IBM

VisualAge Pacbase 2.5

DSMS 2.5 UNIX
OPERATIONS MANUAL

DEDIX000252A

Note

Before using this document, read the general information under "Notices" on the next page.

According to your license agreement, you may consult or download the complete up-to-date collection of the VisualAge Pacbase documentation from the VisualAge Pacbase Support Center at:
http://www.software.ibm.com/ad/vapacbase/support.htm

Consult the Catalog section in the Documentation home page to make sure you have the most recent edition of this document.

Second Edition (August 1999)

This edition applies to the following licensed program:

- VisualAge Pacbase Version 2.5

Comments on publications (including document reference number) should be sent electronically through the Support Center Web site at:
http://www.software.ibm.com/ad/vapacbase/support.htm
or to the following postal address:
IBM Paris Laboratory
VisualAge Pacbase Support
30, rue du Château des Rentiers
75640 PARIS Cedex 13
FRANCE

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

NOTICES

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

Intellectual Property and Licensing
International Business Machines Corporation
North Castle Drive, Armonk, New-York 10504-1785
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of information which has been exchanged, should contact:

IBM Paris Laboratory
SMC Department
30, rue du Château des Rentiers
75640 PARIS Cedex 13
FRANCE

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

IBM may change this publication, the product described herein, or both.

TRADEMARKS

IBM is a trademark of International Business Machines Corporation, Inc.
AIX, AS/400, CICS, CICS/MVS, CICS/VSE, COBOL/2, DB2, IMS, MQSeries, OS/2, PACBASE, RACF, RS/6000, SQL/DS, TeamConnection, and VisualAge are trademarks of International Business Machines Corporation, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and/or other countries.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.
UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

All other company, product, and service names may be trademarks of their respective owners.

TABLE OF CONTENTS

1. FOREWORD

FOREWORD

USE OF THE MANUAL

This manual is intended for the person in charge of the installation and for the DSMS Database Manager.

It describes the DSMS components, the environment, the batch procedures, the instructions for installing the new version and the operations to be carried out for a standard reinstallation of corrected versions.

NOTE

DSMS 2.5 requires a complete installation of the technical package, i.e. files, programs and batch procedures.

SITES WITH FORMER RELEASES

Once the installation is complete, refer to the relevant chapter for the retrieval of the site's previous release:

- Retrieval of DSMS 8.0.1 and adaptation to DSMS 2.5,
- Retrieval of DSMS 8.0.2 01 or 02 and adaptation to DSMS 2.5,
- Retrieval of DSMS 8.0.2 01 or 02 compatible with VA Pac 8.0.1 and adaptation to DSMS 2.5,
- Retrieval of DSMS 1.2 or 1.5 and adaptation to DSMS 2.5,

and follow the instructions carefully in order to ensure compatibility between the new release and the former one.

2. DSMS COMPONENTS

2.1. INTRODUCTION

INTRODUCTION

DSMS manages permanent data in batch and on-line mode.

Three types of resources are required for the operation of DSMS.

- Libraries which store the DSMS operating programs and system parameters:

  . An on-line program library
  . A batch program library

- Permanent files containing data manipulated by the DSMS system programs previously defined:

  . A system file containing error messages and HELP documentation on DSMS,
  . User files containing the User and Administrator data.

- A library containing the batch procedures.

NOTE: This manual describes the installation and operation of DSMS. DSMS can be installed independently of other VisualAge Pacbase functions and facilities.

For further details on the operation of the Function itself, refer to the DSMS Reference Manual.

2.2. ON-LINE PROGRAMS

ON-LINE PROGRAMS

PROGRAM CODE	ASSOCIATED CHOICE - COMMENTS
xxCHOI	Choice-decoding sub-program
xx00AA	Initial screen
xx00AB	Abend map
xx00BA	HC
xx00B1	C ......
xx00B2	C C
xx00B3	C ...... Q
xx00B4	C ...... M
xx00B5	XS
xx00EA	HE
xx00E1	E ......
xx00E2	C ...... D E ...... DN/DT
xx00E3	C ...... F E ...... FN/FT
xx00E4	C ...... T E ...... T
xx00E5	LCE_
xx00E6	C ...... S E ...... S
xx00FA	HPF
xx00FB	HSC
xx00HE	Help function
xx00JO	JO
xx00KA	HK
xx00K1	LGK_ LAK_
xx00K2	LPK_
xx00K3	WS WU
xx00LE	LDE_ LNC_ LSE_ LDC LNC
xx00LS	LIE ....*...
xx00MA	H (Main Menu)
xx00PA	HP
xx00P1	PL
xx00QA	HQ
xx00QB	Q C ...... DD
xx00QC	R . ... CD
xx00Q1	Q . ...
xx00Q2	Q . ... D
xx00Q3	LCQ
xx00Q4	LVQ
xx00Q5	LJQ
xx00Q6	R . ...
xx00Q7	R L

PROGRAM CODE	ASSOCIATED CHOICE - COMMENTS
xx00Q8	R C
xx00Q9	LCR
xx00SA	HS
xx00SI	S ....*... U
xx00S1	S ....*...
xx00S3	S ....*... V
xx00S4	S ....*... C
xx00S5	S ....*... LC
xx00S6	S ....*... G
xx00S7	LSS
xx00S8	LNS LCS
xx00S9	S ....*... LV
xx00TA	HT
xx00TT	TUP
xx00TU	TRA
xx00TV	TLA
xx00TW	TPH
xx00TX	TUG
xx00TY	TUS
xx00TZ	TOP
xx00T1	TST
xx00T2	TSU
xx00T3	TGR
xx00T4	TPR
xx00T5	TRE
xx00T6	TTY
xx00T7	TUD
xx00T8	TVE
xx00T9	TAT
xx00UD	Word-processor Upload/Download

Size : about 6 mega-bytes.

2.3. BATCH PROGRAMS

BATCH PROGRAMS

CODE	PROC	COMMENTS
DAFD10	DPDF	DAF pre-processor
PDTPDF		DAF extraction sub-program
PDCHOI	DUPT	Choice decoding sub-program
PDSA10	DPRT	DPRT print sub-program
PDSB	-	DPRT flow monitor (French)
PDSBE	-	DPRT flow monitor (English)
PDSBAS	DSAV	Checks data integrity
PDSCAM	DUPT	Suppl. check sub-program for PDSUB1
PDSCEV	-	Suppl. check sub-program for PDSUE1
PDSCMQ	-	Suppl. check sub-program for PDSUQ6
PDSCRQ	-	Suppl. check sub-program for PDSUQ1
PDSCSI	-	Suppl. check sub-program for PDSUS1
PDSDAC	JCLDAF	Access sub-program for DAF
PDSBDF	-	Extraction sub-program for DAF
PDSEX	DEXT	DEXT flow monitor (french)
PDSEXE	DEXT	DEXT flow monitor (english)
PDSE90	DPRT	DPRT print sub-program
PDSFAC	-	File-access sub-program
PDSINI	DINI	Initializes DSMS files
PDSJMS	DREN	Changes the codes in the Journal
PDSLVB	DLVB	Replaces low-values by blanks in the BB
	-	backup file
PDSMS	DREN	Replacement monitor for table codes,
	-	keywords and site codes
PDSMSE	-	Same as PDSMS (english version)
PDSRCT	DREN	Input transaction check
PDSRFU	-	Sorts merges
PDSRMS	-	Changes the codes in the backup
PDSRQ0	DPRT	Analyzes requests
PDSRQ1	-	Selects and extracts requests
PDSRQ2	-	Formats elements
PDSRQ3	-	Extracts and prints the data
PDSR10	DREO	Reorganizes the cross-reference file
PDSR20	-	
PDSR30	-	
PDSR40	DREO	Reorganization
PDSUAA	DUPT	Sub-program

CODE	PROC	COMMENTS
PDSUB1	-	-
PDSUB2	-	-
PDSUB3	-	-
PDSUB4	DUPT	-
PDSUE1	-	-
PDSUE2	-	-
PDSUE3	-	Sub-program
PDSUK1	-	-
PDSUP0	-	Batch update monitor
PDSUP1	-	Sub-program
PDSUQ1	-	-
PDSUQ2	-	-
PDSUQ5	-	-
PDSUQ6	-	-
PDSUQ7	-	-
PDSUQ8	-	-
PDSUSI	-	-
PDSUS1	-	-
PDSUS3	-	-
PDSUS4	-	-
PDSUS6	-	-
PDSUTT	-	-
PDSUTV	-	-
PDSUTW	-	-
PDSUTX	-	-
PDSUTY	-	-
PDSUTZ	-	-
PDSUT1	-	-
PDSUT2	-	-
PDSUT3	-	-
PDSUT4	-	-
PDSUT5	-	-
PDSUT6	-	-
PDSUT7	-	-
PDSUT8	-	-
PDSUT9	-	-
PDSXCT	DEXT	Checks validity of input
PDSXDT	DINS	Lists the installed programs
PDSXST	DEXT	Sorts entities
PDSXTH	DEXH	Extracts tables to create external lists
PDSXTR	-	Extracts entities
PDS300	DARC	Archives and deactivates the journal
PDS320	-	Reinitializes the journal

CODE	PROC	COMMENTS
PDS380	DRST	Verifies the journal
PDS400	-	Restores or initializes the files
PDS450	-	Retrieves the archived journal
PDS500	DSAV	Saves data/elements/references
PDS600	DEXP	Extracts VA Pac journal
PDS610	-	
PDS900	DUPD	Update on DSMS Database from DAF tables
REP2PJ	DEXQ	Extracts VA Pac journal < 2.0
PDS700	DXBJ	Extracts journalized transactions
PTU001	All	Copies entries onto a disk file
PDSR8B	DR80	Retrieval of a DSMS 8.0 / 8.01 database
PDSR8C		IMPORTANT: VA PAC MUST HAVE PREVIOUSLY
		BEEN UPDATED TO VERSION 8.0.2.
PDSR8X	DR8X	Retrieval of a DSMS 8.02 Database
		compatible with VA Pac 8.01.
		TO BE USED SIMULTANEOUSLY WHEN SWITCHING
		OVER FROM VA PAC 8.01 TO VA PAC 8.0.2.
PDSR8Q	DR8Q	Retrieval of a DSMS 8.0.2 v01/02
PDSR8R		Database (Retrieval of Queries)
PDSR15	DR15	Retrieval of a DSMS 1.2/1.5 Database
PDSR5J	DR5J	Retrieval of DSMS 1.5 archived journal

Size : about 6 mega-bytes.

2.4. SYSTEM FILES

SYSTEM FILES

System files are the files that make up the system itself and as such they are not affected by user operations and must be reloaded if the system has to be reinstalled.

The system files are:

- those described in the preceding subchapters, i.e.:

  . on-line programs

  (directory: $DSMSDIR/tp/gnt)

  . batch programs

  (directory: $DSMSDIR/batch/gnt)

  . executable functions

  (directory: $DSMSDIR/bin)

- and the file containing the error messages and HELP documentation of the DSMS function (DE), whose characteristics are:

```
Size         : about 27,000 records (about 3.5 Mb)
Organization : indexed
Length       : 90
Key          : 17 (position 1)
Location     : $DSMSDIR/bases/ directory
Internal name: PACDDE
```

Furthermore:

. The DH file is used to backup the screen when calling for the Help function.

This file is found in the $DSMSDIR/bases/'db_name' directory.

NOTES

'$DSMSDIR' and 'db_name' (db = database) are installation parameters.

'$DSMSDIR' is the equivalent of $HOME/dsmsx and 'db_name' designates the Database name.

2.5. USER FILES

USER FILES

These files contain the data entered by the user and managed by DSMS.

The first five files contain data directly managed by DSMS. They are:

```
The DSMS data file (DA)

    .Organization : indexed
    .Length       : minimum 80, maximum 350
    .Key          : 40 (position 3)
    .Location     : $DSMSDIR/bases/'db_name' directory
    .Internal name: PACDDA


The cross-references file (DX)

    .Organization : indexed
    .Length       : 80
    .Key          : 50 (position 1)
    .Location     : $DSMSDIR/bases/'db_name' directory
    .Internal name: PACDDX


The VisualAge Pacbase elements file (DC)

    .Organization : indexed
    .Length       : minimum 50, maximum 168
    .Key          : 31 (position 3)
    .Location     : VA Pac indexed file directory
    .Internal name: PACDDC



The DSMS Journal file (DJ)

    .Organization : relative
    .Length       : 180
    .Location     : $DSMSDIR/journal/"db_name" directory
    .Internal name: PACDDJ


TP DAF work file (SYSDAF)

    . Organization : Indexed
    . Length       : mini 100, maxi 554
    . Key          : 37 (position 2)
    . Location     : $DSMSDIR/journal/'db_name' directory
    . Internal name: PACDDF
```

Three sequential files are used for the DSMS backup.
They are:

```
The Backup file (BB)

    .Organization : variable sequential
    .Length       : 354
    .Location     : $DSMSDIR/save/'db_name' directory
    .Internal name: PACDBB


The Journal Archive file (BJ)

    .Organization : sequential
    .Length       : 180
    .Location     : $DSMSDIR/save/'db_name' directory
    .Internal name: PACDBJ


The Deactivated Archive file (BQ)

    .Organization : sequential
    .Internal name: PACDBQ
    .Location     : None by default (non-assigned file)
                    Assign in $DSMSDIR/save/'db_name'
                    if this file is used, refer to
                    Subchapter'DE-ACTIVATION OF ARCHIVED
                    TRANSACTIONS' in the description of
                    the DARC procedure.
```

NOTE

$DSMSDIR and 'db_name' (db = database) are installation parameters.

