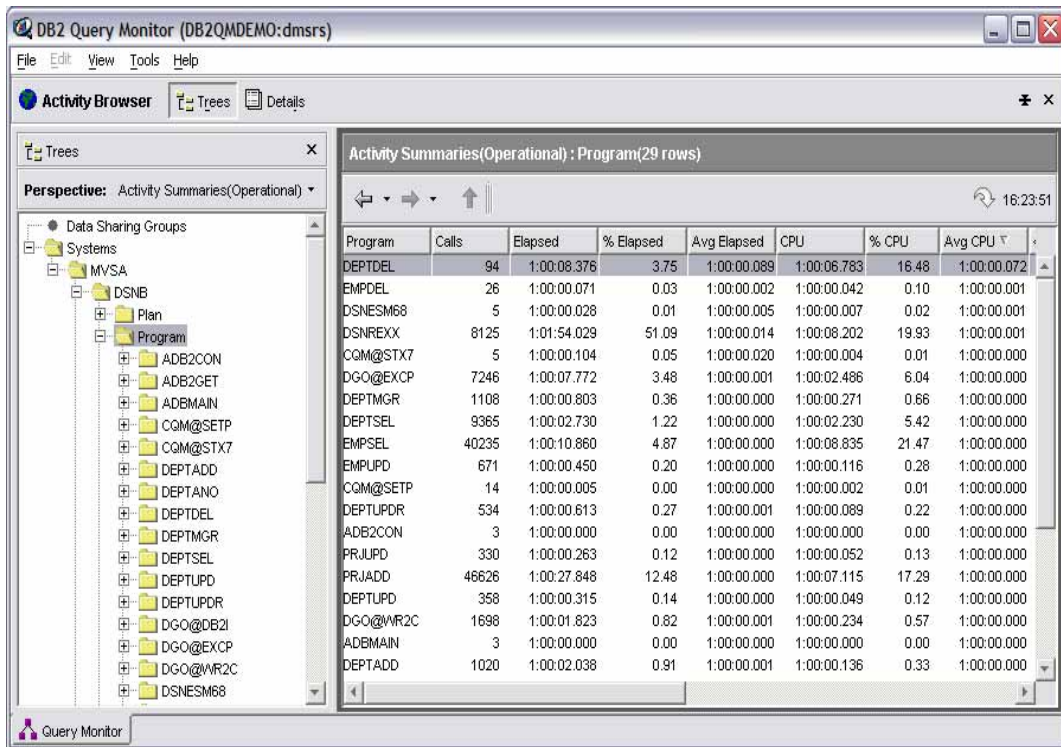


Figure 30: The GUI Client Activity Summaries (Operational) panel (4)

If you let Query Monitor run for a day and then obtain some SQL performance metrics using the method described above, you can then use this information to set exception thresholds, for all of the programs if desired. You can base the exception threshold specification upon the average CPU or elapsed time, plus (say) 50% of the average. This is a good starting point, but you can expect to have to modify the exception thresholds over time. The important point here is not to set the exceptions too low otherwise you will experience many exceptions in a short time, making it impossible to focus on the problem areas. To set alert thresholds, you may wish to use a different rule of thumb - for example, double the average elapsed or CPU time. Again, expect to modify these thresholds as you gain experience.

When setting exceptions for GETPAGES, you need to divide the total number of GETPAGES for each program by the number of calls to derive the average number of GETPAGES. The calculation is not provided on the panel. Use a rule-of-thumb to set the GETPAGE exception threshold (for example, as with CPU and elapsed times, the average plus 50%).

## **Setting Exceptions and Alerts**

To set up exception and alerts refer to the chapter on Data Collection.

Exception and Alert profiles can be created and modified in either the GUI Client or the ISPF UI. Profile created in the ISPF UI can be modified in the GUI Client, and vice versa. However, all profile data is stored in the CQMPROFS VSAM data set(s) on z/OS.

The benefit of using the GUI Client to modify profiles is that you can export the profile to some or all of the QM Data Collectors attached to the CAE Agents connected to the CAE Server. The ISPF UI can only attach to one QM Data Collector at a time.

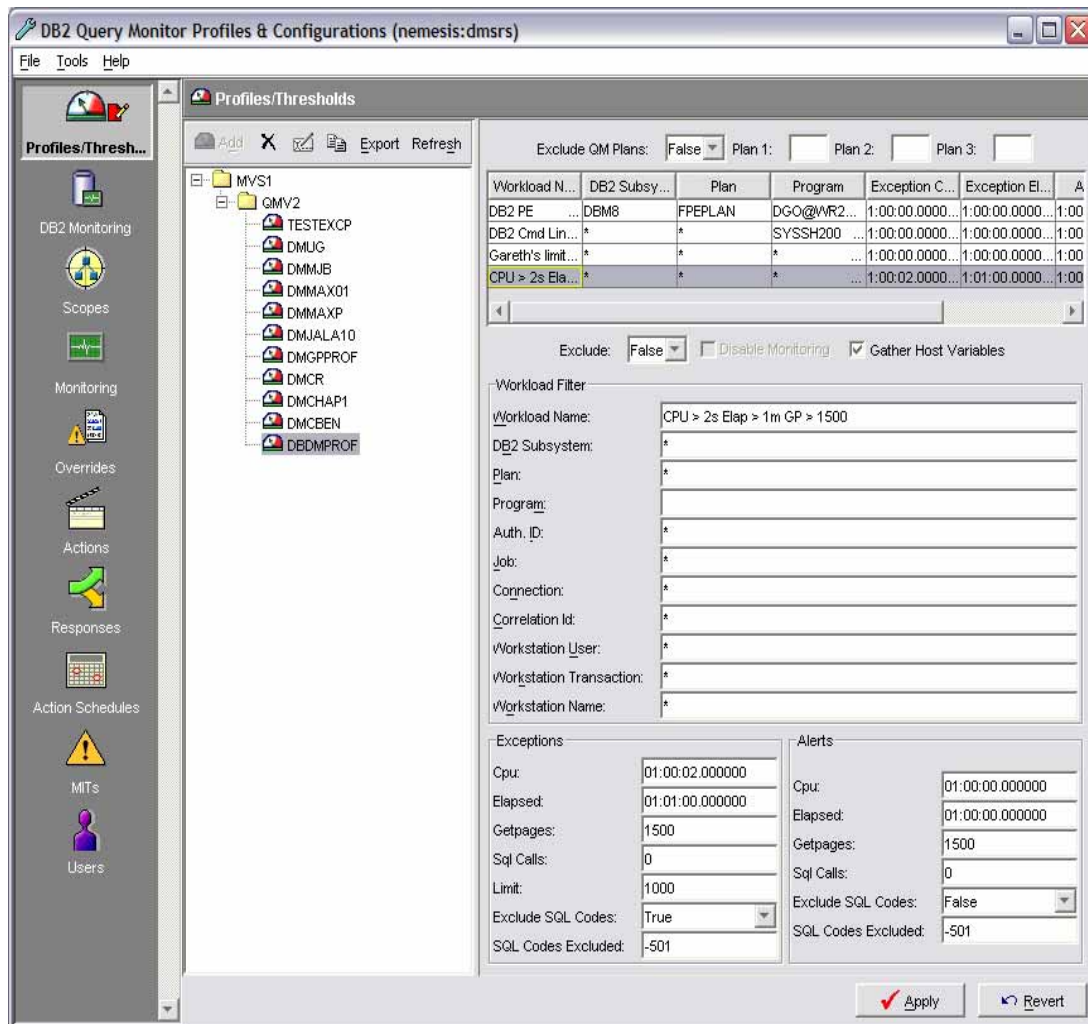


Figure 31: The GUI Client Profiles & Configuration panel (1)

When you want to export a profile, you can choose at what level to do this. There are 4 possibilities:

1. At an individual profile level
2. At QM Data Collector level(s)
3. At MVS level(s)
4. At a global QM level; that is, to all QM Data Collectors connected to this CAE Server via a CAE Agent

The higher the level that you choose to export to, with individual profile level being the lowest and global QM level being the highest, the greater the proliferation of the profile. Remember, profiles will either be created or replaced by the export.

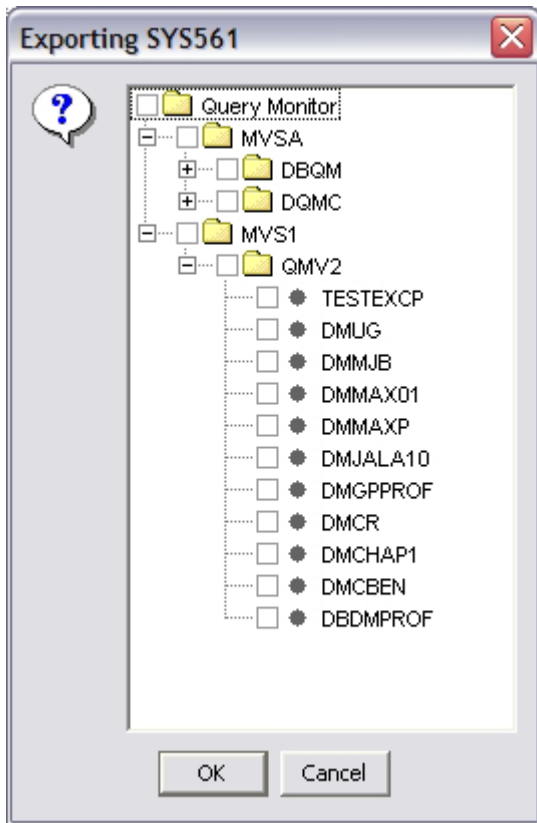


Figure 32: Exporting Monitoring Profiles

Here you can see that the Query Monitor landscape consists of 2 MVS systems (MVSA and MVS1). MVSA has 2 QM Data Collectors (DBQM and DQMC) and MVS1 has 1 (QMV2).

Not shown: Attached to the MVSA and MVS1 QM Data Collectors are 2 DB2 subsystems (one each). Attached to the MVS1 QM Data Collector are 2 DB2 subsystems (DBDM and DBM8).

The SYS561 profile (see the title bar of the pop-up window) can be exported to any parts of the landscape.

## DB2 Monitoring

You can configure DB2 Monitoring from the "DB2 monitoring configuration editor" window (selected from the menu-bar on the left-hand side of the window). This window lets you start/stop monitoring for a specific DB2 attached to the QM Data Collector, as well as being able to switch and refresh the profile.

When you highlight a DB2 subsystem by clicking on it in the left hand pane, the right hand pane displays the details for the profile used to monitor that DB2 subsystem. This is the only place where you see the name of the QM Data Collector that this profile is attached to (assuming that the DB2 is being monitored). If the DB2 subsystem is not being monitored, then you can choose the QM Data Collector to attach to, as well as the monitoring profile to use. Remember that the monitoring profile must be known to the QM Data Collector, that is, it must be stored in the Data Collector's COMPROFS data set. If you try to change the QM Data Collector used to monitor a given DB2 subsystem, then the change is likely to fail with RC128.

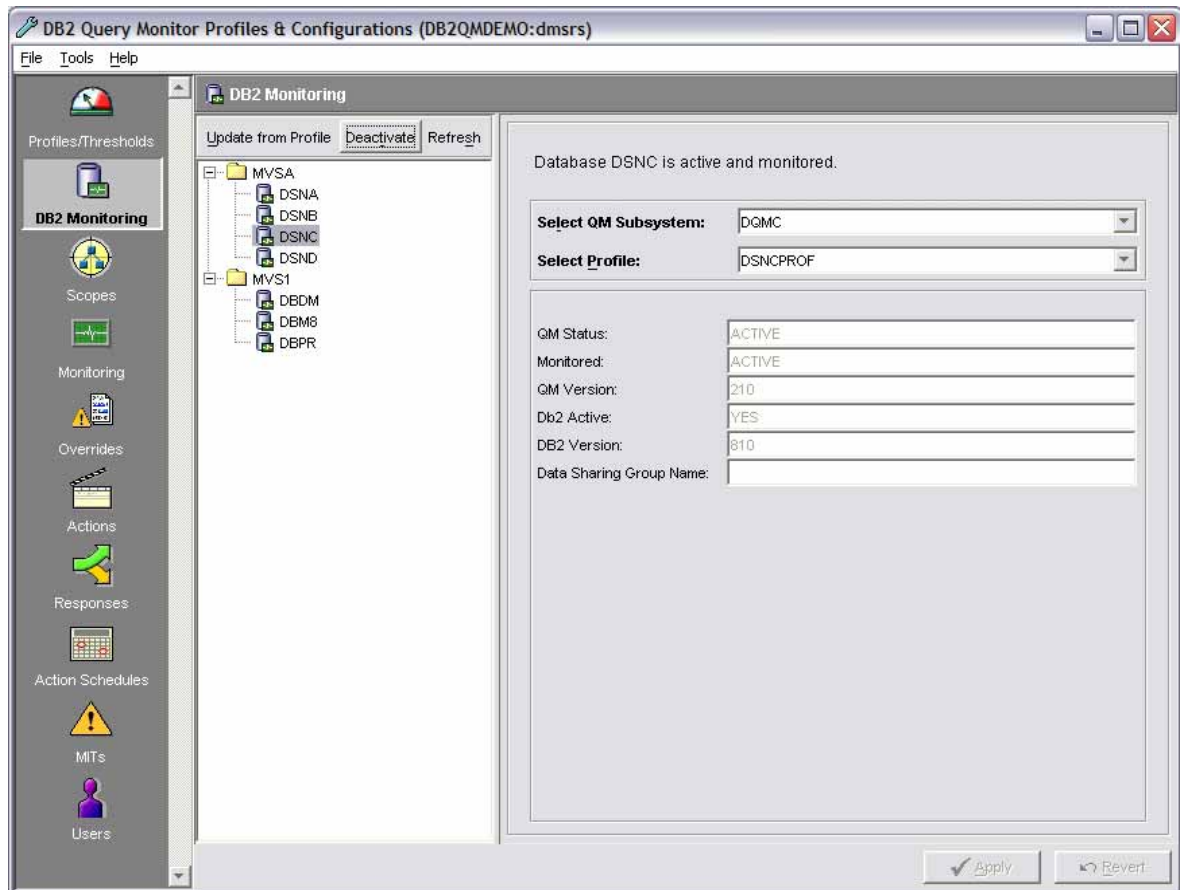


Figure 33: The GUI Client Profiles & Configuration panel (2)

## The Alert Browser

### Adding message boards

For ease of use when viewing alerts, you can define additional message boards in the alert browser so that each message board contains information about a subset of the alert types.