'$DSMSDIR' is the equivalent of $HOME/dsmsx and 'db_name' designates the name of the Database.

The BB and BJ sequential backup files are used as input and output by a number of batch procedures. In this case, they are created and used under two different names: Bx as input file and Bx.NEW as output file (BB and BB.NEW for instance). At the end of the procedure (if there is no error) the BxBACKUP.ini file is called. By default, it rotates the processed backup on two copies: Bx is renamed Bx-1, and Bx.NEW is renamed Bx before being deleted.

2.6. USER CONTROL SUB-PROGRAMS

USER CONTROL SUB-PROGRAMS

These sub-programs enable the user to modify the controls on DSMS accesses.

CODE	MEANING
D8CUAM	On-line control on Change definition
D8CUEV	On-line control on Event definition
D8CUMQ	On-line control on Report definition
D8CURQ	On-line control on Query definition
D8CUSI	On-line control on Site definition
PDSCAM	Batch control on Change definition
PDSCEV	Batch control on Event definition
PDSCMQ	Batch control on Report definition
PDSCRQ	Batch control on Query definition
PDSCSI	Batch control on Site definition

These programs are delivered under two formats:

. '.gnt' compiled in DSMS UNIX environment, which can be used as they are with the same Micro Focus Run Time. They are located under the $DSMSDIR/tp/gnt directory.

. COBOL SOURCE files (.cbl), which can be compiled and used in a Micro Focus environment different from that of DSMS. They are located under the COBOL directory on the installation CD-ROM.

These sub-programs can be used in batch or transactional user applications.

3. ENVIRONMENT

3.1. INTRODUCTION

INTRODUCTION

DSMS operates a DSMS database on a Local Area Network composed of a server running under UNIX on one side and of 'dumb terminal'-type workstations and personal computers running under Windows on the other side. The advantages of personal computing are then combined with the functionalities of a host-based DSMS System.

This chapter describes the environment and resources necessary to operate the system.
It details the DSMS elements (on-line/batch servers, file server), their components, their structure, and their operating mode.

NOTE:

The Microfocus COBOL run-time is required for the operation of the DSMS system.

3.2. ARCHITECTURES

ARCHITECTURES

To access a DSMS database, each workstation communicates with one on-line server. An on-line server can theoretically manage up to 99 workstations. However, for performance considerations, it is advisable to limit this number to 50.

UNIX computers can support several on-line servers running simultaneously.

To execute generation-print requests submitted on-line, each workstation communicates with one or more batch servers. The number of workstations that can be managed by batch servers is not limited.

Resource sharing

The following diagram illustrates the way a local area shares programs, files and connections.

```
+--------+--------+--------+--------+
!        ! ON-LINE!        !        !  +---------+---------+
!        ! /BATCH !        !        !  !         !         !
! BATCH  !  COMM. ! ON-LINE! DSWST  !  !  PAW    ! PACLINK !
! SERVER +--------+ SERVER !  WORK- !  !         !         !
!        !  DSMS  !        ! STATION!  !         !         !
!        !DATABASE!        !        !  !         !         !
+--------+--------+--------+--------+  +---------+---------+
!             UNIX SYSTEM           !  !      WINDOWS      !
+-------------+------------+--------+  +-------------------+
!        ON-LINE/WORKSTATION        !  ! W-STATION/ON-LINE !
!            COMMUNICATION          !  !   COMMUNICATION   !
+-------------+------------+--------+  +-------------------+
!          NETWORK INTERFACE        !  ! NETWORK INTERFACE !
+-------------------+---------------+  +-------------------+
                    !                           !
                    !                           !
--------------------+---------------------------+----------
                       LOCAL AREA NETWORK (LAN)
```

OPERATING PRINCIPLE

Communication must be established with the on-line server when using workstations.

The communication mode is a 'process-to-process dialog', i.e data exchanges are carried out via the socket interface of the TCP/IP network protocol.

Each time a UNIX workstation sends a generation request, the on-line server communicates with the workstation in its usual way, and forwards the generation-print requests to the batch server, via a pair of communication files. The files ensuring communication between the on-line server and the batch server do not need to be shared on the local network.

3.3. WORKSTATIONS

WORKSTATIONS

The DSMS environment is made of workstations operating Windows from release 3.1, including Windows 95 and Windows NT, and connected to the Local Area Network as well as workstations operating under the server (DSWSTSOCK) UNIX system.

Each workstation communicates with a DSMS on-line server. Whenever a workstation sends a request, the on-line server executes the corresponding program, accesses the DSMS database and if necessary, transmits the printing requests to the batch server.

For Windows workstations, a TCP/IP software, compatible with 'Windows socket' ensure communication with the on-line server.

Using UNIX workstations (DSWSTSOCK) implies the opening of a UNIX session on the machine which hosts the on-line server.

DSMS output report files can be accessed either via the sharing of the server USERS directory or via a file transfer software (FTP for example).

3.4. ON-LINE SERVER

ON-LINE SERVERS

Each on-line server is in charge of executing transactional programs allowing the handling of the DSMS Database.

The dialog is carried out in a synchronous way without using the communication files for the sharing of information.

In order to set up a socket connection, the workstations need to know the name of the UNIX machine hosting the on-line server, and the socket number assigned to the on-line server at the time of creation.

3.5. BATCH SERVER

THE BATCH SERVERS

The batch server processes generation-print requests sent from the DSMS LVQ screen. The execution of these requests is performed by a batch monitor which automatically runs the DPRT procedure.

The batch server communicates with on-line server(s) via files defined in the batch server configuration file, and usually named the "db_name".dlb and "db_name".dbd files.

The batch server display mode (view option of dsbatch) gives the list of the requests already processed and of those which are being processed. Each request is serialized and processed sequentially by the batch monitor.

More than one batch server can be installed to process generation-print requests of a DSMS Database. In this case communication between batch servers with different names and the on-line server(s) is still carried out by the DLB and BDD files. Generation-print requests are shared by the batch servers.

For more information, see Chapter MONITOR START-UP, Subchapter 'Batch Monitor Start-up'.

BATCH OPERATION

The operation principles are similar in batch mode and on-line mode, i.e. based on the exchange of messages between batch and on-line servers.

The DLB file contains the list of requests submitted from the LVQ screen of a workstation. The batch server reads the DLB file in cycles.

For each request, the DLB file contains the following:

- date and time of submission

- job number,

- user code,

- database code,

- beginning address of the request in the DBD file,

- end address of the request in the DBD file.

DLB points to the DBD file, which contains the generation-print requests.

DPRT outputs a file for each type of generation-print requests. These files are created in the user directory (see Chapter INSTALLATION, subchapter 'Description of Created Directories').

4. MONITORS START-UP

4.1. TP : ON-LINE MONITOR START-UP

ON-LINE MONITOR START-UP

The on-line monitor, whose executable programs (dstp, dsserver, and dslaunch) are located in the $DSMSDIR/bin directory, has the following purposes:

- activate or de-activate the on-line server(s),

- provide information on the on-line server(s),

- purge the workstation(s) connected to an on-line server,

- purge the on-line server(s).

These operations are performed via the execution of the on-line monitor command interpreter (dstp).

The interpreter can be run in two different modes:

-the command mode, in which you type:

```
dstp <command>
```

-the 'SHELL mode', in which you type :

```
dstp -s
```

The main benefit of the command mode is that it allows the insertion of the on-line monitor commands in a command file.

For example, the com_dsms file contains the following commands:

```
# display of the servers' status
dstp info
# start-up of the tp1 on-line server
dstp start tp1
# 10-second display of information about the tp1 server
dstp info tp1
sleep 10
# shutdown of the tp2 on-line server
dstp stop tp2
# submission of the purge command for workstation 003
connected to server tp1
dstp purge tp1 003
```

The main benefit of the SHELL mode is that it avoids running the interpreter for each command (the interpreter remains pending for the next command).

The following commands are available:

-debug	activation/de-activation of the debud mode
-exit	exit shell mode
-help	help on the specified command
-info	information on the specified server
-purge	purge of workstation specified by number
-purge_server	purge of on-line server
-shutdown	server shutdown without confirmation
-start	start-up of specified server
-stop	server shutdown with confirmation
-trace	activation/de-activation of trace
-view	report display for indicated server

A detailed description of the commands is provided in Subchapter 'Description of Commands'.

EXECUTION CONDITION

The DSMSDIR environment variable must be initialized. See Chapter INSTALLATION, Subchapter 'DSMS Installation'.

The COBPATH environment variable must include the access path to the on-line functions: $DSMSDIR/tp/gnt.

ON-LINE SERVER CONFIGURATION

Upon being started, each server executes a command file named 'server-name'.ini, which will assign all the environment variables needed by the server.

This file is created when the server is created in the $DSMSDIR/assign/tp_server directory.

. On-line server environment variables:

```
  - BASENAME         : Database name
  - RADICAL          : COBOL programs' prefix
  - PAC7LB and PAC7BD: Batch communication files
  - STATIONS         : Maximum number of workstations
  - PACPAUSE         : Workstation polling periodicity
                       in milliseconds
  - PACSOCKET        : Port number (socket interface)
```

. Database environment variables:

```
  - PACDDA           : Data file
```

- PACDDE	: Error message file
- PACDDX	: Cross-references file
- PACDDJ	: Journal file
- PACDDC	: VA Pac elements file
- PACDHE	: Help function work file
- SEMLOCK	: Concurrent update serialization lock

CAUTION

For all creations or deletions of on-line servers, USE THE DSMSINSTALL PROCEDURE.

DESCRIPTION OF COMMANDS

debug

This command activates or de-activates the debugging mode on the on-line server operation. The server's name must be a parameter, followed by 'on' for activation or 'off' for de-activation.

It updates an output file named server_name_process_number.SPY and located in the $DSMSDIR/log directory.

Example:

Starting the debug mode on an on-line server named tp1:

```
dstp debug tp1 on (command mode)
DSMS : debug tp1 on (shell mode)
```

Example:

Stopping the debug mode on the tp1 on-line server:

```
dstp debug tp1 off (command mode)
DSMS : debug tp1 off (shell mode)
```

exit

With this command, you can exit the 'SHELL' mode (command interpreter) previously activated via the dstp -s command.

help

This command allows you to display help on an administration command of the on-line server. If no parameters are specified, the list of all available commands is displayed. If a specific command is indicated, help on this command is displayed.

Example: Display required for the 'start' command syntax

```
dstp help start (command mode)
DSMS : help start (shell mode)
```

info

This command allows you to display information on the on-line server(s). Followed by the server's name, it displays information about the server's configuration files located in

```
      $DSMSDIR/assign/tp_server/'server_name'.cfg
      $DSMSDIR/assign/tp_server/'server_name'.ini
and   $DSMSDIR/assign/'db_name'/'xxxxxx'.ini
```

Used on its own, it displays the following information:

- active (if the server is active)

- not active (if the server is not active)

- not configured (the server is declared in

```
      $DSMSDIR/assign/tp_server/dsmsx.srs
but does not have a configuration file in
      $DSMSDIR/assign/tp_server/'server_name'.ini)
```

- Error (if the server is shut down normally).

purge

This command allows you to purge a workstation (in other words, to interrupt a workstation's connection).

If the dstp interpreter is in command mode, the syntax of the purge command is as follows:

```
'dstp purge <server_name> <workstation number>'
```

In 'shell' mode, the purge command can have two syntaxes:

- 'purge <workstation number>'

  if the prompt is not 'DSMS :'.
  (The prompt takes the server's name as value when, for instance, the 'view' command has been used.)

- 'purge <server_name> <workstation number>'

  if the prompt is 'DSMS :'.

purge_server

This command allows you to purge an on-line server in case of abnormal execution.

In command mode, this command has the following syntax:

```
dstp purge_server <server_name>
```

In shell mode, it has the following syntax:

```
purge_server
```

NOTE:

This command deletes the communication files in the $DSMSDIR/commun and $DSMSDIR/tp/save directories, as well as the message queue (IPC tables) associated with the server.

shutdown

This command allows to stop the on-line server. The server's name must be entered as a parameter of the command. A conversations backup file is created in the $DSMSDIR/tp/save, and whose name is the on-line server's name with the .ts extension.

Example: shutdown of the on-line server called tp1:

```
pactp shutdown tp1   (command mode)
DSMS : shutdown tp1  (shell mode)
```

start

This command allows you to activate an on-line server.

It must be followed by a server name.

The initialization file corresponding to the server (on-line server name + .ini extension) must exist in the $DSMSDIR/assign/tp_server directory.

The server start-up program, dslaunch, starts and activates the dsserver process.

If a problem occurs (on-line server jam), it is possible to kill the pacserver process using the Kill -15 or Kill -9 commands and then to destroy the server message queue using either the ipcs or ipcrm -q command.

Example: activation of the on-line server called tp1:

```
dstp start tp1       (command mode)
DSMS : start tp1  (shell mode)
```

stop

This command shuts down an on-line server. The server name must be entered as one of its parameters.

A conversation backup file is created in the $DSMSDIR/tp/save directory. Its name is the on-line server name and its extension is .ts. You will be asked to confirm the interruption of the server.

Example: shutdown of the on-line server called tp1.

```
dstp stop tp1         (command mode)
DSMS : stop tp1       (shell mode)
```

trace

This command allows you to activate or de-activate the trace mode on an on-line server.

The trace displays the response time of the on-line server.

The server's name must be entered as a parameter, followed by 'on' or 'off' to activate or de-activate.

A file containing the trace result is updated when the server stops. This file is called: 'server_name_process_number.LOG and is located in the $DSMSDIR/log directory.

Example 1: activating the trace on the tp1 on-line server:

```
dstp trace tp1 on          (command mode)
DSMS : trace tp1 on        (shell mode)
```

Example 2: de-activating the trace on the tp1 server:

```
dstp trace tp1 off         (command mode)
DSMS : trace tp1 off       (shell mode)
```

view

This command allows you to display information about an on-line server. The list of connected workstations, the user's name, the name of the on-line program executed or being executed, this program's start-up time and execution time (in milliseconds) are displayed.

Example: display of information about on-line server tp1:

```
dstp view tp1          (command mode)
DSMS : view tp1        (shell mode)
```

4.2. BAT : BATCH MONITOR START-UP

BATCH MONITOR START-UP

The batch monitor, whose executable programs (dsbat, dsbatsrv) are located in the $DSMSDIR/bin directory, has the following purposes:

- To activate or deactivate the batch server(s)

- To provide information on the batch server(s)

- To purge the batch server(s)

These operations are performed via the execution of the batch monitor command interpreter (the dsbatch executable program).

The interpreter can be run in two different modes:

. The COMMAND mode, in which you type:

```
dsbatch <command>
```

. Or the SHELL mode, in which you type :

```
dsbatch -s
```

The main benefit of the COMMAND mode is that it allows for the insertion of the batch monitor commands in a command file.

Example: The com_dsms file contains the following commands:

```
# display of the servers' status
dsbatch info
# launch of the bat1 batch server
dsbatch start bat1
# 10-second display of information about the bat1 server
dsbatch info bat1
sleep 10
# shutdown of the bat2 batch server
dsbatch stop bat2
```

The main benefit of the SHELL mode is that it avoids running the interpreter for each command (the interpreter remains pending for the next command).

The following commands are available:

-exit	Exit shell mode
-help	Help on command syntax
-info	Information on the specified server(s)
-purge_server	Purge of batch server
-shutdown	Server shutdown without confirmation
-start	Server start-up
-stop	Server shutdown with confirmation
-view	Display of report for indicated server

A detailed description of the commands is provided in the paragraph 'Description of commands'.

Batch modules executed by the server are the same as those of procedures (.gnt files in the $DSMSDIR/batch/gnt directory).

EXECUTION CONDITIONS

The DSMSDIR environment variable must be initialized. See Chapter INSTALLATION.

The COBPATH environment variable must include the access path to on-line functions: $DSMSDIR/batch/gnt.

REQUESTS OUTPUT

When a workstation submits a printing request from the LVQ screen, the screen which is sent back displays:

```
      JOB STREAM BUILT - NUMBER: nnnnn
```

The files created by the DPRT procedure are inserted in the 'user code' sub-directory under the 'users' directory.

Their code is composed of the request number followed by the generated type of file (for more details on this coding, see the description of the DPRT procedure.

```
Example: under $DSMSDIR/users/John, one will find  00055.da
                                                   00055.db
                                                   00055.dq
                                                   etc...
```

Only the 'useful' files are saved: printing execution report (DA), description of the requests (DB) and printing (DQ).

OUTPUT FILES PROCESSING

The PLBTAGP variable defined in the server configuration file allows to call the DSMSAGP command file after the print-generations.

The call of this file allows to automate some tasks whose nature varies with the environment.

An example of PACAGP file is provided at installation under the $DSMSDIR/batch/proc directory. This file can of course be modified in order to take into account the specifics of each DSMS site.

In the supplied example, some comments explain in particular parameters passed onto this command file by the batch server and which are thus reusable for the processing of the output files.

The DSMSAGP file is located by default under the SMSDIR/batch/proc directory.

CONFIGURATION OF THE BATCH SERVER

Upon being started, each server executes a command file named 'server-name'.ini, which will assign all the environment variables the server needs.

This file is created when you create the server itself, and copied in the $DSMSDIR/assign/batch_server directory.

. Batch server environment variables:

```
  - BASENAME        : Database name
  - PAC7LB, PAC7BD  : Batch communication files
  - PLBTPAR         : List of assignations made
                      by the batch server
  - SYSOUT          : User directory
  - PLBTDEL         : Temporary file deletion option
                      YES=files are deleted
                      NO =files are kept
  - PLBTLG          : Language option for the reports
                      printed by the GPRT procedure.
                      A=English (default)
                      F=French
  - PTLBMON         : Code of monitor linking the sequence
                      of the DPRT generation-print procedure
                      PDSBE=English; PASB=French
  - PLBTAGP         : PACAGP command file execution option
                      YES=PACAGP is called
                      NO=PACAGP is not called
```

. Database file environment variables

```
  - PACDDA          : Data file
  - PACDDE          : Error message file
  - PACDDX          : Cross-reference file
  - PACDDC          : VA Pac element file
```

CAUTION

For all creations and deletions of batch servers, USE THE DSMSINSTALL PROCEDURE.

DESCRIPTION OF COMMANDS

exit

With this command, you can exit the 'shell' mode (command interpreter) previously activated by the dsbatch -s command.

help

This command allows to display help on an administration command of the batch server. If no parameters are specified the list of all the available commands is displayed. If a specific command is indicated, help on this command is displayed.

Example: Display required for the 'start' command syntax

```
      start

      dsbatch help start      (command mode)
      DSMS : help start       (shell mode)
```

info

This command allows to display information on the batch server(s).

Followed by the server's name (info 'server_name'), it displays information about the server's configuration files located in:

```
$DSMSDIR/assign/batch_server/'server_name'.ini
$DSMSDIR/assign/'db_name'/'xxxxxx'.ini
```

Used on its own, it displays the following information:

```
- active (if the server is active)

- not active (if the server is not active)

- not configured (the server is declared in
  $DSMSDIR/assign/batch_server/dsmsx.srs,
  but does not have a configuration file in
  $DSMSDIR/assign/tp_server/'server_name'.ini),

- Error (abnormal shutdown of the server).
```

<u>purge_server</u>

This command allows to purge a batch server in case of abnormal execution. In command mode, this command has the following syntax:

'dsbatch purge_server <server_name>'.

In shell mode, it has the following syntax:

'purge_server <server_name>'.

N.B.: This command deletes the message queue (IPC Tables) associated with the server.

<u>shutdown</u>

This command allows to stop the batch server without confirmation. The server's name must be entered as a parameter of the command.

Example: shutdown of the batch server called bat1

```
dsbatch shutdown bat1   (command mode)
DSMS : shutdown bat1    (shell mode)
```

start

This command allows to activate a batch server. The server's name must entered as a parameter of the command. The initialization file corresponding to the server (batch server's name + .ini extension) must exist in SMSDIR/assign/batch_server directory.

The server start-up program, dsbatch, starts and activates the dsbatsrv process. If a problem occurs (batch server jam) it is possible to kill the dsbatsrv process using the kill -15, or kill -9 commands and then destroy the server message queue using either the ipcs -q and ipcrm -q commands.

Example: activation of the batch server called bat1

```
dsbatch start bat1    (command mode)
DSMS : start bat1     (shell mode)
```

stop

The batch server's name must be entered as a parameter of this command

Example: shutdown of the batch server called bat1

```
dsbatch stop bat1   (command mode)
DSMS : stop bat1    (shell mode)
```

view

This command allows to display information about a batch server. For each job submitted to the server, the following information is displayed:

- line number,
- job number,
- DSMS user code,
- workstation number,
- date and time of job submission,
- beginning time of the request processing,
- ending time of the request processing.

4.3. DSWSTSOCK : UNIX DUMB TERMINAL START-UP

UNIX DUMB TERMINAL START-UP (DSWSTSOCK)

The interactive dialog between the workstation's user and the on-line server is started up via the following command:

```
dswstsock <-trace>
```

The on-line server to which the workstation is connected is determined by the workstation environment variables:

'PAC_HOST_SERVER_NAME' which contains the name of the UNIX terminal from which the server is started up.

'PAC_HOST_SERVER_SOCKET' which contains the number of the port used by the server.

In other words, the workstation will connect itself to the server, using the 'PAC_HOST_SERVER_SOCKET' communication port on the machine named 'PAC_HOST_SERVER_NAME'.

The 'dswstsock' command is found in the $DSMSDIR/bin directory.

'TRACE' OPTION

The -trace option allows to activate the trace. The file provided by this option has the following form:

> wstyymmdd.LOG (yy = year, mm = month, dd = day). It is located in the $DSMSDIR/Log.

END OF CONVERSATION

End of transaction: when the user quits the transaction (F6 or CH: FT), the initial screen is displayed. To clear this screen, press the F12 key, or its equivalent (see paragraph ' Terminal Configuration', in this subchapter).

ENVIRONMENT

The environment variables configuration file of 'dswstsock' is called 'dswstsock.ini'. This file is created during installation in the $DSMSDIR/assign/monitors directory.

EXECUTION CONDITION

The on-line server must be started up.

CONFIGURATION DESCRIPTION

File: dswstsock.ini

```
- FPARAM           = terminal configuration file
- PAC7CN           = conversation backup file
  TRACE            = trace activation option
                    (YES = activation, NO = no activation)
- PAC_HOST_SERVER_NAME: name of UNIX terminal on which the
                        on-line server is installed.
- PAC_HOST_SERVER_SOCKET: number of the on-line server
                          Communication port
```

Terminal configuration

The dsparam configuration file located in the $DSMSDIR/assign/monitors directory contains the description of the screen's and keyboard's parameters. It is divided in three parts:

- A list of codes for the display configuration (screen attributes)
- A list of parameters that describe the behavior of the character input function
- A table of correspondence between the terminal's controls and the functions of dswstsock.

Display parameters

There are 11 display parameters: 2 color parameters (character color and background color), and 9 screen attribute management parameters.

The two colors available on a monochrome terminal are black (B) and white (W). Black is the character color, white is the background color.

There are 4 possible values for attributes:

. normal mode	N
. semi-intensity mode	D
. double-intensity or bright mode	B
. reverse mode	R

The 9 screen attributes fill the following fields:

```
. protected and normal fields
. protected and bright fields
. normal input fields
. bright input fields
. current field
. normal underlined field
. bright underlined field
. normal blinking fields
. bright blinking fields
```

Parameters for the input function

There are 5 parameters for the input function. They can have the Y or N value:

. The first parameter is not used in this sub-release;

. The second parameter, if set to Y, specifies that the cursor will move to the next field when the current field is filled;

. The third parameter, if set to Y, allows for a repositionning of the cursor on the first field of the screen after the last field;

. The fourth parameter, if set to Y, allows for the deletion of the characters at the end of the current field (in insertion mode);

. The last parameter is not used in this sub-release.

Table of correspondence for keys

The last part of the dsparam file describes the correspondences between some key sequences and the dswstsock functions. In the current sub-release, only the control keys are available. They are described by:

```
ctrl_A, ctrl_B, ..., ctrl_Z.
```

Available functions are:

. Enter	transmit via the Enter key
. Clear	clears the screen
. PF1..PF24	function keys 1 to 24
. Tab	goes to the next input field
. BackTab	goes back to the previous input field
. NL	new line
. Home	goes to the first input field
. End	goes to the last input field
. BeginF	moves cursor to beginning of field
. EndF	moves cursor to end of field
. Curs-U	moves cursor up
. Curs-D	moves cursor down
. Curs-L	moves cursor left
. Curs-R	moves cursor right
. BSpace	deletes character before cursor
. Insert	insertion mode toggle
. Delete	deletes character beneath cursor
. DelEOF	deletes end of current field
. DelINP	deletes whole current field contents
. Recover	re-displays the screen

In the default configuration files, some of the control keys are already assigned:

ctrl_H	BSpace
ctrl_I	Tab
ctrl_J	NL
ctrl_M	Enter

Others, such as ctrl-C or ctrl-Z must be used with care because they may conflict with the terminal's configuration (which can be modified via the stty command).

IMPORTANT NOTE:

> The assignment of function keys is determined by the terminfo configuration file, which depends on the operating system. One should therefore read this file carefully before starting any key assignment.

4.4. PACLINK : PACLINK-EMULATED WORKSTATION START-UP

PACLINK-EMULATED WORKSTATION START-UP (PACLINK)

Under MS-Windows or Win OS/2, double-click on the icon created at the time of installation .

REMINDER:

1. The user must be connected to the Local Area Network in order to access the UNIX terminal (communication via the socket interface).

2. The UNIX terminal and the associated communication port are assigned in the GSTCPIP.PRM file

   The user must update the corresponding file in order to adapt the configuration of the workstation to that of the on-line server to which he/she wishes to connect.

   The communication port associated with the on-line server is created during the creation of the on-line server on the UNIX terminal.

NOTE ON THE OPERATION OF THE PACLINK-EMULATED WORKSTATION

The Paclink-emulated workstation (under Windows) uses the GSTCPIP and PACLINK programs.

Refer to the 'COMMUNICATIONS MANAGER AND PACLINK UTILITY' Manual for a description of the workstation's specific features.