In this example an additional message board that only contains messages pertaining to the CAE Agent has been added.

Click on View>Message Boards>New Message Board and give the message board a meaningful name. In this example it is called "Agent Connections". The next task is to use the Filter option (View>Filters), and in the Event Type pane to just include information about Agent connections.

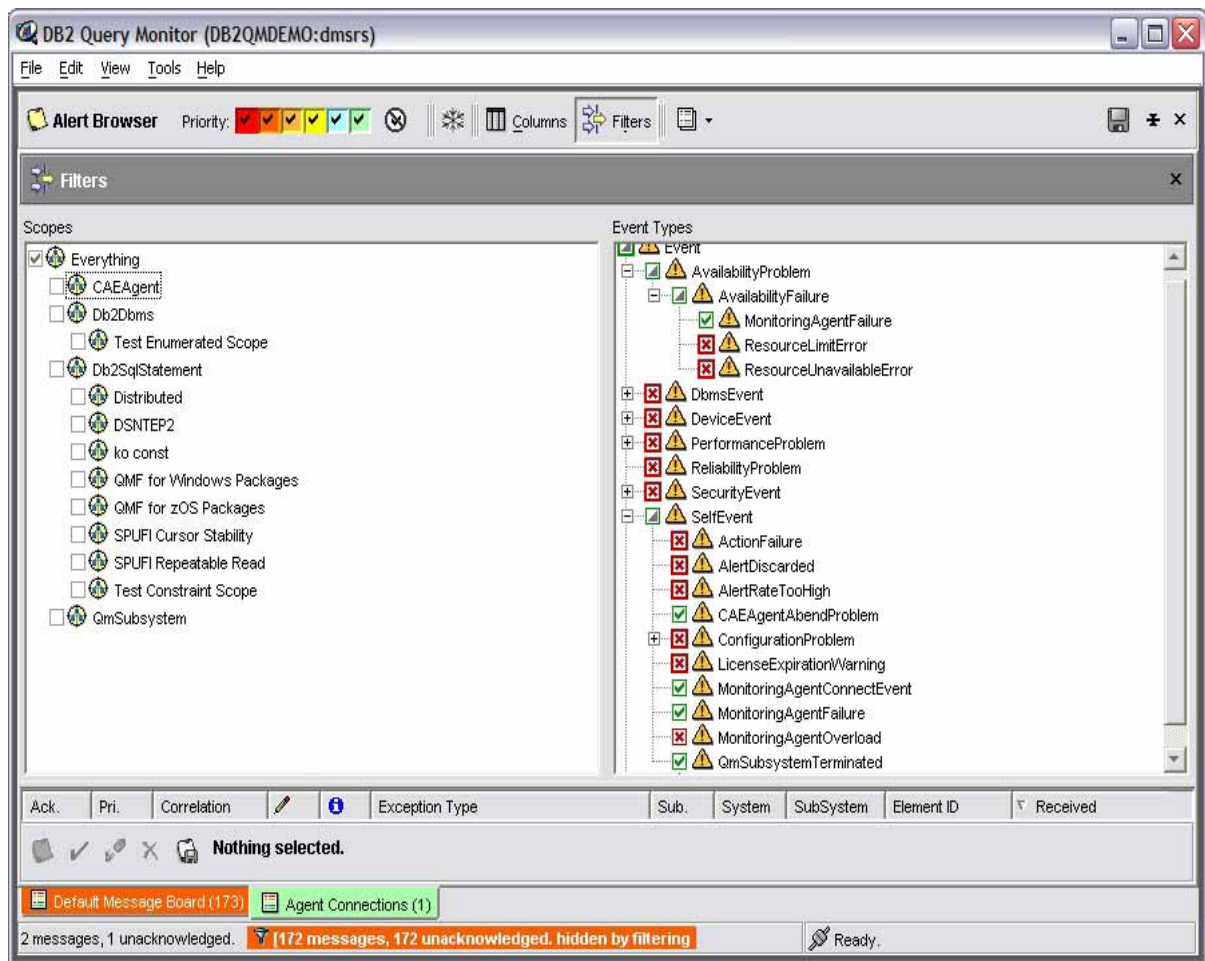


Figure 34: Configuring Message Boards

After completion of this task, only messages pertaining to Agent connections will be shown in the new Message Board. However, since the "Default" Message Board has not been modified, it will also contain the messages about the Agent connections.

In this example the user checked the following in the Event Types pane on the right hand side of the window:

- ✓ MonitoringAgentFailure
- ✓ CAEAgentAbendProblem
- ✓ MonitoringAgentConnectEvent
- ✓ MonitoringAgentFailure
- ✓ QmSubsystemTerminated

New Agent related messages will appear in both the "Default" and "Agent Connections" Message Boards. Any action (such as alert acknowledgement) on these messages will apply to both message boards no matter which message board the action was performed on.

## Alerts

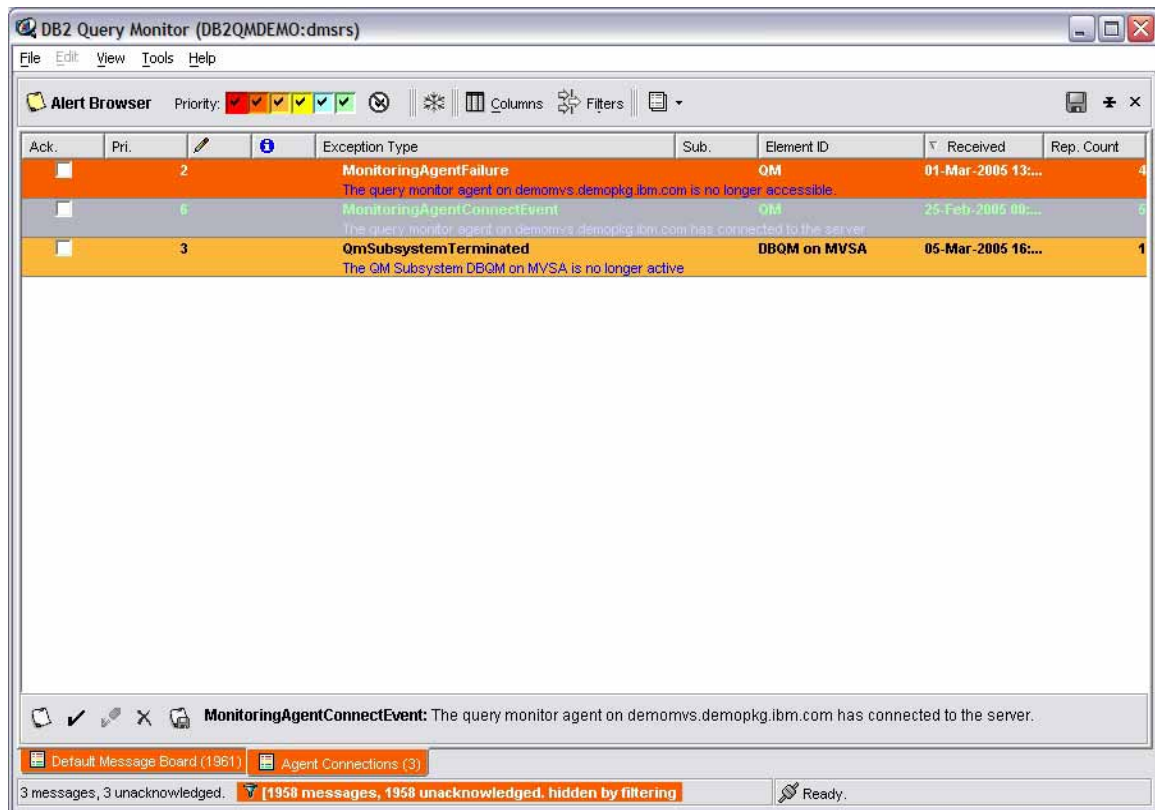


Figure 35: Viewing alerts on the Message Board

When Alerts arrives at the CAE server they will appear in one or more message boards. Each alert will have a priority and an appropriate code assigned to it.

The column name "Rep.Count" indicates how many times this type of alert has occurred. By viewing the Message Details (click on the alert with the right hand mouse button and choose Message Details) and then clicking on the Attributes tab, you can view Date, Time and related information about all the occurrences.



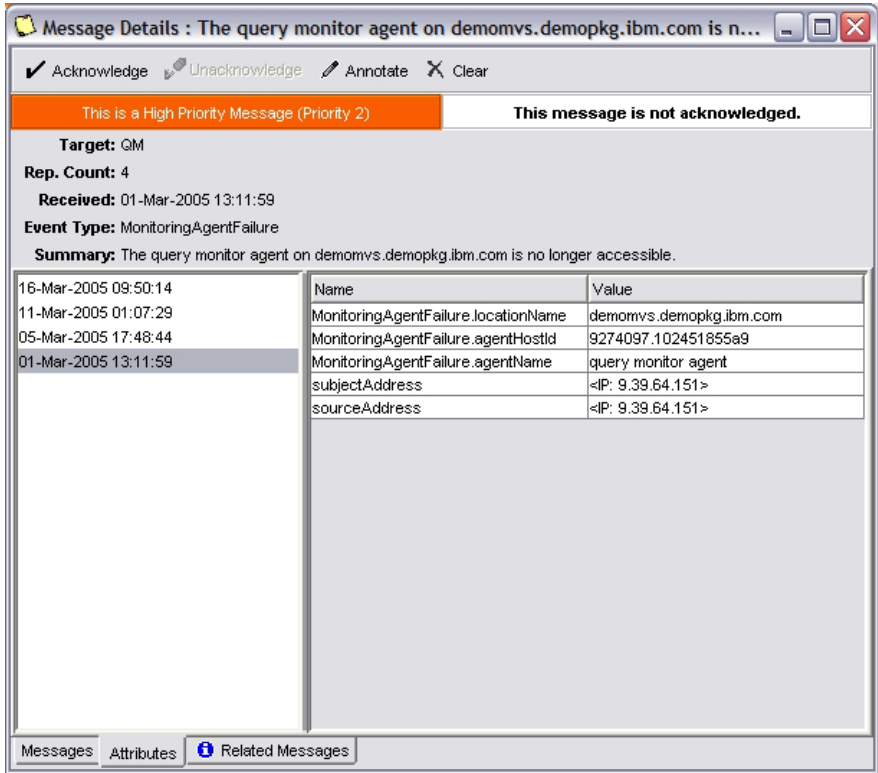


Figure 36: The Message Details window

Another feature of the CAE server is to help determine the root cause of a problem. It can do this when there are multiple exceptions or alerts triggered for a single SQL event. When root cause has been determined then a "+" sign appears next to the priority. Clicking the "+" sign will expand the message board for that event to show the caused events for that SQL statement.

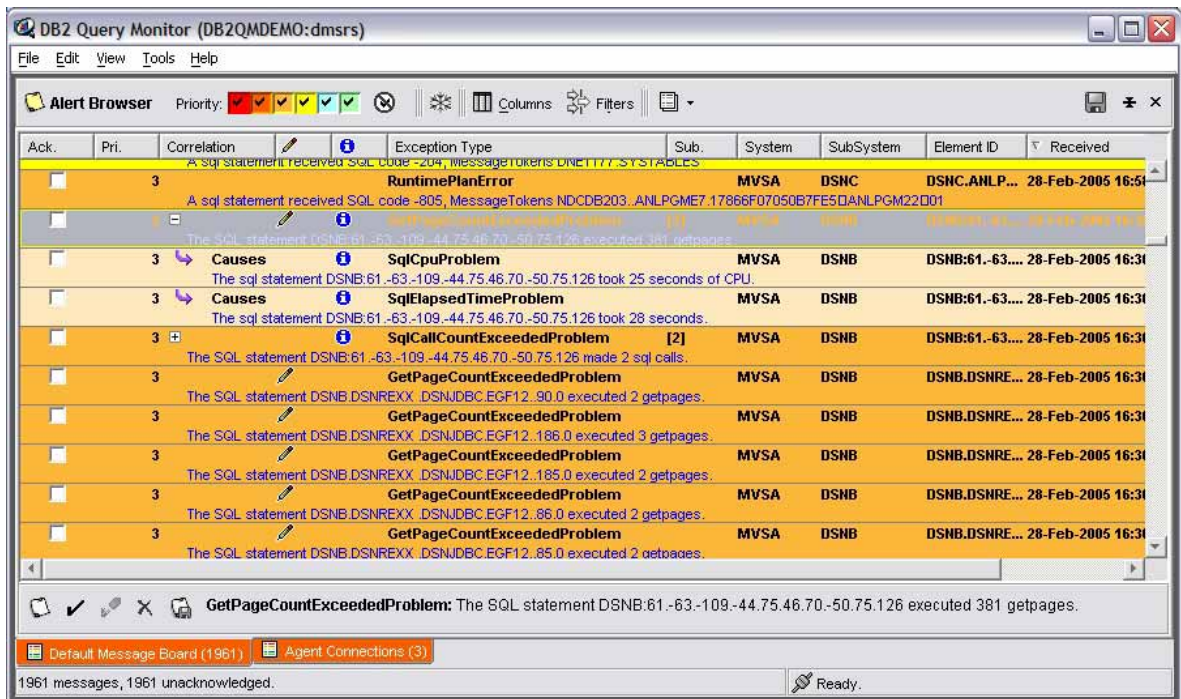


Figure 37: Root cause analysis details on the Alert Browser

Opening the message details pane for the event and clicking on the Related Messages tab will show the “downstream” (events preceding the root cause) and “upstream” events (events following the root cause).