4.5. P.A.W : REWAMPED WORKSTATION START-UP

P.A.W: REWAMPED WORKSTATION START-UP

To start up the user workstation, go to the 'CHEMIN-EXE' directory created during installation and run the PAW.EXE program.

For more information on this directory, refer to the P.A.W DEVELOPER'S GUIDE, Chapter 'REVAMPING OF IBM PRODUCTS'.

REMINDER:

1. The computer must be connected to the Local Area Network in order to access the files that ensure the communication with the DSMS on-line server.

2. Communication files are assigned in a file whose name is specified in the ENVIRON.PRM file. The user updates this file if necessary to adapt the file assignment to the installation features and to the DSMS on-line server to which he/she wishes to connect.

   The DSMS on-line server creates the communication files in the $DSMSDIR/Commun directory of the DSMS file server. On the P.A.W workstations, these files are coded as follows: name of the on-line server + the .DCM or .DCD extensions.

```
Example: U:\DSTP1.DCM
         U:\DSTP1.DCD

with:    U = common disk unit
         DSTP1 = name of the on-line server defined
                 during the start-up of this server
```

NOTE ON THE OPERATION OF THE DSMS WORKSTATION

The workstation uses the Pacbench Automatic Windowing (P.A.W) function.

For more information on the description of this workstation specific features, refer to the P.A.W User's Guide.

5. UTILITY PROGRAMS

5.1. CONVERSION AND SYSTEM UTILITIES

UTILITY PROGRAMS

Conversion utilities

```
cgiux2dos  : for the conversion of a UNIX-format file
             into a DOS-format file

cgidos2ux  : for the conversion of a DOS-format file
             into an UNIX-format file

cgitrans   : for the conversion of HP, iso8859, and
             pc850 characters
```

System utilities

```
dsdate     : specifies the dates of system files and
             programs
```

All utility programs are found in the $DSMSDIR/bin directory.

To know how they operate, type the following command:

```
<utility name> -h
```

Example :

```
cgitrans -h
```

6. BATCH PROCEDURES

6.1. INTRODUCTION

INTRODUCTION

Batch processing with DSMS is divided into various procedures. The following chapters describe each of these procedures that may be used and give details on their specific execution conditions.

For each procedure, there is:

. A general presentation containing:

- an introduction,

- the execution condition(s),

- the action(s) to be taken in case of abnormal execution,

. The description of user input, processing, results, and possible recommendations on use.

. A description of each step containing:

- The files used (temporary and permanent),

- The return codes that may be generated by each step.

6.2. CLASSIFICATION OF PROCEDURES

CLASSIFICATION OF PROCEDURES

There are various types of batch procedures.

DATABASE MANAGEMENT PROCEDURES:

. Initialization of DSMS files (DINI)

. Archiving of file update transactions (DARC)

. Restoration of files using the backup and archived files (DRST)

. Backup of files (DSAV)

. Reorganization of cross-references files (DREO).

UTILITY PROCEDURES:

. Extraction, from the VA Pac Journal, of transactions corresponding to modified VA Pac entities that relate to changes (DEXP).

. Extraction, from the DSMS journal (DXBJ), of transactions for update by the DUPT batch procedure.

. Printing of query results, and of table and keyword lists requests (DPRT).

. Extraction from DSMS of Events, Changes, Sites or Tables as batch transactions (DEXT).

. Extraction of DSMS tables to create lists of external values for the revamped version of the developer's workstation (DEXH).

. Batch update of DSMS files of Events, Changes, Sites or Tables (DUPT, DUPD).

. Pre-processing of DAF source files (DPDF).

. Renaming of Table, Site and Keyword codes (DREN).

RETRIEVAL OF PREVIOUS RELEASES

For sites where DSMS monitors control VA Pac Databases, the installation of DSMS 2.5 requires version 8.0.2, or higher, of VA Pac.

PREVIOUS RELEASE RETRIEVAL PROCEDURES:

. Retrieval of DSMS 8.0.1 Database (DR80).

. Retrieval of DSMS 8.0.2 Database compatible with VA Pac 8.0.1 (DR8X) to be used when switching over from VA Pac 8.0.1 to 8.0.2.

. Retrieval of DSMS 8.0.2 v01 or v02 Database (DR8Q) (retrieval of queries).

. Retrieval of DSMS 1.2 or 1.5 Database (DR15)

. Retrieval of DSMS 1.5 archived journal (DR5J)

RETRIEVAL OF A DATABASE FOR ANOTHER PLATFORM:

. Replacement of low-values by blanks (DLVB).

6.3. STRUCTURE OF PROCEDURE COMMAND FILES

STRUCTURE OF PROCEDURE COMMAND FILES

Batch procedures command files are created in the $DSMSDIR/batch/proc directory during the installation.

ADAPTATION OF PROCEDURES TO THE SITE'S REQUIREMENTS

The DSMS Database Manager must sometimes modify the command files of the batch procedures. For instance, if he/she wants to split the Database's files in two disks, or move the DE file, the modifications induced in the command files may be significant.

This is why the DSMS procedures (batch or server start-up) are designed to facilitate any modification on the standard installation, and to reduce the adaptation of the procedures linked to the environment requirements.

The purpose of this subchapter is to analyse a batch procedure so as to explain its operation and help the user in adapting it if necessary.

STRUCTURE OF A BATCH PROCEDURE

1. Parameters

- General:

Apart from DEXP, batch procedures require only one parameter: the Database name.

DEXP requires a second parameter: the name of the PJ file directory (backup of the VA Pac journal).

- Special (multi-user procedures):

Two more parameters are required for procedures which may be executed simultaneously by several users (DEXH, DEXP, DEXT, DPRT, DUPT, DXBJ):

. the first parameter is used as radical of all the files located in the 'input' and 'tmp' directories in order to differentiate the users.

. the second parameter is used as suffix or subdirectory of the 'input' and 'tmp' directories.

Both parameters may be set or not when submitting any batch procedure.

. If both are left with the blank value, file assignments in the 'input' and 'tmp' directories are not modified.

. The first parameter can be set and the second left with the blank value.

. The value of the first parameter cannot exceed two characters, so that file names can be read under DOS and OS/2.

. The second parameter cannot be set to a value if the first one is not.

. If the second parameter starts with the '/' character, it represents subdirectories of the 'input' and 'tmp' directories. Otherwise, it represents their extensions. In any case, the user is the one who must create the relevant directories.

Both parameters are used in the PACINPUT.ini and PACTMP.ini command files which are called by each procedure. They initialize the PACINPUT and PACTMP environment variables. (Refer to Subchapter 'PARAMETERS ASSIGNMENT AND CODING'

2. Parameter display and check

A procedure's execution starts with a check of parameters, via the call of the USAGE.ini command file:

```
       $DSMSDIR/batch/proc/USAGE.ini
```

This file is created during installation in the SMSDIR/batch/proc directory.

The USAGE.INI command file checks the parameters according to the procedure, and positions the PACRAD and PACSUF environment variables if relevant. (These variables are used in the PACTMP.ini and PACINPUT.ini command files. They represent the radical and extension parameters of the multi-user procedures.)

If it finds a discrepancy, USAGE.ini displays the corresponding error message, stops the procedure, and sends a return code '20'.

If it does not find any discrepancy, the execution carries on with the display of directory assignments. To view the display, at least when testing the installation, interrupt the execution by calling the command file:

```
       sh $DSMSDIR/batch/proc/MSGPAUSE.ini
```

The MSGPAUSE.ini file, created during installation in the $DSMSDIR/batch/proc directory, contains:

```
       echo ********** Check your parameters  *********
       echo Press Control_C to stop the execution
       echo Press Return to carry on
       read REPAUSE
```

If you do not wish to interrupt the execution, delete the line 'read REPAUSE' in MSGPAUSE.ini.

3. Contents of the user input

The procedure's command file then contains the description of the user input.

4. Parameters assignment and coding

For each step, the right files must be assigned.

-DATABASE FILES

These assignments are performed via the call of command files, created at installation in the following directory:

```
        $DSMSDIR/assign/'db_name'.
```

For instance, the assignment of the DE file is:

```
        $DSMSDIR/assign/$1/PACDDE.ini
```

The command files' main purpose is to put in one single place each Database file assignment. To modify the standard location of a file, only the assignment file needs to be adapted.

Note: the same files are used when starting up the servers.

-BACKUP FILES

As for Database files, the assignments are made through the call of some command files created at installation in the directory:

```
        $DSMSDIR/assign/'db_name'.
```

Example: assignment of the BB file:

```
        $DSMSDIR/assign/$1/PACSAVBB.ini
```

By default, the BB and BJ backup files are located in the directory:

```
        $DSMSDIR/save/'db_name'.
```

For all the batch procedures using backup files, there is a standardization on the files names:

```
        Input backup (read) = Bx

        Output backup (created by the procedure) = Bx.NEW
```

This way, file management is facilitated (for instance, see Subchapter 'Backup file management').

-TRANSACTION FILES

All the transaction files that are used as procedure input have the following name format: MBxxxx (where xxxx is the procedure's name).

All procedure output transaction files have the following name format: MVxxxx (where xxxx is the procedure's name). These files can contain, for instance, transactions generated by extraction procedures.

The transaction file location is determined by the PACINPUT environment variable, positioned in every procedure by a call of the PACINPUT.ini command file:

```
       $DSMSDIR/assign/$1/PACINPUT.ini
```

The PACINPUT.ini file is created during installation, or when a database is being created, in the directory:

```
       $DSMSDIR/assign/'db_name'
```

It contains:

```
# PACINPUT environment variable assignment script
# (directory: input)

# Description of parameters: $0      = procedure name
#                            $1      = database name
#                            $PACRAD = file radical
#                            $PACSUF = directory extension

PACINPUT=$DSMSDIR/input$PACSUF/$1/$PACRAD
export PACINPUT
```

Example of assignment in the DEXT procedure:

```
      PACDMB=$PACINPUT'MBEXTR'
      export $PACDMB

      PACDIM=$PACINPUT'MVDEXT'
      export $PACDIM
```

-OUTPUT REPORTS

All procedure output reports are stored in the temporary files directory and their name starts with the code of the procedure that produces them.

For details on report name coding, refer to Subchapter 'OUTPUT REPORTS'.

5. Error-free end of procedure

When no error is detected, the message 'End of the procedure' is displayed.

6. End of the procedure with error(s)

When an error is detected in a step, the next steps are not executed. The error message displays the program's name and, when possible, the type of error detected.

Finally, the procedure executes the ERRPAUSE.ini file, which stops the procedure and displays the error when there is a chain of procedures.

The ERRPAUSE.ini file, created at installation, contains:

```
echo 'Press RETURN to carry on'
read REPAUSE
```

7. Backup file management

For all procedures creating backup files, a command file is called at the end of the procedure showing no error. These command files are located in the $DSMSDIR/assign/'db_name' directory and are named BxBACKUP.ini (x = B or J). They are created at installation or creation, and contain (for instance PJBACKUP.ini, db_name = d400):

```
#  Journal backup file cycle shift script
   . $DSMSDIR/assign/d400/PACSAVBJ.ini
   if  -f '$PACSAVBJ'
   then
   mv  -f $DSMSDIR/save/d400/BJ  $DSMSDIR/save/d400/BJ-1
   fi
   mv  -f $PACSAVBJNEW  $PACSAVBJ
   mv  -f $DSMSDIR/save/d400/BJ.NEW  $DSMSDIR/save/d400/BJ
```

Characteristics of the BxBACKUP files:

. They use 'mv' to avoid time-consuming copies.

. They perform a shift of the last two versions of the backup files.

. They guarantee the Bx file is actually the last backup version (as Bx is automatically used as input of a procedure).

These files do not necessarily meet all the site operations requirements. The Database Manager generally needs to adapt them according to the characteristics specified below.

Use of the BxBACKUP files:

```
. BBBACKUP.ini : used in the DSAV, DREO, DINI procedures
                 and DR80, DR8X,DR8Q and DRBB retrievals.
. BJBACKUP.ini : used in the DARC procedure
```

6.4. RECOMMENDATION ON USE

RECOMMENDATION ON USE

The purpose of this subchapter is to make the Database Manager aware of the specifics of DSMS procedures that run under UNIX.

TEMPORARY FILES

For each procedure, refer to the corresponding chapter for a detailed description of these files. In any case, be sure to free enough disk space in the chosen user directory for a proper execution of the procedure.

TEMPORARY SORT FILES

When a program runs a sort, the called COBOL routines also use a temporary file that is independent of those mentioned above.

This temporary sort file is created in the /usr/tmp directory.

Its size may be 3 or 4 times that of the file to be sorted.

If the default directory is too small, use the TMPDIR variable to assign another directory to the temporary sort files, as follows:

```
files: TMPDIR=/tmp2
export TMPDIR.
```

Location of temporary files

The location of a temporary file is determined by the call of the PACTMP.ini command file, as follows:

```
   . $DSMSDIR/assign/$1/PACTMP.ini
```

This file is created at the time of installation in the directory:

```
   . $DSMSDIR/assign/'db_name'

# Temporary file assignation script

# Description of parameters: $0      = procedure name
#                            $1      = database name
#                            $PACRAD = file radical
#                            $PACSUF = directory extension

# Directory of temporary files: 'tmp'
  PACTMP=$DSMSDIR/tmp$PACSUF/$1/$PACRAD
  export PACTMP

# Directory of temporary sort files
# This directory must have enough capacity to hold
# three times  the size of the file to be sorted.
  TMPDIR=$DSMSDIR/tmp/$1
  export TMPDIR
```

The PACTMP.ini temporary file initializes the PACTMP environment variable, in order to assign the DSMS temporary files.

Example of assignment in the DEXT procedure:

```
  PACDW0=$PACTMP'W0'
  export $PACDW0
```

The PACTMP.ini file also initializes the TMPDIR environment variable in order to assign the temporary sort file directory.

CAUTION: GENERAL COMMENTS

1. Parameters must be supplied for each procedure. All the parameters needed when calling a procedure must be specified, even if they are not used.

2. When user input is allowed in a procedure, even if the input is optional, the corresponding transaction file must be present when the procedure is run.

3. No protection is guaranteed when a batch procedure that updates system or evolving files in the DSMS Database, is running while users are interactively performing updates on these same files. Only one person (the Database administrator) must be able to run batch procedures that update DSMS Databases. That person is therefore responsible for ensuring that the data in the Database is protected (by closing down the on-line servers, for example).

4. The temporary work files created by batch procedures are automatically deleted at the end of a procedure, except if a step does not run sucessfully and sends back a non-null return code.

6.5. SUBMISSION OF BATCH PROCEDURES

SUBMISSION OF BATCH PROCEDURES

Batch procedures must be submitted from the UNIX terminal used as the DSMS server.

6.6. OUTPUT REPORTS

OUTPUT REPORTS

REPORT CODING

All reports created by batch procedures are copied in the temporary files directory. They are coded on 6 characters plus an extension, as follows:

. The first 4 characters correspond to the code of the batch procedure (DARC in PROCDARC).

. The other two are the last 2 characters of the external name of the file (RU in PACDRU).

. The extension is made of the last 3 characters of the program which generated the file (300 in PDS300).

Example: DARC procedure

```
Program:    File  -->  Report code on disk
------------------------------------------

PDS300 :    PACDRU  -->  DARCRU.300

PDS320 :    PACDRU  -->  DARCRU.320
```

6.7. ABNORMAL EXECUTION

ABNORMAL EXECUTION

A batch program execution may abend.

For example, input-output errors on the system files or on the Database files cause the interruption of the current program and the display of the following message.

```
    PROGR : pppppp  INPUT-OUTPUT ERROR : FILE ff  OP : oo
    STATUS : nn
```

In most cases, examining the status and the type of operation allows you to find the cause of the abend.

The table below indicates standard values for the status and type of operation:

OO	OPERATION
W	WRITE
RW	REWRITE
RU	READ UP
OP	OPEN
CL	CLOSE
D	DELETE
R	READ
P	START
RN	READ NEXT

NN	STATUS
21	SEQUENCE ERROR
22	DUPLICATE KEY
23	NO RECORD FOUND
24	BOUNDARY VIOLATION
30	SYSTEM ERROR
34	BOUNDARY VIOLATION (SEQUENTIAL)
35	FILE NOT FOUND
92	LOGICAL ERROR (i.e. OPEN AN ALREADY OPENED FILE)
93	LOCKED FILE
95	INVALID/INCOMPLETE FILE DEFINITION

Some errors, other than input-output errors in a Database file, may also cause the following message to be displayed.

Run Time Error nnn

where 'nnn' is the error number.

Run Time Error 013 is the most common error. It means that the procedure did not find an input file.
In order to find out which file is missing, enter the SET command. This will display the list of allocated files.
You can also consult the procedure description in the corresponding chapter in this manual (Batch procedures).
Then, compare this list with the contents of the directories involved.
Most often, it is the input transactions file that is missing in the 'Release'\INPUT\'db_name' directory: MBxxxx, where xxxx is the procedure's code.

The following subchapter contains the list of the most frequent errors. Each Run Time Error is accompanied by a short explanatory message.

If a Run Time Error does not appear in the following list, or if the message is insufficient and the type of error signals a direct problem in the system programs, contact the IBM Technical Support and save all listings that could help in analyzing the problem.

ERROR CONTROL IN BATCH PROCEDURES

At the end of each batch procedure, the PAUSE instruction stops the execution with a non-null return code. This code can be retrieved in the $? variable just after the command submitting the procedure.

This prevents another procedure of being executed if several procedures are linked.

6.8. LIST OF 'RUN TIME ERRORS'

LIST OF 'RUN-TIME ERRORS'

This list presents the most common errors and their meaning.

Number	Meaning
004	Invalid file name
005	Invalid device specification
007	No more disk space
009	Directory full or does not exist
013	File not found
026	Block I-O error
027	Device not available
028	Disk space exhausted
033	Physical I-O error
105	Memory allocation error
116	Cannot allocate memory
135	File not found
150	Program abandoned on user request
157	Not enough program memory: object file too large to load
170	System program not found
173	Called program file not found
188	File name too long
198	Not enough program memory: object file too large to load
207	Machine does not exist on the network
208	Network communication error
209	Network communication error
221 !	
222 !>	Error during a sort
223 !	

7. JOURNAL ARCHIVING (DARC)

7.1. INTRODUCTION

DARC: INTRODUCTION

The Journal Archiving procedure (DARC) backs up the Journal file (DJ) as a sequential file (BJ), and reinitializes it both logically and physically.

The new archived transactions do not overwrite transactions previously archived; they are added to them.

The previously archived transactions can be deactivated, if requested.

EXECUTION CONDITION

The database must be closed to on-line use.

Even if the closing of on-line access is not controlled by the procedure, it prevents any other update while the procedure is being executed.

ABNORMAL EXECUTION

Refer to Subchapter 'Abnormal Execution' in Chapter THE BATCH PROCEDURES.

If the abnormal end occurs before the step which creates the Journal file (DJ), the procedure can be restarted as it is, after the problem has been solved.

If the abnormal end occurs during or after this step, the user input must be modified before a new execution of the procedure so as to specify a reinitialization request without a backup of the Journal file (already backed-up).

7.2. INPUT - PROCESSING - RESULTS

USER INPUT

The DARC procedure includes an optional input for:

. deactivating previously archived transactions that are now obsolete,

. indicating the absence of previously archived transac- tions during input,

. indicating the unavailability of the Data file (DA) during input,

. requesting a reinitialization of the transaction file only.

The structure of this input is as follows:

POS.	LEN.	VALUE	MEANING
2	1	'S'	Line code
3	4	nnnn	Session number
7	8	CCYYMMDD	OR date up to which the user requests deactivation
15	1	'I'	Absence of previously archived transactions
16	1	'D'	Data file (DA) unavailable
17	1	'J'	Re-initialization without backup

The session number and the date are exclusive. They are ignored if it is indicated that there are no previously archived transactions.

The unavailability of the Data file is to be indicated only when this file has been physically deleted (see paragraph 'RECOMMENDATIONS').

The reinitialization request without a backup is necessary when the Journal file is physically destroyed.

CAUTION:

In this case, the previous archiving is not duplicated on the output archiving. When the cataloging is automatic, previous archiving may be lost if no uncataloging is performed.
In case of an error on one of the options, an error message is printed and the archiving is generated using the default options.

RECOMMENDATIONS

If there is no user input, this procedure can be executed only if the database is in a consistent state, and if the Journal file is correctly formatted.

When a database needs to be restored after a problem, some information in the database may be destroyed and neither the DARC nor the DRST procedures can then be executed.

In this case, AND IN THIS CASE ONLY, columns 15 to 17 of the user input must be used as follows:

. If the Data file (DA) is lost or has been flagged as 'inconsistent', a 'D' in column 16 means that the DARC procedure will not take the Data file (DA) into account. However, the DRST procedure must be executed afterwards, since under these conditions, the DARC procedure renders the database inconsistent.

. If the Journal file (DJ) is lost or destroyed, a 'J' must be entered in column 17. The DARC procedure formats an empty Journal file. The DRST procedure can then be executed.

. If the sequential Archived file (BJ) is lost or destroyed, an 'I' must be entered in column 15. The DARC procedure will format a new sequential archive file.

If one of these columns is accidentally set to its value, and the DARC procedure executed when the Data (DA) file is in a consistent state, the consequences are :

. 'I' in col. 15: The transactions previously archived are lost. All the transactions can be recovered by concatenating BJ(-1) and BJ(0) to obtain BJ(+1).

. 'D' in col. 16: The DARC procedure has to be re-run BEFORE any update. If it is done afterwards, the data is lost and a complete restoration must be executed.

. 'J' in col. 17 : The contents of the Journal file are lost and cannot be retrieved.

REPORT RESULTS

This procedure prints a report giving the number of archived update transactions and, if applicable, the number of records that have been deactivated.

GENERAL RESULTS

Once this procedure is executed, a sequential file containing all the archived transactions is produced.

The Journal file is re-initialized.

It is also possible to store in another file all update transactions that have been deactivated.

NOTE: This procedure does not increment the current session number of the database.

FIRST ARCHIVING OF THE DATABASE

In order that the first archiving process can perform normally, the BJ file of archived transactions, used as input of the procedure, is provided in the form of an empty file in the database's SAVE directory.

DEACTIVATION OF THE ARCHIVED TRANSACTIONS

Two situations are possible for the deactivation of the transaction file:

. The BJ file's deactivated archive is not kept: file PACDBQ (internal name) must be assigned as '/dev/null' (as a default, this is done in the procedure's command file).

. The BJ file's deactivated archive is kept: file PACDBQ (internal name) must be assigned and correspond to a file on a disk. The procedure's command file must be modified (example: "PACDBQ=$DSMSDIR/save/BQ").

7.3. DESCRIPTION OF STEPS

DARC: DESCRIPTION OF STEPS

ARCHIVAL OF JOURNAL FILE: PDS300

This step executes the following:

. Updates the file of archived update transactions,

. Positions a flag in the Data file indicating the journal archiving,

. Writes the deactivated transactions onto a special file, if deactivation is requested by user input.

.Input files:
 -User transaction
 MBDARC file in temporary directory
 -Already archived transactions
 BJ file in SAVE directory
 -Journal file to reinitialize
 DJ file in JOURNAL directory
 -Error message file

.Input-Output file:
 -Data file
 DA file

.Output files:
 -Archived update transactions
 BJ.NEW file in SAVE directory
 -Deactivated archived trans.
 This file can be assigned if necessary

.Output report:
 -Review of archival

.Return codes:
 - 0: No error detected on the files
 - 8: User Input error
 -12: Input-output error on a file.

RE-INITIALIZATION OF THE JOURNAL FILE: PDS320

This step executes the following:

. Creates a record in the Journal file

. Repositions the Data file flag.

.Input files:
 -User transaction
 MBDARC file in INPUT directory
 -Error-message file

.Input-Output file:
 -Data file
 DA file

.Output file:

```
 -Journal file to be reinitialized
 DJ file in JOURNAL directory

.Output report:
 -Reinitialization report
  in temporary directory
```

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

7.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DARC BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "--------------------------------------------------------"
echo "                          DARC PROCEDURE"
echo "                          =============="
echo "Directory 'assign'                : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACDDJ.ini
echo "Directory 'journal'               : `dirname $PACDDJ.`"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                   : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                 : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
. $DSMSDIR/assign/$1/PACSAVBQ.ini
echo "Assignment of the BQ file        : $PACSAVBQ"
echo "--------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : DSMS JOURNAL ARCHIVING
# **************************************************************
# * INPUT TRANSACTION FORMAT :
# * .COMMAND FOR DEACTIVATING ARCHIVED TRANSACTIONS
# *  COL  2    : 'S'
# *  COL  3-6  : SESSION NUMBER
# *  COL  7-14 : DATE (SSYYMMDD)
# *  COL 15    : ' ' PRESENCE OF THE ARCHIVED TRANSACTION FILE
# *            : 'I' ABSENCE OF THE ARCHIVED TRANSACTION FILE
# *  COL 16    : ' ' PRESENCE OF THE DATA FILE (DJ)
# *            : 'D' ABSENCE OF THE DATA FILE (DJ)
# *  COL 17    : ' ' ARCHIVING AND REINITIALIZATION
# *            : 'J' REINITIALIZATION WITHOUT ARCHIVING
# *
# * IF THERE IS NO ENTRY, OR IF THERE IS AN ERROR ON A COMMAND
# * PARAMETER, DEACTIVATION DOES NOT TAKE PLACE BUT ARCHIVING
# * AND REINITIALIZATION ARE NORMALLY EXECUTED.
# *
# * TRANSACTIONS WHOSE SESSIONS (DATES) ARE LOWER OR EQUAL TO
# * THAT SPECIFIED WILL NOT BE KEPT. THEY WILL BE RETRIEVED IN
# * THE DEACTIVATED TRANSACTION FILE.
# *
# **************************************************************
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDA.ini
. $DSMSDIR/assign/$1/PACDDJ.ini
. $DSMSDIR/assign/$1/PACSAVBJ.ini
PACDJB=$PACSAVBJ
export PACDJB
PACDBJ=$PACSAVBJNEW
export PACDBJ
PACDMB=$PACINPUT'MBDARC'
export PACDMB
PACDBQ=/dev/null
export PACDBQ
PACDRU=$PACTMP'DARCRU.300'
export PACDRU
echo "Execution : PDS300"
```