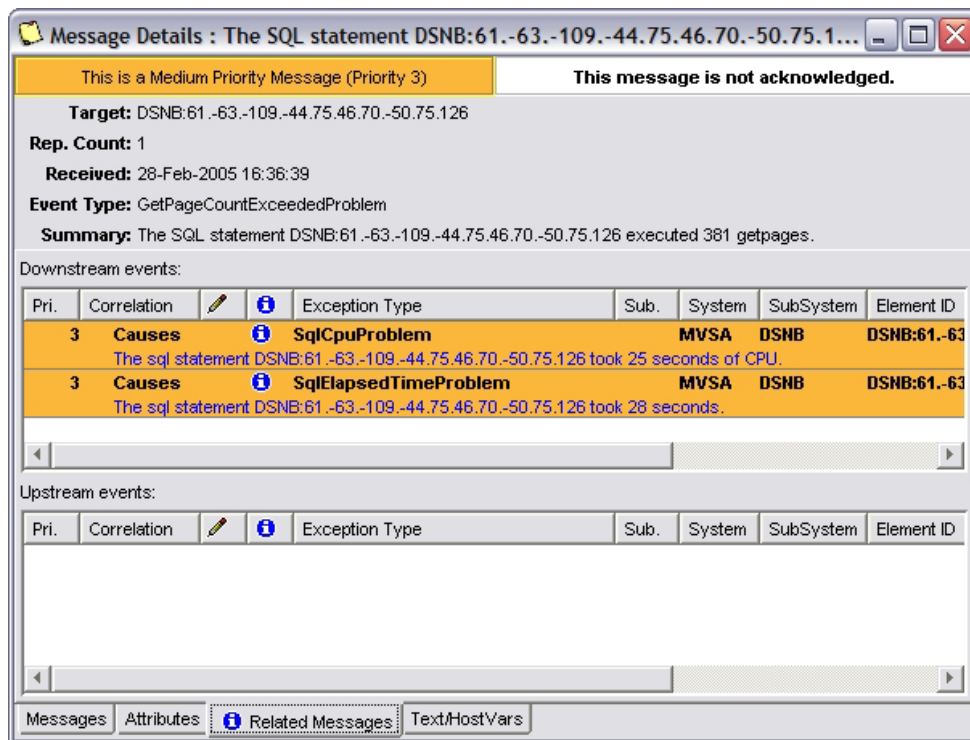


Figure 38: Root cause analysis in the Message Details window

In this case we can see that there are 2 downstream events that were caused by the main event:

**SQLElapsedTimeProblem** and **SQLCpuProblem** were caused by the **GetPageCountExceededProblem**.

You can examine the SQL text for the statement that triggered the alert by clicking on the Text/Host Vars tab.

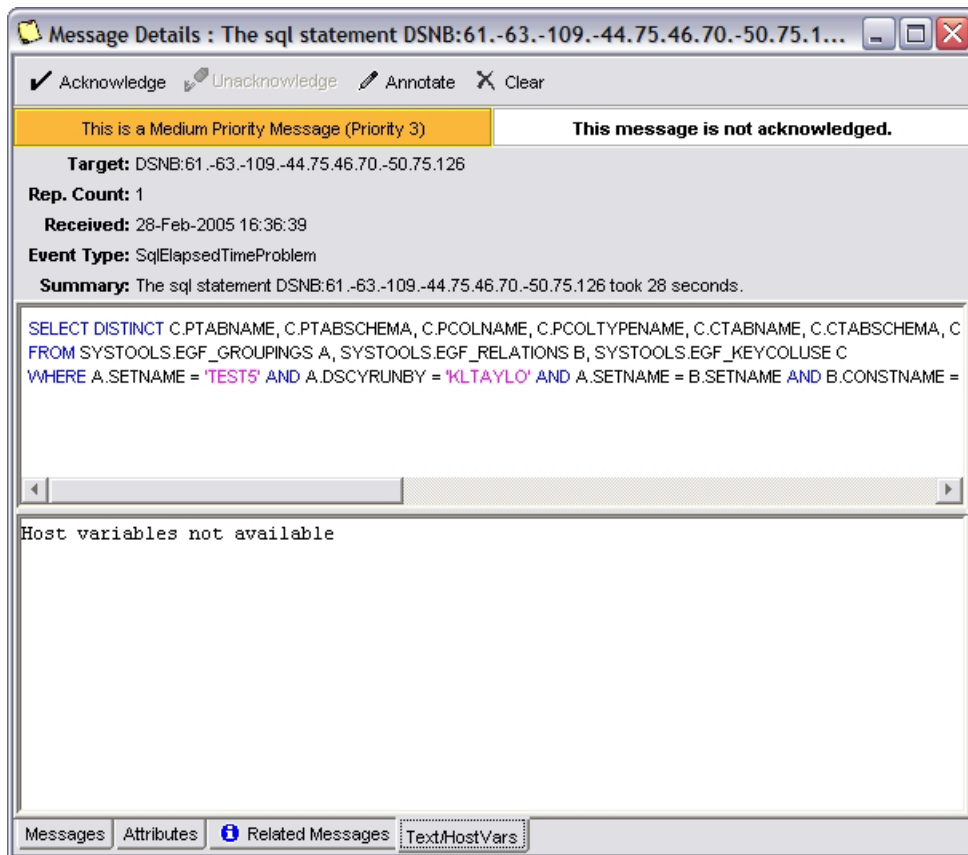


Figure 39: SQL Statement text in the Message Details window

# Chapter 6: The DB2 Query Monitor Performance Database

## Overview

DB2 Query Monitor provides batch programs which can be used to load the data from the VSAM backstores into DB2 tables. The DB2 tables can then be queried to answer such questions as:

- What are the total and average CPU and elapsed times by SQL statement for a plan, collection or program?
- What are the elapsed time, getpages, synchronous and asynchronous reads by object for an SQL statement?
- What are the total and average CPU and elapsed times for all SQL statements that access a specific table?
- What is the SQL statement text for a dynamic or static SQL call?

To help you maximise the benefits of the performance database, this chapter describes its structure, examining the relationship of the tables to each other. It discusses in some detail the process of defining the performance database, suggesting possible indexes to define on the database tables and outlining which columns should be used as join columns when querying the database. Joining the Query Monitor performance database to DB2 catalog tables is discussed, as this is needed to be able to obtain information about the SQL statement text for static SQL statements.

As Query Monitor stores SQL statement text in Unicode, a User-Defined Function (UDF) is required for sites using DB2 Version 7. This chapter discusses the implementation and use of a sample UDF available from the Rocket Software FTP site.

We then move on to discuss some operational issues concerning the loading of the database, and finish with some sample queries against the performance database which you can adapt for your own purposes.

## Implementation

The Query Monitor Performance Database was introduced with APAR PQ90774 and PTF UQ93292. The implementation process is documented in Appendix C of the *IBM DB2 Query Monitor for z/OS User's Guide*, SC18-9202, which should be used as the primary guide for implementing the Query Monitor performance database. The following sections contain some additional guidance:

- Database Tables - table structure information
- Load Process - operational considerations
- Queries - some samples to get you started

## Database Tables

### Table structure

The performance database comes with a predefined set of tables. The table names and their associated function are described in the table below. Each table is a repository for the information stored in the VSAM backstores. There are also three indexes used to support the LOB tables. The following table lists all the tables and provides a brief description (the description also lists the Query Monitor backstore files which contains the data to be loaded into the database).

Table name	Description
CQM21_SUMM_METRICS	METRDATA - summary level information related to SQL call execution
CQM21_SUMM_OBJECTS	OBSDATA - summary object level data
CQM21_SUMM_TEXT	TEXTDATA - summary level SQL text data
CQM21_INTERVALS	INTERVALS - QM interval information
CQM21_STMT_TYPES	SQL statement type and description
CQM21_EXCEPTIONS	EXCPINDX and EXCPDATA - exception data such as SQL calls, text, SQLCA, and host variables
CQM21_EXCP_CALLS	EXCPINDX and EXCPDATA - exception data SQL calls information
CQM21_EXCP_HOSTV	EXCPINDX and EXCPDATA - exception host variables information
CQM21_EXCP_OBJES	EXCPINDX and EXCPDATA - exception objects information
CQM21_EXCP_TEXT	EXCPINDX and EXCPDATA - exception text information
CQM21_DB2_COMMANDS	DB2CMDS - DB2 Commands
CQM21_SQLCODES	SQLCDATA - negative SQLCODE information
CQM21_SQLCODE_DET	SQLCDATA - details about negative SQLCODE
CQM21_SQLCODE_TEXT	SQLCDATA - text for negative SQLCODEs

Figure 40: Query Monitor Performance Database tables

NOTE: You can change the names of the tables and the creator; however, if you do so, you must remember to make corresponding changes to the programs that load the data. This is discussed later on in this chapter.

The next three diagrams illustrate the data structures for these objects and the columns that you should use to join the data. Tables which are logically related are presented together, as parent and child tables. Be aware that these relationships are not implemented using DB2-enforced referential integrity. Provided for each table is a list of suggested primary key columns. Those tables with an asterisk (\*) next to the name in the title bar have a non-unique key.

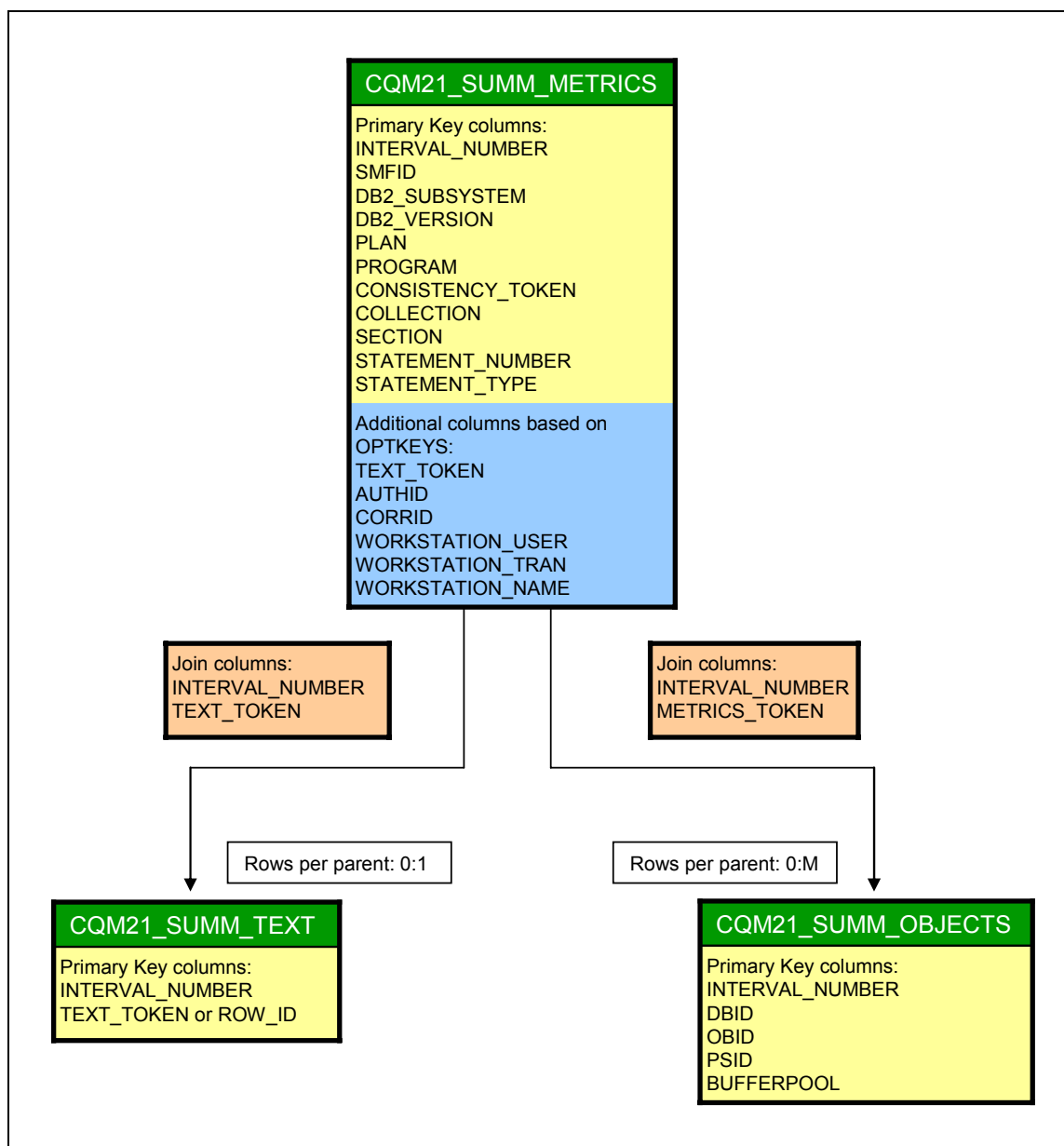


Figure 41: The summary data tables

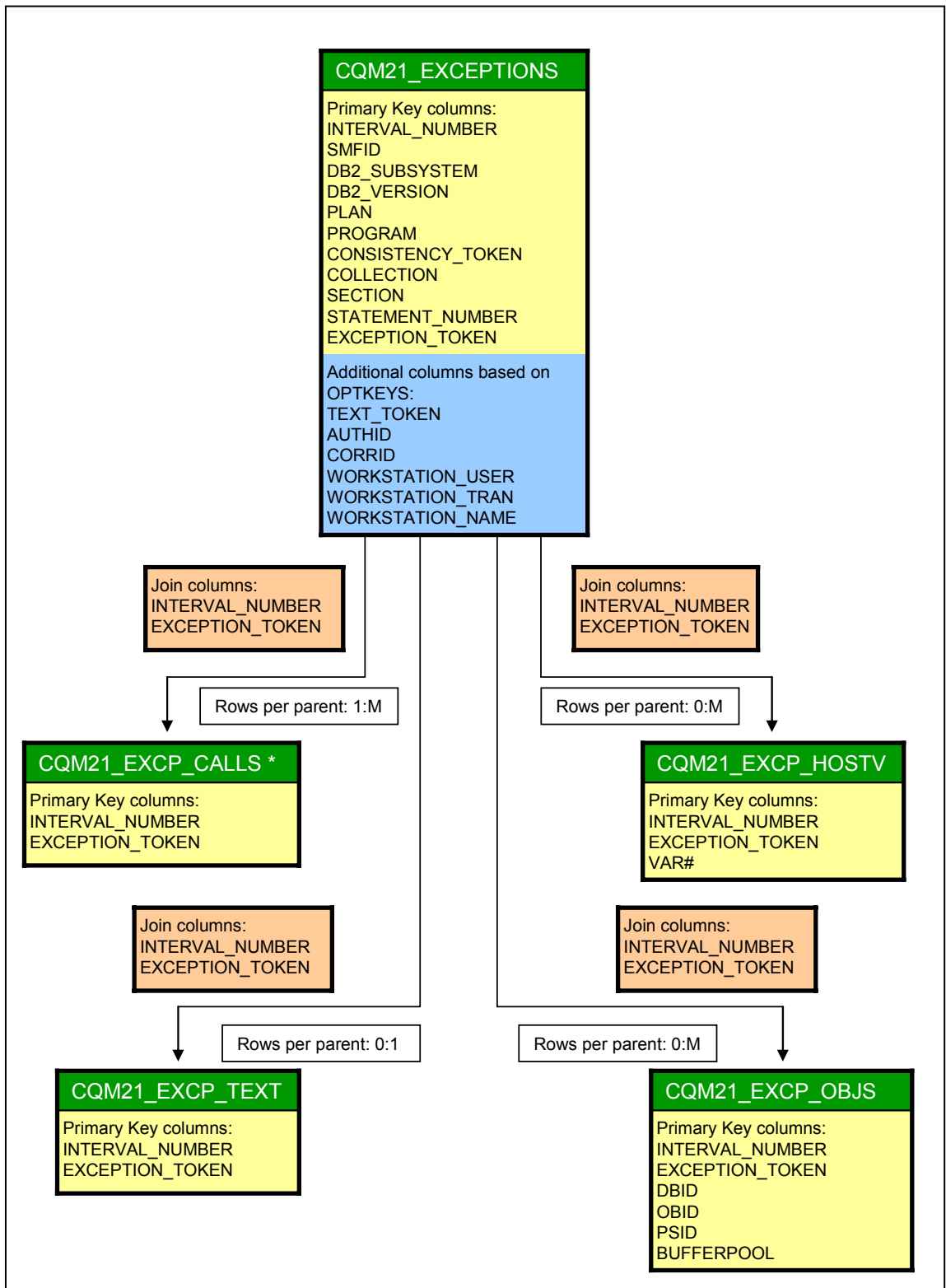


Figure 42: The exception tables

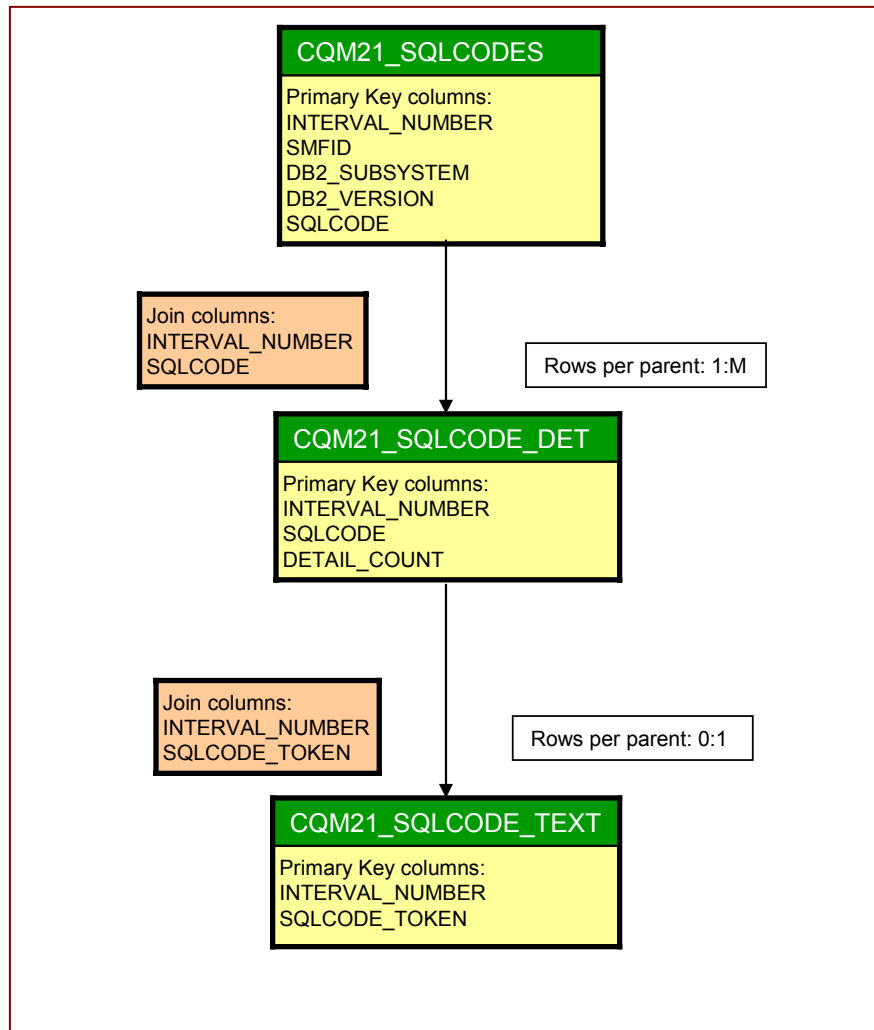


Figure 43: The SQLCODE tables

## Defining the performance database

You need to define the database to store all of the objects in the performance database. The sample job located in &prefix.SCQMSAMP(CQMCRDB) creates the database as SYSTOOLS and uses SYSTOOLS as the creator. If you are creating only one QM data collector, then you can use these defaults.

If you plan on having more than one QM data collector feeding data into the QM Performance database, then we recommend that you define a separate database and set of tables for each collector. This is due to the fact that some of the tables, including the CQM\_INTERVALS table, lack the column for the QM data collector subsystem name and so uniqueness cannot be guaranteed. The default name for the database and table creator is SYSTOOLS and so can be changed with a single change command. We also recommend that the database and table creator name reflect the QM Data Collector sub-system name such as CQMqmssid. This way you can segregate the data from a particular QM data collector in a single repository.



Our QM data collector name is DBQM. An example of the CREATE DATABASE DDL follows where we changed SYSTOOLS to CQMDBQM. We included CQM as a high-level qualifier in case we use a catalog navigation tool to view all databases beginning with CQM, as it facilitates the catalog search for all of the QM performance databases. We also used CQMDBQM as the creator name when creating the tables.

```
CREATE DATABASE CQMDBQM
  BUFFERPOOL BP0
  STOGROUP     SYSDEFLT
  CCSID        EBCDIC;

COMMIT;
```

## Creating the performance database objects

Next you need to create the performance database objects. Edit member CQMDDL7 or CQLDDL8 in the library &prefix.SCQMSAMP depending on whether you are running DB2 V7 or DB2 V8 respectively. If you have decided to change the database name as discussed above then edit the member and make the following changes:

- Change database SYSTOOLS to the database name used in CQMCRDB above

Change the table creator SYSTOOLS to the same name as the database name, or any other name that conforms to your naming standards. This name must be used in the load program CQM@LDB2.

## Table Creator

It is advised that you use at least one set of tables per QM data collector as there is no CQM\_SUBSYSTEM column in the CQM21\_INTERVALS table.

## Primary Keys

In figures 1 to 3 above, suggested primary keys are provided, plus optional additional keys you can add to the primary index if you request finer levels of data summarization through the use of the OPTKEYS setting in CQMPARMS. Note that the suggested primary key for CQM21\_EXCP\_CALLS is not unique. As well as these primary keys, you may choose to add additional indexes to support joins, grouping and ordering when querying the performance database. Because values for DBNAME, OBJECT\_CREATOR, OBJECT\_NAME and PAGESET\_NAME in CQM21\_SUMM\_OBJS and CQM21\_EXCP\_OBJS are not guaranteed to be collected, these columns may contain the value 'N/A' and are therefore not suitable for use as a primary index.

## Additional Tables

There are two additional tables, CQM21\_INTERVALS and CQM21\_STMT\_TABLES, not presented in the structure diagrams (as

COM21\_INTERVALS is not related to any of the other tables, and COM21\_STMT\_TYPES is just a fact table):

- COM21\_INTERVALS, which contains one row per interval, and indicates which OPTKEYS settings were in force during that interval. The suggested primary key is INTERVAL\_NUMBER.
- COM21\_STMT\_TYPES, which allows a numeric code to be used for each statement type. The suggested primary key is STATEMENT\_TYPE.

### Join Columns

In figures 1 to 3 above, suggested columns for joining the logical child tables to the logical parent tables are provided. Additionally, you can join to COM21\_INTERVALS and COM21\_STMT\_TYPES using INTERVAL\_NUMBER and STATEMENT\_TYPE respectively.

### Text Columns

The SQL Statement text for static SQL is kept in the DB2 catalog, and not in the QM performance database tables. To obtain the SQL statement text, you may want to join to the SYSIBM.SYSSTMT and SYSIBM.SYSPACKSTMT catalog tables. Be aware that, if plans or packages have been rebound since the data was collected, the SQL statement text may not be available.

### Sample User Defined Function for DB2 Version 7

Note: This section applies only to installations using versions of DB2 prior to Version 8.

As discussed in the Appendix C of the User Guide, the SQL text is stored in Unicode in BLOB columns in DB2 V7 whereas in DB2 V8 the text is stored in CLOB columns. This is so that in both versions the text tables can be joined to other Query Monitor or DB2 catalog tables. In order to view the SQL text when running a Query against a DB2 V7 QM Performance Database, a sample User Defined Function has been written to convert the SQL text from Unicode to the local code page. This UDF is currently unsupported and is distributed on an as-is basis.

### Installing the UDF.

The UDF (CQMUDF01) is not shipped with DB2 QM V2.1. If you wish to use this piece of code, you will need to FTP the executable from the Rocket FTP site. Below is a sample job to achieve this.

Update the dataset name shown in red to suit your installation standards before submitting the job.

```

// *
//FTPGET EXEC PGM=FTP
//SYSPRINT DD SYSOUT=*
//OUTPUT DD SYSOUT=*
//INPUT DD *
ftp.rocketsoftware.com
downloadl
rocket
cd /fromRocket/CQM_UDF
locsite rec=fb lr=80 blk=3120
locsite unit=dasd cy pri=1 sec=1
binary
get LOADLIB 'CQM.UDF.LOADLIB.XMIT'
get
quit

```

**Figure 44: The FTP command stream to download the V7 UDF**

After the job has run, this dataset will contain a XMIT file. You should now run the sample JCL shown below, which will issue a RECEIVE command to place module CQMUDF01 into dataset CQM.UDF.LOADLIBX. (Tailor to site standards and run).

```

//RECV EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=A
//SYSTSIN DD *
RECEIVE INDSN('CQM.UDF.LOADLIB.XMIT')
DSN('CQM.UDF.LOADLIBX')

```

**Figure 45: Extracting the UDF load library**

Note - If you wish to ship this function to other LPARs, via the PC, make sure that you FTP the LOADLIB.XMIT dataset to the PC in binary format, as this will ensure the integrity of the file when it arrives at the other LPAR.