```
cobrun PDS300
RETURN=$?
case $RETURN in
 0)
   . $DSMSDIR/assign/$1/PACDDE.ini
   . $DSMSDIR/assign/$1/PACDDA.ini
   . $DSMSDIR/assign/$1/PACDDJ.ini
   PACDMB=$PACINPUT'MBDARC'
   export PACDMB
   PACDRU=$PACTMP'DARCRU.320'
   export PACDRU
   echo "Execution : PDS320"
   cobrun PDS320
   RETURN=$?
   case $RETURN in
    0)
      echo "End of procedure"
      echo ""
      echo "Call the file BJBACKUP.ini"
      sh $DSMSDIR/assign/$1/BJBACKUP.ini
      ;;
    *)
      echo "Error in executing PDS320"
      echo "Error $RETURN"
      ;;
    esac
   ;;
12)
   echo "Error in executing PDS300"
   echo "Error $RETURN : Input-Output on a file"
   ;;
 *)
   echo "Error in executing PDS300"
   echo "Error $RETURN"
   ;;
esac
if [ "$RETURN" != '0' ]
then
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
fi
exit $RETURN
```

8. PRINTING OF QUERIES AND OUTPUT REPORTS (DPRT)

8.1. INTRODUCTION

DPRT: INTRODUCTION

The DPRT procedure performs all the printing operations for DSMS:

. Results of User Queries on Events, Changes and Sites, (this order must be respected)

. Printouts of Tables, Keywords, Queries and Reports.

See the DSMS Reference Manual for practical information on how to submit a DPRT execution in either batch or on-line mode.

NOTE: Printouts of Tables and Keywords can be submitted in batch mode only.

Technical information regarding the JOB Function allowing for DPRT on-line submissions is given at the end of this chapter.

EXECUTION CONDITIONS

None.
The Database can remain open to on-line processing.

ABNORMAL EXECUTION

Refer to Chapter THE BATCH PROCEDURES, Subchapter 'Abnormal Execution'.

8.2. INPUT - PROCESSING - RESULTS

USER INPUT

A '*' line (required):

Col.	Len.	Value	Description
2	1	'*'	Line Code
3	8	uuuuuuuu	DSMS User Code
11	8	pppppppp	Password
19	3	ppp	Product Code
22	2	su	Subsidiary Code
24	1	l	Language Code

4 report types exist, 1 line per printout is necessary :

Col.	Len.	Value	Description
TABLES			
02	03	Txx	Table codes for Txx
06	02	C1	... with their label in connected user language (default option)
06	02	C2	... with all labels
02	03	TUD	User codes with all authorizations (TUG + TUP + TUS)
QUERIES / REPORTS			
02	04	X QC	Query on Changes
		X QE	Query on Events
		X QS	Query on Sites
02	04	X RC	Report on Changes
		X RE	Report on Events
		X RS	Report on Sites
06	06	xxxxxx	Query or Report code
12	08	uuuuuuuu	User code for Query or Report owner (default value: connected user code)
20	02	C1	Print of all description pages for the Query/Report type (default option)
		C2	Print of only useful Query/Report description lines

Col.	Len.	Value	Description
LISTS			
02	03	LJQ	Control cards
02	04	LCQC	Query on Changes
		LCQE	Query on Events
		LCQS	Query on Sites
02	04	LCRC	Reports on Changes
		LCRE	Reports on Events
		LCRS	Reports on Sites
07	02	C1	Print of all description pages for the Query/Report type (default option)
		C2	Print of only useful Query/Report description lines
12	08	uuuuuuuu	User code for Query/Report owner
KEYWORDS			
02	04	LAKC	Stand-alone Keywords for Changes
		LPKC	Principal keywords for Changes
		LGKC	All keywords for Changes
06	01	l	Keywords language code (default: connected user language code)
02	04	LAKE	Stand-alone Native Keywords for Evnts
		LPKE	Principal Native Keywords for Events
		LGKE	All Native Keywords for Events
02	04	LAKT	Stand-alone Techn. Keywords for Evnts
		LPKT	All main keywords for Events
		LGKT	All keywords

Col.	Len.	Value	Description
			.PRINT VIA USER QUERY:
5	6	rrrrrr	Code of the user Query (required) 'Q' Entity used.
5	6	mmmmmmm	Code of the Report (optional)
17	1	d	Delimiter (optional)
			Parameters:
18	1	s	Symbol -
19	1	x	Separator -
20	54	.....	Parameter values -
			If optional fields have not been filled in, default values are used. They come from the definition lines of the user Query found in the Data-Base.

PRINTED OUTPUT

Two types of printed output are obtained:

- Results of user-defined Queries on Events, Changes and Sites.

- Standard printouts of Tables, Keywords, Queries and Reports.

RETURN CODE

0	OK with Queries
4	OK with Tables, kw, Query or Report list requests
8	OK with erroneous Query or other requests
12	Fatal error
16	Sort error

8.3. DESCRIPTION OF STEPS

DPRT: DESCRIPTION OF STEPS

This procedure calls a unique program (PDSB) that acts as a flow monitor for the various programs, which are therefore sub-programs of this monitor.
The procedure includes the following steps:

The input file is automatically formatted when QUERIES are submitted on-line.

```
PRINTING:  PDSB

.Permanent input files:
 -Data file
  DA file
 -VA Pac element file
  DC file
 -Error message file
  DE file

.Input file:
 -User Queries
 MBDPRT file in INPUT directory

.Work files:
 -Print requests
  KD file
 -Queries
  KQ file
 -Temporary files
  WKD, WKQ, W1....W4 in temporary directory

.Output reports:
 -Flow report
  DPRT.DA in temporary directory
 -List of Queries and requests
  DPRT.DB in temporary directory
 -Print of tables and keywords
  DPRT.DD in temporary directory
 -Report of Query extractions
  DPRT.DQ in temporary directory
 -Print of Query extractions
  DPRT.QI in temporary directory
 -Print of Queries/Reports
  DPRT.RQ in temporary directory
 -Print of control cards
  DPRT.JQ in temporary directory


SYMBOLICS IN USE

+---------+---------------------------------------------+
!SYMBOLICS! MEANING                                     !
+---------+---------------------------------------------+
! $1      ! Database name                               !
! $2      ! Radical of 'input' and 'tmp' files          !
! $3      ! Suffix of 'input' and 'tmp' directories     !
+---------+---------------------------------------------+
```

8.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DPRT BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                         DPRT PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : PRINTS AND QUERIES
# **************************************************************
# * INPUT TRANSACTION FORMAT :
# * .ONE LINE FOR USER CODE AND DSMS PASSWORD
# *  COL  2    : *
# *  COL  3    : DSMS USER CODE
# *  COL 11    : PASSWORD
# *  COL 19-21 : PRODUCT CODE
# *  COL 22-23 : SUBSIDIARY CODE
# * .LINE(S)  EXTRACTION COMMAND(S)
# *  COL  2-4  : EXTRACTION TYPE
# *            : 'Txx' TABLE EXTRACTION   (Txx = TABLE CODE)
# *            : 'QC ' DSMS CHANGE EXTRACTION
# *            : 'QE ' DSMS EVENT EXTRACTION
# * .IF QUERY 'QC' OR 'QE'
# *  COL  5-10 : USER QUERY CODE            (REQUIRED)
# *  COL 11    : DELIMITER                  (OPTIONAL)
# *  COL 12    : SYMBOL                     (   "    )
# *  COL 13    : SEPARATOR                  (   "    )
# *  COL 14-73 : VALUES OF PARAMETERS       (   "    )
# **************************************************************
. $DSMSDIR/assign/$1/PACDDA.ini
. $DSMSDIR/assign/$1/PACDDC.ini
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDX.ini
PACDMB=$PACINPUT'MBDPRT'
export PACDMB
PACDIA=$PACTMP'DPRT.DA'
export PACDIA
PACDIB=$PACTMP'DPRT.DB'
export PACDIB
PACDID=$PACTMP'DPRT.DD'
export PACDID
PACDIQ=$PACTMP'DPRT.DQ'
export PACDIQ
PACDQI=$PACTMP'DPRT.QI'
export PACDQI
PACDQR=$PACTMP'DPRT.QR'
export PACDQR
PACDRQ=$PACTMP'DPRT.RQ'
export PACDRQ
PACDJQ=$PACTMP'DPRT.JQ'
```

```
export PACDJQ
PACDQJ=$PACTMP'DPRT.QJ'
export PACDQJ
PACDKD=$PACTMP'WKD'
export PACDKD
PACDKQ=$PACTMP'WKQ'
export PACDKQ
PACDW1=$PACTMP'W1'
export PACDW1
PACDW2=$PACTMP'W2'
export PACDW2
PACDW3=$PACTMP'W3'
export PACDW3
PACDW4=$PACTMP'W4'
export PACDW4
echo "Execution : PDSBE"
cobrun PDSBE
RETURN=$?
case $RETURN in
 0)
   echo "End of procedure"
   echo ""
   echo "Deletion of the temporary files"
   rm -f $PACTMP'W'*
   ;;
 8)
   echo "End of procedure"
   echo "Some requests are erroneous"
   echo ""
   echo "Deletion of the temporary files"
   rm -f $PACTMP'W'*
   ;;
 *)
   echo "Error in executing PDSBE"
   echo "Error $RETURN"
   ;;
esac
if [ "$RETURN" != '0' ]
then
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
fi
exit $RETURN
```

9. DATABASE RESTORATION (DRST)

9.1. INTRODUCTION

DRST: INTRODUCTION

The Database Restoration procedure (DRST) restores the files, using the sequential image produced by the Database Backup procedure (DSAV).

Archived transactions can also be retrieved once this procedure has been executed.

EXECUTION CONDITIONS

The database must be closed to on-line processing.

Even if the closing of on-line access is not controlled by the procedure, it prevents any other update while the procedure is being executed.

The procedure physically and logically re-initializes the Journal file which must have been saved previously by the DARC procedure.

ABNORMAL EXECUTION

Refer to Subchapter 'Abnormal Execution' in Chapter THE BATCH PROCEDURES.

Whatever the cause of the abend, the procedure can be restarted as it is, after the problem has been solved.

DEFINITION CONTROL SUB-PROGRAMS

Sub-programs (delivered as COBOL sources) are designed to add specific controls or initializations on the 5 DSMS definitions.
At the beginning, these sources only include 3 examples:

- 1 'WARNING'-type error
- 1 critical error
- 1 initialization.

Their linkage is made up of the displayed fields, the entered fields or some other fields directly or indirectly associated with the definition.

At these sub-programs' return, an error message can then be displayed or the values of the displayed fields can be overwritten.

NOTES:

. The usual controls on definitions are executed before and after their call.

. When WARNING errors are set, a message is sent to the Definition screen and the sub-program is recalled to reinitialize the PR which is set to 'W'.

These sub-programs are called via tops indicated in the technical record of the DRST procedure.

9.2. INPUT - PROCESSING - RESULTS

USER INPUT

The following chart lists the DRST procedure's input.

POS.	LEN.	VALUE	MEANING
2	1	'R'	Line code
3	1	'l'	Language code 'E' or 'F' (optional)
4	1		Journal inhibition flag
		'0'	No inhibition (default option)
		'1'	Inhibition
5	1		Not used
6	3	'REC'	Restoration and retrieval of archived transactions
9	12		12-position table indicating the PFkeys assignment (default: 123456789ABC, but you may move or set to blank one or several values)
21	1		SECURITY SYSTEM INTERFACE
		' '	Retrieval of the previous value or no interface (for creation)
		'&'	Clear = Deactivation
		'R'	RACF
		'S'	TOPSECRET
22	1		USER CONTROL USING ON-LINE RACF
		' '	Retrieval of the previous value
		'&'	Clear = it is possible to enter a user-password different from the one entered at the firt connection
		'N'	It is not possible to enter another user-password
23	1	'C'	Encryption of passwords
		'D'	Decryption of passwords
		' '	Unchanged passwords NOTE: it is not advised at all to request an encryption or decryption of passwords at the same time as the retrieval of archived transactions request (because the action is not performed on the journal).

COL.	Len.	Value	Designation
26	1	'C'	Call of the sub-routine of additional controls for Change definition
		'&'	No call of sub-routine
27	1	'E'	Call of the sub-routine of additional controls for Event definition
		'&'	No call of sub-routine
28	1	'Q'	Call of the sub-routine of additional controls for Query definition
		'&'	No call of sub-routine
29	1	'R'	Call of the sub-routine of additional controls for Report definition
		'&'	No call of sub-routine
30	1	'S'	Call of the sub-routine of additional controls for Site definition
		'&'	No call of sub-routine

OUTPUT REPORT

This procedure prints a report listing the requested options, associated errors, the number of records restored in the database for each file, and the options memorized in the new database.

RESULT

Once this procedure is executed, the current session number is that of the sequential image or that of the most recent transaction, if the retrieval of archived transactions has been requested.