## Defining the UDF to DB2.

The DDL to define the UDF to DB2 is shown below.

```

--
-- Licensed Materials - Property of IBM
-- 5697-I03
-- (c) Copyright IBM Corp. 1999, 2004 All Rights Reserved.
-- (c) Copyright Rocket Software, Inc. 1999, 2004 All Rights Reserved.
-- US Government Users Restricted Rights - Use, duplication, or
-- disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
--
--
-- Member: CQMCRUDF
--
-- This member creates the UDF objects for
-- IBM DB2 Query Monitor for Z/OS 2.1.0
--
-- Instructions:
-- 1) If recreation of the CQM21_GET_TEXT function is desired,
--    then un-comment the DROP FUNCTION statement
-- 2) Change WLMX to the WLM environment name that the CQM21_GET_TEXT
--    function will execute in
-- 3) NOTE: Do not change the value of the EXTERNAL NAME parameter.
--    Be aware that this value is case sensitive and must be
--    in all caps.
--
-- DROP FUNCTION CQMTOOLS.CQM21_GET_TEXT;
--
--
CREATE FUNCTION CQMTOOLS.CQM21_GET_TEXT(BLOB, INTEGER)
  RETURNS CLOB(2M)
  FENCED
  DETERMINISTIC
  STAY RESIDENT YES
  NO SQL
  NO EXTERNAL ACTION
  LANGUAGE ASSEMBLE
  WLM ENVIRONMENT WLMX
  PARAMETER STYLE DB2SQL
  EXTERNAL NAME 'CQMUDF01';

GRANT EXECUTE ON FUNCTION
  CQMTOOLS.CQM21_GET_TEXT(BLOB, INTEGER) TO PUBLIC;

```

**Figure 46: DDL to create the V7 UDF**

Change WLMX to a valid WLM Application Environment before running this DDL.

### **Making the UDF available to DB2.**

The final step for installing the UDF is to add the module (CQMUDF01) to a load library on the STEPLIB of the started task that supports the WLM Application Environment chosen, or if using a new load library to add it to this STEPLIB.

If you chose to add a new library to the WLM Application Environment started task, you will need to refresh WLM using a command similar to that below:-

```
F WLM, APPLENV=WLMX, RESUME
```

## Using the CQMUDF01 function in an SQL statement

The UDF function takes two parameters, the first is the SQLTEXT column name and the second is an integer representing the desired output CCSID.

Assuming you have defined the UDF with the name of GET\_TEXT and a schema name of SYSTOOLS, the following example shows the conversion of the SQLTEXT column to CCSID 37:

```
SELECT CQMTOOLS.CQM21_GET_TEXT(SQLTEXT,37) FROM  
SYSTOOLS.CQM21_EXCP_TEXT;
```

## Error conditions reported by the CQMUDF01 User Defined Function

If the CQMUDF01 UDF encounters an error invoking conversion services, the SQL statement that invoked the UDF will terminate with an SQLCODE of -443 and an SQLSTATE of 38yzz where y is the hexadecimal return code from conversion services and zz is the hexadecimal reason code. Please refer to "z/OS Support for UNICODE Using Conversion Services" (SA22-7649) for more information about any return code you receive from conversion services.

## Load Process - Operational Considerations

### Scheduling the Load Program CQM@LDB2

The load process, performed by the batch program CQM@LDB2, has the following input parameters for controlling the interval(s) to be loaded:

- STARTING\_INTERVAL
- ENDING\_INTERVAL
- INTERVAL\_UNITS

The INTERVAL\_UNITS parameter controls how the other two parameters are interpreted. The setting of this parameter allows for two basic options for running the load program:

1. As soon as an interval switch occurs - use INTERVALS
2. As part of the overnight batch suite - use DAYS

The actual frequency also depends on the interval selected in the CQMPARMS file which is typically either 1 hour, 24 hours or some subdivision. The first option enables the data to be loaded as soon as it becomes available if the interval is less than 24 hours. The starting and ending interval parameters can be left out so that the default of -1 is used. The load job could be triggered by the message written to the log to record the end of interval processing:

```
CQM2401I INTERVAL PROCESSING ENDED FOR qmsubsysid
```

The second option would be to schedule the job to run as part of the overnight batch suite. If the job was run every day then, as above, the starting and ending interval can be omitted so that the default of -1, in other words the previous day, is used.

### Load Program CQM@LDB2 Execution considerations

The load program has three steps. The first step extracts the data from the VSAM backstore datasets for all tables except the three text tables and creates the LOAD utility files and control card dataset. The second step extracts the text data and uses INSERT processing to load this data as the text is stored in LOB columns. The third step executes the DB2 LOAD utility to load the remaining tables.

#### Step 1 CQM@WDB2

The space parameters for the LOAD files created by CQM@WDB2 are:

```
SPACE=(CYL,(1,1),RLSE)
```

We suggest that a global change is made to:

```
SPACE=(CYL,(10,10),RLSE)
```

to avoid running out of space. We found the default of 1,1 too small, but that 10,10 was large enough to make a more accurate assessment of the disk space requirements. Once experience is gained then the space parameters can be adjusted to accommodate the largest amount of data unloaded in an execution.

A row is INSERTed into the CQM21\_INTERVALS table for each interval selected in the input parameters. Note that these rows are inserted irrespective of the outcome of the whole process and so the table will contain duplicate rows if the load program is rerun for the same interval unless a suitable unique index is created.

#### Step 2 CQM@ITXT

This program loads the text data to the three TEXT tables by using INSERTs.

#### Step 3 DSNUTILB

The JCL for this step does not include either an error or discard file for the LOAD utility. Should a failure occur then these statements can be added to the JCL:

```
//SYSDISC DD DSN=DB2QM.V2R1.SYSDISCD,DISP=(,CATLG),
// UNIT=SYSALLDA,SPACE=(CYL,(10,10),RLSE),
// DCB=(LRECL=32756,BLKSIZE=0,RECFM=VB)
//SYSERR DD DSN=DB2QM.V2R1.SYSERROR,DISP=(,CATLG),
// UNIT=SYSALLDA,SPACE=(CYL,(10,10),RLSE),
// DCB=(LRECL=32756,BLKSIZE=0,RECFM=VB)
```

Figure 47: Error and Discard DD cards for the performance database load process

## Sample Queries

Here are some sample queries you can use against the tables in the QM performance database and the DB2 catalog tables. These include view definitions to turn timestamps into decimal fields, and joins to the DB2 catalog to extract information about SQL text for static SQL queries.

### Selecting summary metrics data with SQL text (dynamic SQL)

```
SELECT A.TEXT_TOKEN, A.AUTHID, A.PROGRAM, A.SECTION,
       A.STATEMENT_NUMBER, C.STATEMENT_DESC, A.CONSTENCY_TOKEN,
       B.SQLTEXT, A.DB2_CPU
       FROM SYSTOOLS.CQM21_SUMM_METRICS A,
            SYSTOOLS.CQM21_SUMM_TEXT B,
            SYSTOOLS.CQM21_STMT_TYPES C
       WHERE A.TEXT_TOKEN = B.TEXT_TOKEN AND
            A.INTERVAL_NUMBER = B.INTERVAL_NUMBER AND
            C.STATEMENT_TYPE = A.STATEMENT_TYPE
```

Figure 48: Selecting summary metrics data with SQL text (dynamic SQL)

### Selecting summary metrics data and associated SQL text from SYSPACKSTMT (static SQL)

```
SELECT A.AUTHID, A.PROGRAM, A.SECTION,
       A.STATEMENT_NUMBER, C.STATEMENT_DESC, A.CONSTENCY_TOKEN,
       B.SEQNO, B.STMT, A.DB2_CPU
       FROM SYSTOOLS.CQM21_SUMM_METRICS A,
            SYSIBM.SYSPACKSTMT B,
            SYSTOOLS.CQM21_STMT_TYPES C
       WHERE A.TEXT_TOKEN = X'00000000000000000000' AND
            A.CONSTENCY_TOKEN = B.CONTOKEN AND
            A.COLLECTION = B.COLLID AND
            A.PROGRAM = B.NAME AND
            A.SECTION = B.SECTNO AND
            A.STATEMENT_NUMBER = B.STMTNOI AND
            C.STATEMENT_TYPE = A.STATEMENT_TYPE
```

Figure 49: Selecting summary metrics data and associated SQL text from SYSPACKSTMT (static SQL)

## Selecting summary metrics data and associated DECLARE CURSOR SQL text from SYSPACKSTMT (static SQL)

```
SELECT A.AUTHID, A.PROGRAM, A.SECTION,
       A.STATEMENT_NUMBER, C.STATEMENT_DESC, A.CONSTENCY_TOKEN,
       B.SEQNO, B.STMT, A.DB2_CPU
       FROM SYSTOOLS.CQM21_SUMM_METRICS A,
            SYSIBM.SYSPACKSTMT B,
            SYSTOOLS.CQM21_STMT_TYPES C
WHERE A.TEXT_TOKEN = X'00000000000000000000' AND
      A.CONSTENCY_TOKEN = B.CONTOKEN AND
      A.COLLECTION = B.COLLID AND
      A.PROGRAM = B.NAME AND
      A.SECTION = B.SECTNO AND
      SUBSTR(B.STMT,9,7) = 'DECLARE' AND
      C.STATEMENT_TYPE = A.STATEMENT_TYPE
```

Figure 50: Selecting summary metrics data and associated DECLARE CURSOR SQL text from SYSPACKSTMT (static SQL)

## Selecting exception metrics and associated SQL text (dynamic SQL).

```
SELECT A.AUTHID, A.PROGRAM, A.SECTION, A.THREAD_TOKEN,
       A.LAST_SQLCODE, A.DB2_CPU, A.DB2_ELPASED,
       B.SQLTEXT
       FROM SYSTOOLS.CQM21_EXCEPTIONS A,
            SYSTOOLS.CQM21_EXCP_TEXT_V2 B
WHERE A.EXCEPTION_TOKEN = B.EXCEPTION_TOKEN
```

Figure 51: Selecting exception metrics and associated SQL text (dynamic SQL)

## Selecting SQLCODE detail data and associated SQL text (dynamic SQL)

```
SELECT A.AUTHID, A.SQLCODE, A.SQLERRP, A.SQLERRM,
       B.SQLTEXT
       FROM SYSTOOLS.CQM21_SQLCODE_DET A,
            SYSTOOLS.CQM21_SQLCODE_TEXT B
WHERE A.SQLCODE_TOKEN = B.SQLCODE_TOKEN
```

Figure 52: Selecting SQLCODE detail data and associated SQL text (dynamic SQL)

## Selecting summary metrics data with associated statement type description.

```
SELECT A.PLAN, A.PROGRAM, A.STATEMENT_TYPE, B.STATEMENT_DESC FROM
       SYSTOOLS.CQM21_SUMM_METRICS A,
       SYSTOOLS.CQM21_STMT_TYPES B
WHERE A.STATEMENT_TYPE = B.STATEMENT_TYPE
```

Figure 53: Selecting summary metrics data with associated statement type description



## Summary metrics for all statements in a collection.

```
SELECT A."PROGRAM"  
      , A.STATEMENT_NUMBER  
      , SUM(A.SQL_CALLS          ) AS SQL_CALLS  
      , SUM(DEC(3600*HOUR(A.DB2_CPU)  
            +60*MINUTE(A.DB2_CPU)  
            +SECOND(A.DB2_CPU)  
            +DEC(MICROSECOND(A.DB2_CPU),15,7)/1000000.0,15,6))  
            AS CPU_SECONDS  
      , SUM(DEC(3600*HOUR(A.DB2_CPU)  
            +60*MINUTE(A.DB2_CPU)  
            +SECOND(A.DB2_CPU)  
            +DEC(MICROSECOND(A.DB2_CPU),15,7)/1000000.0,15,6))  
            /SUM(A.SQL_CALLS)  
            AS CPU_SECONDS_PER_CALL  
      , SUM(DEC(3600*HOUR(A.DB2_ELAPSED)  
            +60*MINUTE(A.DB2_ELAPSED)  
            +SECOND(A.DB2_ELAPSED)  
            +DEC(MICROSECOND(A.DB2_ELAPSED),15,7)/1000000.0,15,6))  
            AS ELAPSED_SECONDS  
      , SUM(DEC(3600*HOUR(A.DB2_ELAPSED)  
            +60*MINUTE(A.DB2_ELAPSED)  
            +SECOND(A.DB2_ELAPSED)  
            +DEC(MICROSECOND(A.DB2_ELAPSED),15,7)/1000000.0,15,6))  
            /SUM(A.SQL_CALLS)  
            AS ELAPSED_SECONDS_PER_CALL FROM  
      SYSTOOLS.CQM21_SUMM_METRICS A  
WHERE (A."COLLECTION" = 'SU2005B')  
GROUP BY A."PROGRAM", A.STATEMENT_NUMBER  
ORDER BY 5 DESC
```

Figure 54: Summary metrics for all statements in a collection

## Summary metrics including object access for a single statement.

```
SELECT B.PAGESET_NAME  
      , SUM(DEC(3600*HOUR(B.GETPAGE_ELAPSED)  
            +60*MINUTE(B.GETPAGE_ELAPSED)  
            +SECOND(B.GETPAGE_ELAPSED)  
            +DEC(MICROSECOND(B.GETPAGE_ELAPSED),15,7)/1000000.0,15,6))  
            AS ELAPSED_SECONDS  
      , SUM(B.GETPAGES          ) AS GETPAGES  
      , SUM(B.SYNC_READS        ) AS SYNC_READS  
      , SUM(B.SEQ_PREFETCH      ) AS SEQ_PREFETCH  
      , SUM(B.LIST_PREFETCH     ) AS LIST_PREFETCH  
      , SUM(B.DYNAMIC_PREFETCH  ) AS DYNAMIC_PREFETCH  
      , SUM(B.ASYNCH_PAGES_READ ) AS ASYNCH_PAGES_READ  
FROM SYSTOOLS.CQM21_SUMM_METRICS A  
     INNER JOIN  
     SYSTOOLS.CQM21_SUMM_OBJECTS B  
     ON B.METRICS_TOKEN = A.METRICS_TOKEN  
WHERE (A."PROGRAM" = &Program)  
      AND (A.STATEMENT_NUMBER = &Statement_No)  
      AND (B.DATABASE_NAME = 'SU2005B')  
GROUP BY B.PAGESET_NAME  
ORDER BY 3 DESC
```

Figure 55: Summary metrics including object access for a single statement

## Summary metrics for all statements that access a single table.

```

SELECT M."PROGRAM"
, M.STATEMENT_NUMBER AS STMT_NO
, S.STATEMENT_DESC AS STMT_DESC
, SUM(M.SQL_CALLS ) AS SQL_CALLS
, SUM(DEC(3600*HOUR(M.DB2_CPU)
+60*MINUTE(M.DB2_CPU)
+SECOND(M.DB2_CPU)
+DEC(MICROSECOND(M.DB2_CPU),15,7)/1000000.0,15,6))
AS CPU_SECONDS
, SUM(DEC(3600*HOUR(M.DB2_CPU)
+60*MINUTE(M.DB2_CPU)
+SECOND(M.DB2_CPU)
+DEC(MICROSECOND(M.DB2_CPU),15,7)/1000000.0,15,6))
/SUM(M.SQL_CALLS)
AS CPU_SECONDS_PER_CALL
, SUM(DEC(3600*HOUR(M.DB2_ELAPSED)
+60*MINUTE(M.DB2_ELAPSED)
+SECOND(M.DB2_ELAPSED)
+DEC(MICROSECOND(M.DB2_ELAPSED),15,7)/1000000.0,15,6))
AS ELAPSED_SECONDS
, SUM(DEC(3600*HOUR(M.DB2_ELAPSED)
+60*MINUTE(M.DB2_ELAPSED)
+SECOND(M.DB2_ELAPSED)
+DEC(MICROSECOND(M.DB2_ELAPSED),15,7)/1000000.0,15,6))
/SUM(M.SQL_CALLS)
AS ELAPSED_SECONDS_PER_CALL
FROM CQM.CQM21_SUMM_OBJECTS O
, CQM.CQM21_SUMM_METRICS M
, CQM.CQM21_STMT_TYPES S
WHERE O.METRICS_TOKEN = M.METRICS_TOKEN
AND S.STATEMENT_TYPE = M.STATEMENT_TYPE
AND M.COLLECTION = 'SU2005B'
AND TBCreator = &TBCreator
AND TBName = &TBName
GROUP BY M.PROGRAM, M.STATEMENT_NUMBER, S.STATEMENT_DESC
ORDER BY 4 DESC

```

Figure 56: Summary metrics for all statements that access a single table

## View Definition to change all timestamps to decimal in the summary metrics table.

```

CREATE VIEW "SYSTOOLS"."CQM21_SUMM_METRS_V" AS
SELECT INTERVAL_NUMBER
, INTERVAL_START
, INTERVAL_END
, SMFID
, DB2_SUBSYSTEM
, DB2_VERSION
, CQM_SUBSYSTEM
, CQM_VERSION
, "PLAN"
, "COLLECTION"
, "PROGRAM"
, SECTION
, STATEMENT_NUMBER
, STATEMENT_TYPE
, CORRID
, AUTHID
, WORKSTATION_USER
, WORKSTATION_TRAN
, WORKSTATION_NAME
, CONSISTENCY_TOKEN
, TEXT_TOKEN
, METRICS_TOKEN
, DEC(3600*HOUR(DB2_CPU )+60*MINUTE(DB2_CPU) +SECOND(DB2_CPU)
+DEC(MICROSECOND(DB2_CPU),15,7)/1000000.0,15,6) AS DB2_CPU
, DEC(3600*HOUR(DB2_ELAPSED )+60*MINUTE(DB2_ELAPSED) +SECOND(DB2_ELAPSED)
+DEC(MICROSECOND(DB2_ELAPSED),15,7)/1000000.0,15,6) AS DB2_ELAPSED

```

```

, SQL_CALLS
, DEC(3600*HOUR(LOCK_LATCH_DELAY )+60*MINUTE(LOCK_LATCH_DELAY)
+SECOND(LOCK_LATCH_DELAY )+DEC(MICROSECOND(LOCK_LATCH_DELAY)
,15,7)/1000000.0,15,6) AS LOCK_LATCH_DELAY
, DEC(3600*HOUR(SYNC_IO_DELAY)+60*MINUTE(SYNC_IO_DELAY)
+SECOND(SYNC_IO_DELAY)+DEC(MICROSECOND(SYNC_IO_DELAY)
,15,7)/1000000.0,15,6) AS SYNC_IO_DELAY
, DEC(3600*HOUR(OTHER_READ_DELAY )+60*MINUTE(OTHER_READ_DELAY)
+SECOND(OTHER_READ_DELAY )+DEC(MICROSECOND(OTHER_READ_DELAY)
,15,7)/1000000.0,15,6) AS OTHER_READ_DELAY
, DEC(3600*HOUR(OTHER_WRITE_DELAY)+60*MINUTE(OTHER_WRITE_DELAY)
+SECOND(OTHER_WRITE_DELAY)+DEC(MICROSECOND(OTHER_WRITE_DELAY)
,15,7)/1000000.0,15,6) AS OTHER_WRITE_DELAY
, DEC(3600*HOUR(SERVTASK_SW_DELAY)+60*MINUTE(SERVTASK_SW_DELAY)
+SECOND(SERVTASK_SW_DELAY)+DEC(MICROSECOND(SERVTASK_SW_DELAY)
,15,7)/1000000.0,15,6) AS SERVTASK_SW_DELAY
, DEC(3600*HOUR(ARCHLOG_QS_DELAY)+60*MINUTE(ARCHLOG_QS_DELAY)
+SECOND(ARCHLOG_QS_DELAY)+DEC(MICROSECOND(ARCHLOG_QS_DELAY)
,15,7)/1000000.0,15,6) AS ARCHLOG_QS_DELAY
, DEC(3600*HOUR(ARCHLOG_RD_DELAY )+60*MINUTE(ARCHLOG_RD_DELAY)
+SECOND(ARCHLOG_RD_DELAY )+DEC(MICROSECOND(ARCHLOG_RD_DELAY)
,15,7)/1000000.0,15,6) AS ARCHLOG_RD_DELAY
, DEC(3600*HOUR(DRAIN_LOCK_DELAY)+60*MINUTE(DRAIN_LOCK_DELAY)
+SECOND(DRAIN_LOCK_DELAY)+DEC(MICROSECOND(DRAIN_LOCK_DELAY)
,15,7)/1000000.0,15,6) AS DRAIN_LOCK_DELAY
, DEC(3600*HOUR(CLAIM_REL_DELAY )+60*MINUTE(CLAIM_REL_DELAY)
+SECOND(CLAIM_REL_DELAY )+DEC(MICROSECOND(CLAIM_REL_DELAY)
,15,7)/1000000.0,15,6) AS CLAIM_REL_DELAY
, DEC(3600*HOUR(PAGE_LATCH_DELAY)+60*MINUTE(PAGE_LATCH_DELAY)
+SECOND(PAGE_LATCH_DELAY)+DEC(MICROSECOND(PAGE_LATCH_DELAY)
,15,7)/1000000.0,15,6) AS PAGE_LATCH_DELAY
, DEC(3600*HOUR(SP_DELAY )+60*MINUTE(SP_DELAY)
+SECOND(SP_DELAY)+DEC(MICROSECOND(SP_DELAY)
,15,7)/1000000.0,15,6) AS SP_DELAY
, DEC(3600*HOUR(NOTIFY_MSGS_DELAY)+60*MINUTE(NOTIFY_MSGS_DELAY)
+SECOND(NOTIFY_MSGS_DELAY)+DEC(MICROSECOND(NOTIFY_MSGS_DELAY)
,15,7)/1000000.0,15,6) AS NOTIFY_MSGS_DELAY
, DEC(3600*HOUR(GLOBAL_CONT_DELAY)+60*MINUTE(GLOBAL_CONT_DELAY)
+SECOND(GLOBAL_CONT_DELAY)+DEC(MICROSECOND(GLOBAL_CONT_DELAY)
,15,7)/1000000.0,15,6) AS GLOBAL_CONT_DELAY
, LOCK_LATCH_EVENTS
, SYNC_IO_EVENTS
, OTHER_READ_EVENTS
, OTHER_WRITE_EVENTS
, SERVTASK_SW_EVENTS
, ARCHLOG_QS_EVENTS
, ARCHLOG_RD_EVENTS
, DRAIN_LOCK_EVENTS
, CLAIM_REL_EVENTS
, PAGE_LATCH_EVENTS
, SP_EVENTS
, NOTIFY_MSGS_EVENTS
, GLOBAL_CONT_EVENTS
, LOCK_DEADLOCKS
, LOCK_SUSPENSIONS
, LOCK_TIMEOUTS
, LATCH_SUSPENSIONS
, OTHER_SUSPENSIONS
, LOCK_REQUESTS
, UNLOCK_REQUESTS
, QUERY_REQUESTS
, CHANGE_REQUESTS
, OTHER_REQUESTS
, CLAIM_REQUESTS
, CLAIM_FAILED
, DRAIN_REQUESTS
, DRAIN_FAILED
, XES_LOCK_REQUESTS
, SYNC_READS
, SEQ_PREFETCH
, SYNC_WRITES
, LIST_PREFETCH
, DYNAMIC_PREFETCH
, HPOOL_READS
, HPOOL_READS_FAIL
, HPOOL_WRITES

```

```
, HPOOL_WRITES_FAIL
, GETPAGES_FAILED
, ASYNCH_PAGES_READ
, ASYNCH_HPOOL_PAGES
FROM
"SYSTOOLS"."CQM21_SUMM_METRICS"
```

**Figure 57: View Definition to change all timestamps to decimal in the summary metrics table**

## Chapter 7: Frequently Asked Questions

In this section we answer a number of Frequently Asked Questions (FAQ's) about Query Monitor:

- What is meant by a 'workload'?
- What kind of overhead can I expect from running Query Monitor?
- What happens when a Query Monitor components fails?
- What happens during interval processing?
- Do I have to use the TCP/IP port number 3444 for the CAE Agent?
- Is the order in which the QM Data Collector, the CAE Agent and the CAE Server are started important?
- What is the difference between Exceptions and Alerts?
- What data is stored on the z/OS host and what data is stored on the CAE Server?
- How do I manage user ids via the CAE Server?
- Can I monitor multiple releases of DB2 from a single QM Data Collector?
- I am unable to use the ISPF interface because I get the message "CQM244E Unable to determine terminal CCSID. The Coded Character Set Identifier (CCSID) of the terminal cannot be determined." What can I do?
- Sometimes, when viewing call level statistics for an exception negative SQLCODE, I see multiple detail lines for that negative SQLCODE. What does this mean?
- How can I tell from that an SQL statement is dynamic when viewing negative SQLCODEs?
- What is Lock Avoidance and how can Query Monitor tell me whether I am benefiting from it or not?
- What does an '\*' next to the 'Occurrences' column mean on the 'DB2 QM SQL Code Summary' ISPF panel mean?
- How do I tell which threshold caused an alert or exception event to be triggered?
- How should I use the information about synchronous reads and writes in the Buffer Pool Statistics display?
- What other information can I glean from the Buffer Pool Statistics display?
- Can the QM CAE Server be configured so that when the CAE Server machine is rebooted, the CAE Server automatically starts up?

### What is meant by a 'workload'?

A workload is a means of identifying a group of applications to Query Monitor so that performance data can be collected for SQL statements executed by those applications. Workloads are defined in the monitoring profile used by Query Monitor to monitor a given DB2 subsystem. Broadly, workloads can be said to have three characteristics:

- The workload name, which must be unique and is used purely to identify the workload.

- The workload filters, in which identifiers such as plan name, subsystem name, authorization id and so on are used to identify which applications to include or exclude from data collection.
- The thresholds for collecting exceptions and alerts, such as elapsed time, CPU time or GETPAGEs.

Please see the chapter on Data Collection for more details on monitoring profiles and how to create them.

### **What kind of overhead can I expect from running Query Monitor?**

This is very difficult to say. This is because the overhead is dependent on so many factors:

- Application characteristics, including such things:
  - The types of SQL call (dynamic/static, fetch intensive/non-fetch intensive).
  - SQL call volumes, by type.
  - Fetch intensive applications are likely to incur greater overhead than non-fetch intensive ones.
  - Whether the application is update intensive or read intensive.
  - Whether the application is batch or online.
  - For online applications, whether it is an OLTP or Data Warehouse application.
  - Whether the SQL is simple or complex.
  - The COMMIT frequency.
- What is specified in COMPARMS, including OPTKEYS. The greater the number of OPTKEYS you specify, the greater the overhead is likely to be. Please see the chapter on Data Collection for more details.
- The number of DB2 objects (databases, table spaces, index spaces) for which data is collected.
- The monitoring profiles in place, which determine:
  - Which workloads you collect summary data for.
  - The workloads you monitor for exceptions and alerts.
  - Which SQLCODEs are excluded from the data collection process for the workloads defined in the monitoring profile.
- The number of exceptions and alerts captured.

The larger the number of workloads for which you collect data, the larger the total overhead on your system will be.

In summary, there is no formula for calculating the overhead, but the more data you collect and the greater the amount of detail, the greater the overhead is likely to be.