9.3. DESCRIPTION OF STEPS

DRST: DESCRIPTION OF STEPS

```
VALIDATION OF JOURNAL CONTENTS:  PDS380

This step is executed only when the Journal file exists. In
this case, it verifies that the journal has been archived.

.Input files:
 -Journal file
 DJ file in JOURNAL directory
 -Error message file

.Output report:
 -AJ file status report
 It is printed if the journal file has not been archived.

.Return codes:


 -0:  The Journal file was archived.
 -4:  The Journal file was not archived.
      In this case, none of the DRST steps is executed.

DATABASE RESTORATION:  PDS400

This step is executed only when the Journal file has been
archived.

.Permanent input files:
 -Backup of the files
 BB file in SAVE directory
 -Error message file

.Permanent output files:
 -Data file
 DA file
 -VA Pac element file
 DC file
 -Journal file
 DJ file in JOURNAL directory
 -Cross-reference file
 DX file

.Input transaction file:
 -User transactions
 MBDRST file in INPUT directory

.Output file:
 -Work file    (2 records)
 MS file in temporary directory

.Output report:
 -Restoration report

RETRIEVAL OF ARCHIVED JOURNAL:  PDS450

This step is executed only when there are transactions to be
retrieved. It does not cause a 'journalization' of processed
transactions.
```

.Permanent input-output files:
-Data file
DA file
-VA Pac element file
DC file
-Cross-reference file
DX file

.Input files:
-Work file (2 records)
MS file in temporary directory
-Error message file

.Input archived file:
-Archiving of the journal to retrieve
BJ file in SAVE directory

.Output report:
-Update report

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

9.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DRST BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                         DRST PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACDDJ.ini
echo "Directory 'journal'              : `dirname $PACDDJ.`"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : LOADING-RESTORING DSMS DATABASE
# **************************************************************
# * INPUT TRANSACTION FORMAT :
# * .ONE COMMAND LINE
# *  COL  2    : R
# *  COL  3    : INITIAL LANGUAGE CODE ( E=ENGLISH, F=FRENCH)
# *  COL  4    : 1 DEACTIVATION OF TRANSACTIONS' LOG
# *  COL  5    : SYSTEM DATE FORMAT (N FOR DD/MM/YY)
# *            :                    (I FOR MM/DD/YY)
# *  COL  6-8  : REC FOR RETRIEVAL OF ARCHIVED TRANSACTIONS
# *            : INI FOR INITIALIZATION OF FILES
# *  COL 09-20 : (NOT USED)
# *
# * IF YOU DO NOT RE-INITIALIZE THE DJ FILE OF DISK TRANSACTIONS
# * THE RESTORATION PROCEDURE WILL NOT BE EXECUTED. IN THIS CASE
# * EXECUTE THE ARCH PROCEDURE FIRST.
# *
# **************************************************************
. $DSMSDIR/assign/$1/PACDDJ.ini
if [ -r $PACDDJ ]
then
. $DSMSDIR/assign/$1/PACDDE.ini
PACDRU=$PACTMP'DRSTRU.380'
export PACDRU
echo "Execution : PDS380"
cobrun PDS381
RETURN=$?
case $RETURN in
 0)

   ;;
 8)
   echo "Error in executing PDS380"
   echo "Error $RETURN : the journal was not archived"
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
   exit $RETURN
   ;;
 *)
   echo "Error in executing PDS380"
   echo "Error $RETURN"
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
   exit $RETURN
   ;;
esac
fi
# ***************************************************************
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDA.ini
. $DSMSDIR/assign/$1/PACDDX.ini
. $DSMSDIR/assign/$1/PACDDC.ini
. $DSMSDIR/assign/$1/PACDDJ.ini
. $DSMSDIR/assign/$1/PACSAVBB.ini
PACDBB=$PACSAVBB
export PACDBB
PACDMB=$PACINPUT'MBDRST'
export PACDMB
PACDRU=$PACTMP'DRSTRU.400'
export PACDRU
PACDMS=$PACTMP'MS'
export PACDMS
echo "Execution : PDS400"
cobrun PDS400
RETURN=$?
case $RETURN in
 0)
   . $DSMSDIR/assign/$1/PACDDE.ini
   . $DSMSDIR/assign/$1/PACDDA.ini
   . $DSMSDIR/assign/$1/PACDDX.ini
   . $DSMSDIR/assign/$1/PACDDC.ini
   . $DSMSDIR/assign/$1/PACSAVBJ.ini
   PACDBJ=$PACSAVBJ
   export PACDBJ
   PACDRU=$PACTMP'DRSTRU.450'
   export PACDRU
   PACDMS=$PACTMP'MS'
   export PACDMS
   echo "Execution : PDS450"
   cobrun PDS450
   RETURN=$?
   case $RETURN in
    0)
      echo "End of procedure"
      echo ""
      echo "Deletion of temporary files"
      rm -f $PACTMP'MS'*
      ;;
    *)
      echo "Error in executing PDS450"
      echo "Error $RETURN"
      ;;
   esac
   ;;
 *)
   echo "Error in executing PDS400"
   echo "Error $RETURN"
   ;;
esac
if [ "$RETURN" != '0' ]
then
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
fi
exit $RETURN
```

10. DATABASE BACKUP (DSAV)

10.1. INTRODUCTION

DSAV: INTRODUCTION

The purpose of the backup procedure (DSAV) is to convert the main files that make up DSMS into a BB sequential format.

The backed-up files are :

. The Data file (DA),

. The VA Pac Element file (DC),

. The Cross-reference file (DX).

EXECUTION CONDITION

The database must be closed to on-line processing in order to ensure its consistency during the execution of the DSAV procedure.

ABNORMAL EXECUTION

Refer to Subchapter 'Abnormal Execution' in Chapter THE BATCH PROCEDURES.

The main cause of an abend is that the database has not been closed to on-line use.

After correction, the procedure can be restarted as it is.

10.2. INPUT - PROCESSING - RESULTS

<u>USER INPUT</u>

One optional line code.

Col.	Len.	Value	Designation
2	1	'O'	Line Code
3	3	'ENC'	Encryption of passwords
		'DEC'	Decryption of passwords
		' '	Unchanged passwords

<u>REPORT RESULTS</u>

Once the backup is executed, a report is printed. It includes the number of records saved in each file and the session number.

<u>OUTPUT RESULT</u>

The output is a single sequential file (BB) of variable length, containing the image of the three saved files.

If the database is in an inconsistent state as a result of an abnormal end in the last update, the DSAV procedure is not executed.

<u>NOTE:</u>

The DSAV procedure increments the current session number.

10.3. DESCRIPTION OF STEPS

DSAV: DESCRIPTION OF STEPS

DATABASE CONSISTENCY CHECK: PDSBAS

.Permanent input files:
 -Data file
  DA file
 -Error message file

.Output report
 -Validity report

Return code

This utility sends a return code 4 and causes an ABEND
in case of database invalidity.

DATABASE BACKUP: PDS500

.Input-Output file:
 -Data file
  DA file

.Permanent Input files:
 -VA Pac element file
  DC file
 -Cross-reference file
  DX file
 -Error message file

.Input transaction file:
 -User transactions
  MBDSAV file in INPUT directory

.Output file:
 -Sequential image of files
  BB.NEW file in SAVE directory

.Output report:
 -Backup report
  in temporary directory

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

10.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DSAV BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "----------------------------------------------------------"
echo "                         DSAV PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "----------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : DSMS DATABASE BACKUP
# **************************************************************
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDA.ini
PACDRS=$PACTMP'DSAVRS.BAS'
export PACDRS
echo "Execution : PDSBAS"
cobrun PDSBAS
RETURN=$?
case $RETURN in
 0)
   . $DSMSDIR/assign/$1/PACDDE.ini
   . $DSMSDIR/assign/$1/PACDDA.ini
   . $DSMSDIR/assign/$1/PACDDX.ini
   . $DSMSDIR/assign/$1/PACDDC.ini
   . $DSMSDIR/assign/$1/PACSAVBB.ini
   PACDBB=$PACSAVBBNEW
   export PACDBB
   PACDMB=$PACINPUT'MBDSAV'
   export PACDMB
   PACDRU=$PACTMP'DSAVRU.500'
   export PACDRU
   echo "Execution : PDS500"
   cobrun PDS500
   RETURN=$?
   case $RETURN in
    0)
      echo "End of procedure"
      echo ""
      echo "Call the file BBBACKUP.ini"
      sh $DSMSDIR/assign/$1/BBBACKUP.ini
      ;;
    *)
      echo "Error in executing PDS500"
      echo "Error $RETURN"
      ;;
   esac
   ;;
 4)
   echo "Error in executing PDSBAS"
   echo "Database unavailable"
   ;;
 *)
   echo "Error in executing PDSBAS"
```

```
       echo "Error $RETURN"
       ;;
    esac
    if [ "$RETURN" != '0' ]
    then
       sh $DSMSDIR/batch/proc/ERRPAUSE.ini
    fi
    exit $RETURN
```

11. REORGANIZATION OF CROSS-REFERENCES (DREO)

11.1. INTRODUCTION

INTRODUCTION

The Cross-Reference Reorganization procedure (DREO) rebuilds a sequential image of the database using another sequential image as a starting point. The resulting file is used as input to the Restoration (DRST) procedure.

The operating principle of this procedure is to rebuild the cross-references associated with the data from the 'image' of this data.

EXECUTION CONDITIONS

The database can remain open during reorganization since the procedure operates on sequential images of the database (backups).

The updates executed after the file backing up used for reorganization, can be retrieved during the restoration of the reorganized database.

ABNORMAL EXECUTION

Refer to Subchapter 'Abnormal Execution' in Chapter THE BATCH PROCEDURES.

In case of an abnormal end, the procedure must be restarted from the beginning.

11.2. INPUT - PROCESSING - RESULTS

USER INPUT

Three different types of user input can be entered, but only one line of each type can be created.

The format of this input is given in the table below.

POS.	LEN.	VALUE	MEANING
1	1	Not Used	
2	1	'P'	Deletion of Products
	1	'S'	Deletion of Subsidiaries
	1	'X'	Deletion of Product/Subsidiary
3	60	Product code	(20 x 3 char.) if Col.2 = 'P'
	60	Subsid. code	(30 x 2 char.) if Col.2 = 'S'
	60	Prod./ Subsid.	(12 x 5 char.) if Col.2 = 'X'

REPORT

This procedure prints messages stating inconsistencies found in the Data file.

RESULT

The result of this procedure is a reorganized sequential image of the DSMS database, used as input to the Restoration (DRST) procedure.

11.3. DESCRIPTION OF STEPS

DREO : DESCRIPTION OF STEPS

```
BUILDING OF INDEXES (not keywords):  PDSR10

.Input file:
 -Input file
  MBDREO file in INPUT directory

.Permanent Input files:
 -DSMS database backup
  BB file in SAVE directory
 -Error message file

.Work files:
 -Data and VA Pac elements
  W1 file in temporary directory
 -Keywords and keyword references
  W2 file in temporary directory
 -Cross-references (not keywords)
  W3 file in temporary directory
 -Sort files

.Output reports:
 -Inconsistencies in DSMS data


 -Reorganization report

BUILDING OF KEYWORD INDEXES:  PDSR20

.Work files:
 -Keywords and keyword references
  W2 file in temporary directory
 -Keywords
  W4 file in temporary directory
 -Keyword references
  W5 file in temporary directory
 -Sort files

MERGE OF INDEXES:  PDSR30

.Work files:
 -Cross-references (except keywords)
  W3 file in temporary directory
 -Keyword references
  W5 file in temporary directory
 -Keyword references
  W6 file in temporary directory
 -Sort files

GENERAL MERGE FOR BACKUP:  PDSR40

.Work files:
 -Data and VA Pac elements
  W1 file in temporary directory
 -Keywords
  W4 file in temporary directory
 -Keyword references
  W6 file in temporary directory
 -Sort files

.Permanent input file:
```

-Error message file

.Permanent output file:
-Reorganized DSMS database backup
BB.NEW file in SAVE directory

.Output report:
-Reorganization report
in temporary directory

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

11.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DREO BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                         DREO PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : DSMS REORGANIZATION
# **************************************************************
# * INPUT TRANSACTION FORMAT :
# * .ONE LINE OF EACH TYPE AT THE MAXIMUM
# *  COL  2    : 'P' DELETION OF PRODUCTS
# *            : 'S' DELETION OF SUBSIDIARIES
# *            : 'X' DELETION OF PRODUCTS/SUBSIDIARIES
# *  COL  3-62 : IF P, PRODUCT CODE    (20 x 3 CAR.)
# *            : IF S, SUBSIDIARY CODE (30 x 2 CAR.)
# *            : IF X, PRODUCT/SUBSIDIARY (12 x 5 CAR.)
# **************************************************************
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACSAVBB.ini
PACDBB=$PACSAVBB
export PACDBB
CARTE=$PACINPUT'MBDREO'
export CARTE
PACDRH=$PACTMP'DREPRH.R10'
export PACDRH
PACDRK=$PACTMP'DREORK.R10'
export PACDRK
PACDW1=$PACTMP'W1'
export PACDW1
PACDW2=$PACTMP'W2'
export PACDW2
PACDW3=$PACTMP'W3'
export PACDW3
echo "Execution : PDSR10"
cobrun PDSR10
RETURN=$?
case $RETURN in
 0)
   PACDW2=$PACTMP'W2'
   export PACDW2
   PACDW4=$PACTMP'W4'
   export PACDW4
   PACDW5=$PACTMP'W5'
   export PACDW5
   echo "Execution : PDSR20"
   cobrun PDSR20
   RETURN=$?
   case $RETURN in
```