### **What happens when a Query Monitor component fails?**

If the CAE Server fails, all alerts in the Alert Browser Message Boards are lost. Any new alerts are queued up for the CAE Agent until the alert limit is reached. When the alert limit is reached, any subsequent alerts are discarded. This does not affect any other data collected by the QM Data Collector.

If the CAE Agent fails no alerts will be sent to the CAE Server until the CAE Agent is restarted. This failure does not affect the QM Data Collector and so QM will continue to collect data and store it in its data spaces.

If the QM Data Collector fails, no new data is collected until the Data Collector is successfully restarted. Any data in the QM data spaces is lost but data already externalised is retained. You may want to consider this when determining interval size. This does not affect the alerts on the Alert Browser Message Boards in the CAE Server.

If the CAE GUI fails, no data is lost, either from the CAE Server or from the data collected by the QM Data Collector.

If DB2 itself fails, then the QM Data Collector will wait for DB2 to be restarted. You should see the following messages, at shutdown (or time of failure) and restart respectively:

```
CQM3003I DB2 SHUTDOWN DETECTED FOR SUBSYSTEM ISC3
CQM3001I DB2 STARTUP DETECTED FOR SUBSYSTEM ISC3
```

If you have decided, for whatever reason, to have one QM Data Collector per DB2 subsystem, then you may want to implement some automation to stop and start the QM Data Collector at the same time as DB2, probably starting it before DB2 restart and stopping it after DB2 shutdown to ensure no loss of performance data.

### What happens during interval processing?

An interval switch occurs when the time allotted to an interval is exceeded or when the dataspace fills. The length of an interval is defined using the parameter INTERVAL in the CQMPARMS dataset.

When an interval switch occurs, operational and structural data relations stored in the dataspace have to be written to the backstore VSAM datasets.

QM records this process using a paired set of messages written using a WTO:

```
CQM2400I INTERVAL PROCESSING STARTED FOR QM21
CQM2401I INTERVAL PROCESSING ENDED FOR QM21
```

QM21 represents the subsystem name of the QM Data Collector and may well be different in your case.

The first message indicates that interval processing begins and the second message is written once interval processing completes. Normally, the time it takes to complete interval processing should end quickly but really depends on the volume of data being collected. The more data collected the longer this process takes to complete. On syslog the messages are all time-stamped by MVS.

The following table lists the most important events associated with interval processing:

Event	Description
Data is written from the summary-based data spaces to the VSAM data sets	The greater the volume of data the more time needed
Close old VSAM data sets	Data sets for the prior interval are closed
Allocate new VSAM data sets	Data sets for the new interval are dynamically allocated
Delete of expired VSAM data sets	Data sets that have reached the RETAIN limit are deleted
Reinitialise summary buckets	Summary buckets are reinitialised for the new interval.

Figure 58: Interval processing events

### Do I have to use the TCP/IP port number 3444 for the CAE Agent?

Generally, the answer is yes. Changing the product to use another port number is very complicated. If port 3444 has already been reserved for another application, then you have 3 choices:

4. Change the application that has reserved port 3444 to use another port.
5. Wait for the availability of a planned PTF which will allow you to specify the port number. At the time of writing the PTF number is unknown.
6. Contact IBM and ask *if* they can provide a means of changing the CAE agent so that it uses another port.

### Is the order in which the QM Data Collector, the CAE Agent and the CAE Server are started important?

Not at all. The QM Data Collector doesn't care whether the CAE Agent is there or not, and is ignorant of the CAE Server. If the CAE Agent starts up and the QM Data Collector is not there, it simply waits for it to start up. Remember, the CAE Agent detects QM Data Collectors on the same LPAR, and multiple QM Data Collectors per CAE Agent are allowed. If the CAE Server is not active when the CAE Agent is started up, it waits until it can detect that the CAE Server has been started before connecting to it. The CAE Server doesn't care if no CAE Agents have connected to it - it simply waits to be contacted. Again, there can be zero to many CAE Agents connecting to any given CAE Server.

### What is the difference between Exceptions and Alerts?

Exceptions and alerts have two very different purposes. Exceptions are designed to allow you to collect information about SQL statements that are causing potential performance problems. Alerts are designed to identify problems that require immediate action. Therefore, it is desirable that the number of alerts represent a small proportion of the number of exceptions.

The QM ISPF interface only reports on exceptions. For any given SQL statement, the first measured performance value that exceeds one of the



exception values in a matching monitoring profile line is highlighted in yellow on the 'Display Exceptions' panel (CQM\$EXCA).

There is no option in the ISPF interface to allow you to look specifically at alerts. Information about alerts that are not also exceptions is not available. Alerts that are also exceptions are, however, highlighted in red on the 'Display Exceptions' panel.

The Alert Browser in the CAE GUI only displays alerts, not exceptions. Exceptions can be viewed from the CAE GUI using the Exceptions perspective in the Activity Browser. Again, alerts that are not also exceptions are not viewable through the Exceptions perspective in the Activity Browser.

When defining alerts, it is always a good idea to make sure that they are also defined as being exceptions, with the exception thresholds being less than or equal to the alert threshold. This allows you to view the alerts via the ISPF interface.

### **What SQL performance data is stored on the z/OS host and what data is stored on the CAE Server?**

The SQL performance data stored on the z/OS host (in the VSAM backstore data sets) is as follows:

- All summary data
- All data about negative SQLCODEs
- All data about exceptions
- Where alerts also qualify as exceptions, data about alerts
- Information about DB2 commands

The SQL Performance data stored on the CAE Server is as follows:

- Alert detail data, which is presented on the Alert Browser message boards.

### **How do I manage user ids via the CAE Server?**

This information is available in the *IBM DB2 Query Monitor for z/OS, User's Guide*, SC18-9202. Here is a summary of that information:

The only user ids that can be managed via the CAE Server are GUI Client user ids, and the only way these ids can be managed is to use the GUI Client to connect to the CAE Server. Of course the only ids that can administer user ids are those defined with an Administrator role.

To add a user, do the following:

From the GUI Client, select the Tools menu-bar item, and from the drop-down select 'Profiles and Configurations ...'. This will open up a new window, and from the menu on the left side of this window select Users. Click on the Add button at the top of the left hand pane, and enter the user id you wish to add in the pop-up window. You then define the characteristics of the user id,

including full name, password, and role, all of which are required, and optionally a description:

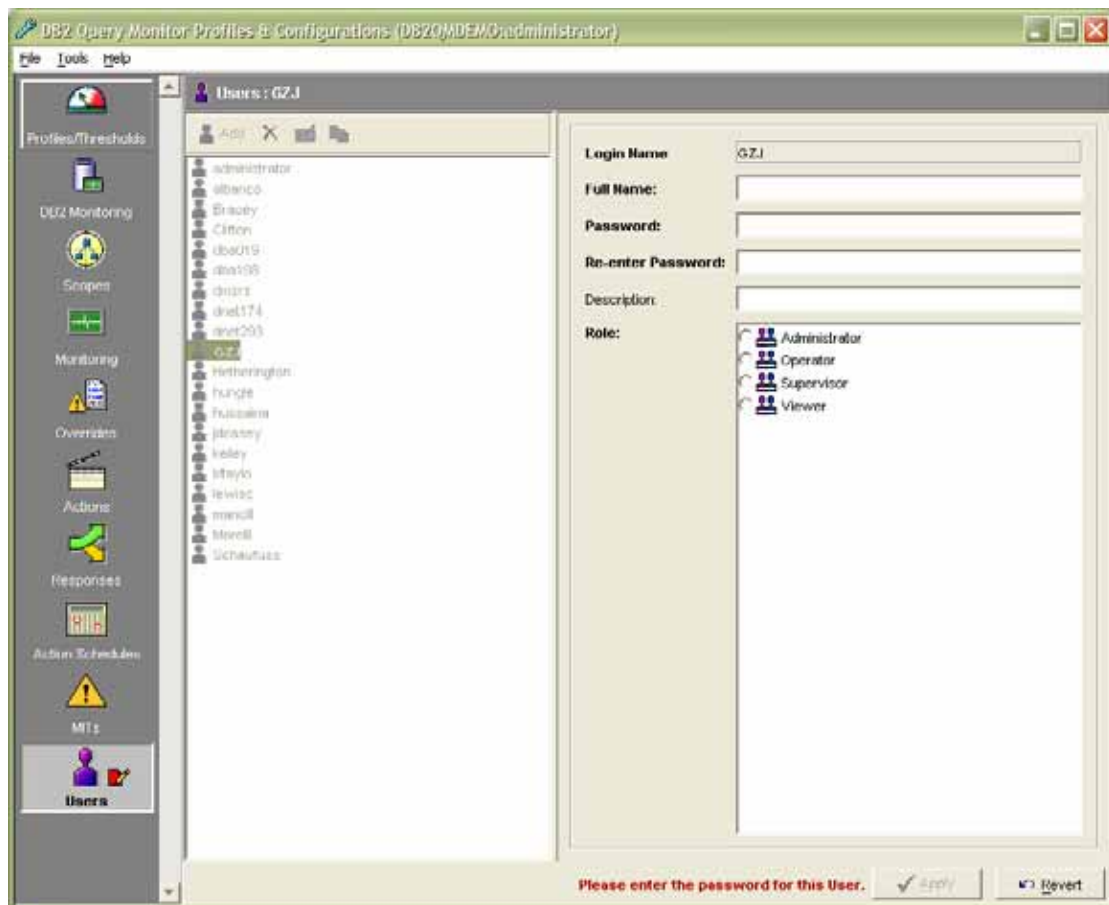


Figure 59: GUI Client user administration

When all the details are complete, click the Apply button at the bottom of the screen to make the change effective, or the revert button to cancel the change.

To change the details for a user, simply left-click on the user id in the left-hand pane, make the desired changes in the right-hand pane, and click Apply.

To remove a user, left-click on the user id in the left-hand pane, and click the delete button (the cross) above the user id list.

You can also rename a user id or clone it using the buttons above the list of users.

### Can I monitor multiple releases of DB2 from a single QM Data Collector?

Yes, you can, provided they are on the same LPAR, of course. You need a minimum of one QM Data Collector per LPAR and that Data Collector can monitor SQL statements executing in DB2 for z/OS subsystems from DB2 V6.1 to DB2 V8.1.

I am unable to use the ISPF monitor because I get the message "COM244E Unable to determine terminal CCSID. The Coded Character Set Identifier (CCSID) of the terminal cannot be determined." What can I do?

This problem arises when the QM ISPF interface is unable to determine the CCSID used by your terminal (usually a 3270 emulator). It needs this information to be able to perform all the conversions necessary, and does this by querying an ISPF variable, ZTERMCID, which normally contains the terminal CCSID. Sometimes, QM cannot do this because the necessary VTAM definitions are not in place and ZTERMCID contains nulls. As a workaround, until the definitions are updated, you can use a feature introduced by APAR PQ96050 which allows you to specify the CCSID when invoking calling QOM\$MAIN from QOMCLIST. This allows you to add a value for ZTERMCID when invoking QOM\$MAIN:

```
QOM$MAIN ZTERMCID(nnnn)
```

where 'nnnn' is the desired value for ZTERMCID. This should ideally used only on a temporary basis, until the necessary VTAM definitions have been updated.

Sometimes, when viewing call level statistics for an exception negative SQLCODE, I see multiple detail lines for that negative SQLCODE. What does this mean?

When investigating an exception, where that exception is a negative SQL code, you can see some interesting behaviour if the application doesn't handle negative SQL codes appropriately:

```

2005/02/11 13:08:29 ----- Display Exceptions ----- Row 1 of 12
Option ==> Scroll ==> PAGE
DB2 QM Subsystem: QMV2 Interval Start Date: 02/11/2005 Time: 08:00:04
Filters Enabled: N Interval End Date: CURRENT Time: CURRENT
C:A-Analyze,B-Buffers,C-Calls,D-Delays,L-Locks,O-Objects,S-SQL Text
----->
CMD SSID Plan Program DB2 CPU Time DB2 Elapsed GETPAGES SQLCODE
-----
- DBDM DSNEsprR DSNEsm68 0.002703 0.002904 22 -601
- DBDM DSNEsprR DSNEsm68 0.002570 0.003000 20 -601
- DBDM INZPLAN1 INZACDB2 0.157457 2.505669 1,897 +100
- DBDM DSNTIAUL DSNTIAUL 0.001863 0.003981 2 -204
- DBDM ADB ADBMAIN 0.117961 0.954451 3,602 0
- DBDM ADB ADBMAIN 0.090737 1.955602 6,298 +100
c DBDM DSNREXX DSNREXX 0.001958 0.004905 2 -204
- DBDM DSNREXX DSNREXX 0.004093 0.008137 27 -551
- DBDM ADB ADBMAIN 0.099337 2.226061 2,718 +100
- DBDM ADB ADBMAIN 0.098924 2.341057 2,694 +100
- DBDM ALASQLP ALASQL6 0.002649 0.002876 9 -601
- DBDM ALASQLP ALASQL6 0.003121 0.006711 23 -204
***** Bottom of Data *****

```

Figure 60: The Exceptions display for negative SQL codes

With the exceptions displayed, use the 'C' (Calls) line command to bring up information about the call level statistics:

```

2005/02/11 13:14:07 ----- Call Level Statistics ----- Row 1 of 4
Option ==> _____ Scroll ==> PAGE

DB2 SSID: DBDM   Plan: DSNREXX   DBRM: DSNREXX   Coll: DSNREXCS
Cursor: PRIME   Section:      9

C:B-Buffers,C-SQLCA,D-Delays,L-Locks,H-Host Variables,S-Call Text
----- >
CMD STMT#  Call Type          SQLCODE   DB2 CPU Time   DB2 Elapsed
-----
-   4,759  CLOSE                   -514       0.000014      0.000014
-   4,735  FETCH                    -518       0.000020      0.000020
-   4,733  OPEN                      -514       0.000020      0.000020
-   4,731  PREPARE                    -204       0.001902      0.004848
***** Bottom of Data *****

```

**Figure 61: Call-level statistics for negative SQL codes**

As the negative SQL code was not correctly handled at PREPARE time, and the application went on to issue an OPEN, FETCH and CLOSE for the cursor referenced in the PREPARE, the negative SQL codes returned as a result of the first failure are listed on this panel. These are not identified as exceptions, because only the first negative SQL code will cause Query Monitor exception processing to be triggered.

**How can I tell from that an SQL statement is dynamic when viewing negative SQLCODES?**

It is possible, when viewing SQL Codes (option 'N' from the QM ISPF main menu), to identify that the SQL statement was dynamic. From the DB2 QM SQL Code Summary panel, select the negative SQL code you're interested in (line command 'S'), then from the DB2 QM SQL Code Detail Display panel select SQL Text (option 'S', again) to see the Display SQL Statement Text panel:

```

2005/02/11 11:45:08 ---- Display SQL Statement Text ---- Row 1 of 1
Option ==> _____ Scroll ==> PAGE

DB2 SSID: DBDM   Plan: DSNTEP71   DBRM: DSNTEP2   Coll: DSNTEP2
Cursor: C1      Section:      1

-----

SELECT COUNT(*) FROM SYSIBM.SYSTABLEPART, SYSIBM.SYSTABLES
***** Bottom of Data *****

```

**Figure 62: Displaying SQL Statement Text**

You can see from the summary information above the SQL text that the SQL statement is described as being a cursor, but that the statement text is neither an OPEN nor a FETCH, but the SELECT statement associated with the cursor (in this example, cursor C1). This indicates that the SQL Code was issued for a dynamic SQL statement.

## What is Lock Avoidance and how can Query Monitor tell me whether I am benefiting from it or not?

LOCK AVOIDANCE was introduced in DB2 V3 to minimize the number of IRLM requests. For plans and packages bound with CURRENTDATA(NO), DB2 can test a row or page to see if contains committed data only. If this is true, DB2 does not have to obtain a lock on the data at all. Non-locked data is returned to the application, and in cursor-based processing can be changed while the cursor is positioned on the row.

Some other lock avoidance features in DB2 are: single lock on a partitioned table space in V4, type 2 index lock avoidance in V4, uncommitted read (UR) in V4, and selective partition locking (V5).

Query Monitor can help you identify when lock avoidance has been successful, when you use the 'L' (Locks) line command from any of the valid panels. If the number of lock requests is greater than the number of unlock requests, then lock avoidance has been successful:

```
2005/02/11 13:45:48 ----- Lock Related Statistics ----- Row 1 of 23
Option ==> _____ Scroll ==> PAGE

DB2 SSID: DBDM   Plan: DSNREXX   DBRM: DSNREXX   Coll: DSNREXX
Cursor:                Section: 202

-----
Lock Event                      Event Count
Lock Deadlocks                   0
Lock Suspensions                 0
Lock Timeouts                    0
Latch Suspensions                0
Other Suspensions                0
Lock Requests                    225
Unlock Requests                  70
Query Requests                   0
Change Requests                  7
Other Requests                   0
Claim Requests                   40
Claim Failures                   0
Drain Requests                   0
Drain Failures                   0
XES Lock Requests                0
XES Change Requests              0
XES Unlock Requests              0
IRLM Global Resource Contention  0
XES Global Resource Contention  0
False Resource Contention        0
Incompatible Retain Lock         0
Shared Lock Escalations          0
Exclusive Lock Escalations       0
***** Bottom of Data *****
```

Figure 63: Lock-related statistics

You can evaluate the effect of lock avoidance in your environment by looking at UNLOCK requests per COMMIT and compare with LOCK requests per COMMIT.

As a rule-of-thumb, you can say that if the number of UNLOCK requests per commit is greater than five and is also greater than one third of the number of

LOCK requests, then most likely your lock avoidance is not working well. A typical reason is the use of the BIND parameter CURRENTDATA(YES).

### What does an '\*' next to the 'Occurrences' column mean on the 'DB2 QM SQL Code Summary' ISPF panel mean?

When viewing negative SQLCODEs through the ISPF interface, you may see an '\*' next to one of the values in the 'Occurrences' column in the 'DB2 QM SQL Code Summary' panel:

```

2005/02/18 12:07:59 ----- DB2 QM SQL Code Summary ----- Row 1 of 5
Option ==> Scroll ==> PAGE
DB2 QM Subsystem: QMV2 Interval Start Date: 02/17/2005 Time: 14:34:35
Interval End Date: CURRENT Time: CURRENT
Group by S (SQLCODE, AuthID, DBRM/Package, Plan)
Specify "*" for no grouping
C: A-AUTHIDs, D-DBRMs/Packages, P-Plans, S-Select
-----
CMD SQL Code Occurrences
- - - - -
- -803 1
- -904 1
- -551 1
- -443 3
- -501 32 *
***** Bottom of Data *****

```

Figure 64: The SQL Code Summary panel

Next to the 'Occurrences' value for SQLCODE -501, the '\*' indicates that the value specified for MAX\_SQLCODE\_DETAIL in CQMPARMS has been exceeded. When you issue any of the available line commands, the number of detail lines you see is equal to MAX\_SQLCODE\_DETAIL (in the example below, this is equal to 5):

```

2005/02/18 14:28:12 --- DB2 QM SQL Code Detail Display --- Row 1 of 5
Option ==> Scroll ==> PAGE
DB2 QM Subsystem: QMV2 Interval Start Date: 02/17/2005 Time: 14:34:35
Interval End Date: 02/18/2005 Time: 14:00:02
Detail data captured for SQLCODE: -501
C: C-SQLCA,S-SQL text
-----
CMD SSID Plan DBRM/Package JOBNAME Stmt # Collection ID Sect# Authid
- - - - -
- DBDM DSNREXX EMPUPD2 DBDMWLM1 676 DMMJBL01 2 DMMJB
- DBDM DSNREXX DPTMGR2 DBDMWLM1 533 DMMJBL01 2 DMMJB
- DBDM DSNREXX PRJADD2 DBDMWLM1 1,259 DMMJBL01 2 DMMJB
- DBDM DSNREXX EMPUPD2 DBDMWLM1 676 DMMJBL01 2 DMMJB
- DBDM DSNREXX PRJADD2 DBDMWLM1 1,259 DMMJBL01 2 DMMJB
***** Bottom of Data *****

```

Figure 65: The SQL Code Detail Display

## How do I tell which threshold caused an alert or exception event to be triggered?

On the 'Display Exceptions' panel, scroll right until you see two columns called 'EXCEPTIONS' and 'ALERTS'. This tells you which threshold(s) caused the exception and/or alert condition to be triggered. The values in this column have the following meaning:

- C for CPU
- E for Elapsed
- G for Getpages
- S for SQL count
- N for negative SQL code

```
2005/02/24 15:23:22 ----- Display Exceptions ----- Row 1 of 4
Option ==>                                           Scroll ==> PAGE
DB2 QM Subsystem: QMV2           Interval Start Date: 02/24/2005   Time: 14:00:00
Filters Enabled: N               Interval End   Date: CURRENT     Time: CURRENT
C:A-Analyze,B-Buffers,C-Calls,D-Delays,L-Locks,O-Objects,S-SQL Text
----- < >
CMD  SSID  EXCEPTIONS  ALERTS      WORKLOAD
-----
-   -
-   DBDM  G          G          CPU > 2s Elap > 1m GP > 1500
-   DBDM  G          G          CPU > 2s Elap > 1m GP > 1500
-   DBDM  G          G          CPU > 2s Elap > 1m GP > 1500
-   DBDM  CEG       CEG       CPU > 2s Elap > 1m GP > 1500
***** Bottom of Data *****
```

Figure 66: The Display Exceptions panel

In the example above, we can see that in three cases, the cause was getpages, and in the fourth, it was CPU time, elapsed time and getpages. Be aware that the flags are always listed in the order given above - no indication is provided of the order in which the thresholds were exceeded (other than the fact that, if you follow the recommendations in this document, alert thresholds will always be higher than exception thresholds, and therefore exception thresholds will always be triggered first). This information is in addition to the colour-coding used on the performance metrics fields, where a triggered field is displayed in yellow to indicate an exception and in red to indicate an alert.

## How should I use the information about synchronous reads and writes in the Buffer Pool Statistics display?

A good indicator of a probable performance problem can be obtained from the buffer pool display, obtained by issued the 'B' line command from any of the valid panels. A non-zero value for synchronous reads, but more especially a high value compared to the number of asynchronous reads and/or getpage requests indicates DB2 suspensions, and therefore extended elapsed times, which could possibly be avoided:

```

2005/02/11 13:48:33 ----- Buffer Pool Statistics ----- Row 1 of 16
Option ==> _____ Scroll ==> PAGE

DB2 SSID: DBDM    Plan: DSNREXX    DBRM: DSNREXX    Coll: DSNREXX
Cursor:                Section:    202
-----
Buffer Pool: ALL
Buffer Pool Hit Ratio (%)      Total      Average
Hiper Pool Hit Ratio (%)      N/A        100.00
Get Page Requests              8,671      8,671.00
Buffer Pages Updated            742        742.00
Synchronous Pages Read         10         10.00
Synchronous Pages Written      0          0.00
Sequential Prefetch Requests   0          0.00
List Prefetch Requests         0          0.00
Dynamic Prefetch Requests      0          0.00
Successful Hiper Pool Reads    0          0.00
Hiper Pool Read Failures       0          0.00
Successful Hiper Pool Writes   0          0.00
Unsuccessful Hiper Pool Writes 0          0.00
Async Pages Read                0          0.00
Async Pages Read by Hiper Pool 0          0.00
***** Bottom of Data *****

```

**Figure 67: Buffer Pool Statistics**

Synchronous writes are expensive (normally DB2 uses asynchronous writes) and are caused when DB2 has to externalize changed pages to DASD during physical close. DB2 will issue a synchronous write for those changed pages that remain in the bufferpool after 2 checkpoints. Sync writes occur if the 'Data Manager critical threshold (95%)' is reached. When this occurs, it is likely the 'Synchronous write threshold (97.5%)' would have been reached also.

Sync writes should be avoided at all costs as it can add significantly to the elapsed time for an SQL statement.

You can expect, however, to see a synchronous write for a COMMIT or ABORT SQL request as a synchronous write to the DB2 log is required for the synch point to be established.

### **What other information can I glean from the Buffer Pool Statistics display?**

When looking at buffer pool statistics, there are some things to bear in mind. The figures on this panel represent:

- The total number of requests for the summarised data - be aware of the level at which the data is summarised, be it at subsystem, plan, package, SQL statement or any other summarisation level.
- The average number of requests for the summarised data.

These figures should not be confused with those you get from DB2 PM or DB2 PE statistics reports and traces, which represent totals and averages for the buffer pool as a whole.

A negative value in the Buffer Pool Hit Ratio is possible. This happens when a page is read for an application asynchronously, via prefetch, but by the time



the application needs a row on that page, the page has been overlaid by a different page needed by another application process. DB2 then issues a synchronous read for the page that was previously read asynchronously. These synchronous reads are called *sequential synchronous reads*.

Other reasons sequential synchronous read I/Os can occur include:

- Prefetch is disabled (typically, when the sequential prefetch threshold has been reached).
- The pages requested are not consecutive: DB2 has estimated that the selected range of pages is so small that prefetch makes no sense.

It is normal to have a small value for sequential synchronous reads because before the sequential prefetch is scheduled, the first page of a prefetch is read by a synchronous read I/O. However, if this number is large, you should consider increasing the size of the buffer pool or reviewing the sequential steal thresholds VPSEQT and, up to and including DB2 V7.1, HPSEQT.

**Can the QM CAE Server be configured so that when the CAE Server machine is rebooted, the CAE Server automatically starts up?**

This would improve CAE Server availability, and simplify administration of the workstation where it is installed. Currently, the CAE Server does not run as a Windows Service, but IBM is investigating an enhancement to do just this. As a workaround, until the enhancement is available, you can create shortcuts to start the CAE Server and the Agent Console in the Windows Start folder. As described in the chapter on setting up the Consolidation and Analysis Engine, it is highly recommended that the Agent Console be started as well as the CAE Server.

## Bibliography

### Query Monitor manuals

*IBM DB2 Query Monitor for z/OS User's Guide*, SC18-9202

*Program Directory for IBM DB2 Query Monitor for z/OS*, G110-8587

Download the latest copy from:

<http://www-306.ibm.com/software/data/db2imstools/db2tools-library.html>

### DB2 Manuals

*DB2 Universal Database for OS/390 and z/OS Installation Guide Version 7*, GC26-9936

*DB2 Universal Database for z/OS Installation Guide Version 8*, GC18-7418

Download the latest Version 7 documentation from:

<http://www-306.ibm.com/software/data/db2/zos/v7books.html>

Download the latest Version 8 documentation from:

<http://www-306.ibm.com/software/data/db2/zos/v8books.html>

### z/OS Manuals

Note: the manual reference numbers may vary according to your release of z/OS.

*z/OS Support for Unicode Using Conversion Services*, SA22-7649

*z/OS Planning for Installation*, GA22-7504

Download the latest version for your z/OS release from:

<http://www-1.ibm.com/servers/eserver/zseries/zos/bkserv/>