```
        0)
          PACDW3=$PACTMP'W3'
          export PACDW3
          PACDW5=$PACTMP'W5'
          export PACDW5
          PACDW6=$PACTMP'W6'
          export PACDW6
          echo "Execution : PDSR30"
          cobrun PDSR30
          RETURN=$?
          case $RETURN in
           0)
             . $DSMSDIR/assign/$1/PACDDE.ini
             PACDBB=$PACSAVBBNEW
             export PACDBB
             PACDRR=$PACTMP'DREORR.R40'
             export PACDRR
             PACDW1=$PACTMP'W1'
             export PACDW1
             PACDW4=$PACTMP'W4'
             export PACDW4
             PACDW6=$PACTMP'W6'
             export PACDW6
             echo "Execution : PDSR40"
             cobrun PDSR40
             RETURN=$?
             case $RETURN in
              0)
                echo "End of procedure"
                echo ""
                echo "Deletion of the temporary files"
                rm -f $PACTMP'W'*
                echo ""
                echo "Call the file BBBACKUP.ini"
                sh $DSMSDIR/assign/$1/BBBACKUP.ini
                ;;
              *)
                echo "Error in executing PDSR40"
                echo "Error $RETURN"
                ;;
             esac
             ;;
           *)
             echo "Error in executing PDSR30"
             echo "Error $RETURN"
             ;;
          esac
          ;;
        *)
          echo "Error in executing PDSR20"
          echo "Error $RETURN"
          ;;
       esac
       ;;
     *)
       echo "Error in executing PDSR10"
       echo "Error $RETURN"
       ;;
    esac
    if [ "$RETURN" != '0' ]
    then
       sh $DSMSDIR/batch/proc/ERRPAUSE.ini
    fi
    exit $RETURN
```

12. EXTRACTION FROM VA PAC ARCHIVED JOURNAL (DEXP)

12.1. INTRODUCTION

EXTRACTION FROM ARCHIVED JOURNAL (DEXP): INTRODUCTION

The Archived Journal Extraction procedure (DEXP) extracts transactions associated to Changes from the VA Pac Archived Journal file, and formats them in order to update, in the DSMS Database, the modified elements corresponding to each Change.

EXECUTION CONDITIONS

None.

ABNORMAL EXECUTION

Refer to Subchapter 'Abnormal Execution' in Chapter THE BATCH PROCEDURES.

If an abnormal end occurs, the procedure can be restarted as it is, after the problem has been solved.

NOTES:

The DEXP procedure operates with a VA Pac 2.0 or higher Journal.

The DEXQ procedure operates with a Journal in a VA Pac release lower than 2.0.

12.2. INPUT - PROCESSING - RESULTS

USER INPUT

One '*'-line is required:

POS.	LEN.	VALUE	MEANING
2	1	'*'	Line code
3	8	uuuuuuuu	DSMS user code
11	8	pppppppp	User password

One extraction line is also required:

POS.	LEN.	VALUE	MEANING
2	1	'J'	Line code (required)
			THE FOLLOWING FIELDS ARE OPTIONAL :
3	1	' '	List of selected transactions
		'N'	No list
4	24		Selection in the VA Pac Database:
4	4	nnnn	Session number, begin. of selection
8	4	pppp	Session number, end of selection
			--> Selection on session(s)
			prohibits selection on date(s)
12	8	CCYYMMDD	Starting date for selection
		'TODAY'	Starting date = current date
20	8	CCYYMMDD	Ending date for selection
		'TODAY'	Ending date = current date
			(default value if st. date ='today')
28	1		Version of selected transactions
		' '	Selection of all sessions
		'T'	Selection of frozen session
		'Z'	Selection of current session
29	3	ppp	Product code
32	4	xxxx	VA Pac Database logical code
36	3	lll	Code of selected library
39	16		Type of selected entities
55	1	' '	Extraction of transactions made
			under change 999999
		'N'	No extraction of 999999-change
			transactions
56	1	' '	Printing of duplicate transactions
			for the same VA Pac entity
		'N'	No printing of duplicate transact-
			ions
57	6	nnnnnn	Change number

REPORT

Extraction report showing the list of formatted transactions.

RESULT

A DSMS database update transaction file to be used as input to the DUPT procedure.

12.3. DESCRIPTION OF STEPS

DEXP: DESCRIPTION OF STEPS

```
TRANSACTION EXTRACTION AND FORMATTING:  PDS600

.Permanent imput files:
 -Data file
  DA file


 -Error message file


 -VA Pac archived journal
 VisualAge Pacbase PJ file in VA Pac SAVE directory

.Input transaction file:
 -User transactions
  MBDEXP file in INPUT directory

.Sort files:

.Output file:
 -Update transaction file for DUPT
   MVDEXP file in temporary directory

.Output report:
 -Report on selection request

.Return codes:
 - 0:  No error and no list requested
 -04:  No error and list of transactions printout requested
 -08:  Error on the user line or parameter input line
 -12:  I/O error on a file

PRINTING OF DSMS UPDATE TRANSACTIONS:  PDS610

.Permanent input files:
 -Data file
  DA file


 -Error message file

.Input File:
 -DSMS update transactions file
  MVDEXP file in temporary directory

.Output report:
 -List of update transactions
  in temporary directory

.Return codes:
 - 0:  No error
 -12:  I/O error on a file


SYMBOLICS IN USE
```

SYMBOLICS	MEANING
$1	Database name
$2	Directory of VA Pac archived journal file (PJ)
$3	Radical of 'input' and 'tmp' files
$4	Suffix of 'input' and 'tmp' directories

12.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DEXP BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "-------------------------------------------------------------"
echo "                          DEXP PROCEDURE"
echo "                          =============="
echo "Directory 'assign'                : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                   : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                 : `dirname $PACINPUT.`"
if [ -n "$3" ]
then
   echo "Radical 'tmp' and 'input' files  : $3"
fi
echo "Directory of PJ file (VA Pac)    : $2"
echo "-------------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# ***************************************************************
# * VA Pac DSMS : EXTRACTION OF VA Pac JOURNAL FOR DSMS UPDATE
# ***************************************************************
# * INPUT TRANSACTION FORMAT :
# * .ONE LINE FOR USER AND DSMS PASSWORD
# *  COL  2    : *
# *  COL  3    : DSMS USER CODE
# *  COL 11    : PASSWORD
# * .EXTRACTION COMMAND LINE(S)
# *  COL  2    : J
# *  COL  3    : ' ' LIST OF SELECTED TRANSACTIONS
# *            : 'N' NO LIST OF SELECTED TRANSACTIONS
# *  COL  4-7  : SESSION NUMBER FOR BEGINING OF SELECTION
# *  COL  8-11 : SESSION NUMBER FOR END OF SELECTION
# *  COL 12-19 : DATE FOR BEGINING OF SELECTION (SSYYMMDD)
# *  COL 20-27 : DATE FOR END OF SELECTION (SSYYMMDD)
# *  COL 28    : VERSION OF SELECTED TRANSACTIONS SESSION
# *            : ' ' ALL SESSIONS
# *            : 'T' TEST FROZEN SESSION
# *            : 'Z' CURRENT SESSION
# *  COL 29-31 : PRODUCT CODE
# *  COL 32-35 : VA Pac DATABASE LOGICAL CODE
# *  COL 36-38 : LIBRARY CODE
# *  COL 39-54 : TYPE OF ENTITIES TO BE SELECTED
# *  COL 55    : ' ' EXTR. OF TRANS. MADE UNDER CHANGE 999999
# *            : 'N' NO EXTRACTION OF 999999 CHANGE
# *  COL 56    : ' ' PRINTING OF DUPLICATE TRANSACTIONS
# *            : 'N' NO PRINTING OF DUPLICATE TRANSACTIONS
# *  COL 57-62 : CHANGE NUMBER
# ***************************************************************
. $DSMSDIR/assign/$1/PACDDA.ini
. $DSMSDIR/assign/$1/PACDDE.ini
PACDMB=$PACINPUT'MBDEXP'
export PACDMB
PACDMV=$PACINPUT'MVDEXP'
export PACDMV
PAC7PJ=$2/PJ
export PAC7PJ
PACDRU=$PACTMP'DEXPRU.600'
export PACDRU
echo "Execution : PDS600"
cobrun PDS600
```

```
RETURN=$?
case $RETURN in
 0)
   echo "No printing of selected transactions"
   ;;
 4)
   . $DSMSDIR/assign/$1/PACDDA.ini
   . $DSMSDIR/assign/$1/PACDDE.ini
   PACDMV=$PACINPUT'MVDEXP'
   export PACDMV
   PACDRU=$PACTMP'DEXPRU.610'
   export PACDRU
   echo "Execution : PDS610"
   cobrun PDS610
   RETURN=$?
   case $RETURN in
    0)
      echo "End of procedure"
      ;;
    *)
      echo "Error in executing PDS610"
      echo "Error $RETURN"
      ;;
   esac
   ;;
 *)
   echo "Error in executing PDS600"
   echo "Error $RETURN"
   ;;
esac
if [ "$RETURN" != '0' -a "$RETURN" != '4' ]
then
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
fi
exit $RETURN
```

13. EXTRACTION OF ENTITIES (DEXT)

13.1. INTRODUCTION

ENTITY EXTRACTION (DEXT): INTRODUCTION

The Entity Extraction procedure (DEXT) extracts all DSMS entities and formats them into batch transactions to be used as input to the DSMS Database Update procedure (DUPT).

PRINCIPLE

In order to select the extraction of Changes, Events or Sites, the procedure uses Queries ("Q" entities) that must have been previously defined in the DSMS Database. These three types of extraction must be requested in the above order.

The Query code should also be specified in the extraction request (see 'User Input').

The screen Report ("R" entity) associated with the Query used for the extraction does not interfere in the extraction.

EXECUTION CONDITIONS

None.

ABNORMAL EXECUTION

Refer to Subchapter 'Abnormal Execution' in Chapter THE BATCH PROCEDURES.

If an abnormal end occurs, the procedure can be restarted as it is after the problem has been solved.

13.2. INPUT - PROCESSING - RESULTS

USER INPUT

One '*'-line is required:

POS.	LEN.	VALUE	MEANING
2	1	'*'	Line code
3	8	uuuuuuuu	DSMS User code
11	8	pppppppp	User password
19	3	ppp	Product code
22	2	su	Subsidiary code
24	1	l	Language code

Four types of extractions are available. One line per request is necessary:

Pos.	Len.	Value	Meaning
02	03	'PL'	Locking of databases
02	03	Txx	Codes of the Txx table (all tables except TRA)
QUERIES / REPORTS:			
02	04	X QC	Query on Changes
		X QE	Query on Events
		X QS	Query on Sites
02	04	X RC	Report on Changes
		X RE	Report on Events
		X RS	Report on Sites
12	08	uuuuuuuu	Owner of the Query or Report (Default=logged-in user)

```
+----+-----+--------+-------------------------------------+
!Pos.! Len.! Value  ! Meaning                             !
+----+-----+--------+-------------------------------------+
! 02 !  04 !  LCQC  ! Queries on Changes                  !
!    !     !  LCQE  ! Queries on Events                   !
!    !     !  LCQS  ! Queries on Sites                    !
! 02 !  04 !  LCRC  ! Reports on Changes                  !
!    !     !  LCRE  ! Reports on Events                   !
!    !     !  LCRS  ! Reports on Sites                    !
! 12 !  08 !uuuuuuuu! Owner of Queries or Reports         !
+----+-----+--------+-------------------------------------+
! KEYWORDS:                                               !
+---------------------------------------------------------+
! 02 !  04 !  LAKC  ! Isolated keywords of Changes        !
!    !     !  LGKC  ! All Changes' Keywords               !
! 06 !  01 !   l    ! Language code of Keywords           !
!    !     !        ! (Default=Language of logged-in user) !
! 02 !  04 !  LAKE  ! Native isolated Keywords of Events   !
!    !     !  LGKE  ! All Events' Keywords                !
! 02 !  04 !  LAKT  ! Techn. isolated Keywords of Events   !
!    !     !  LGKT  ! All Keywords                        !
+----+-----+--------+-------------------------------------+
```

Pos.	Len.	Value	Meaning
			.EXTRACTION VIA USER QUERY:
5	6	rrrrrr	User Query code (required)
			- 'Q' Entity use
5	6	mmmmmm	Report code (optional)
17	1	d	Delimiter (optional)
			Parameter settings:

18	1	s	Symbol -
19	1	x	Separator -
20	54	.....	Parameter values -
			If some optional fields were not completed, default values will be used. They come from the User Query's definition lines found in the Database.

PRINTED OUTPUT

Extraction report showing the number of extracted transactions.

RESULT

DSMS database update transactions to be used as input to the DUPT procedure.

This procedure displays a general return code:

0	OK
8	Error on the user line code (*) or on a parameter line
12	I/O error or inconsistent DSMS database
16	Sort error

13.3. DESCRIPTION OF STEPS

DEXT: DESCRIPTION OF STEPS

This procedure calls a single program (PDSEX) that acts as a flow monitor for all programs, which are then considered as its sub-programs.

The procedure includes the following steps:

```
EXTRACTIONS:  PDSEX

.Permanent input files:
 -Data file
  DA file
 -VA Pac element file
  DC file
 -Error message file
  DE file

.Input transaction file:
 -Extraction requests
  MBDEXT file in INPUT directory

.Work files:
 -Queries
  PACDKQ
 -Temporary files
  in temporary directory

.Output reports:
 -Flow report
  PACDIA : DEXT.IA in temporary directory
 -Extraction request report
  PACDRU : DEXT.RU in temporary directory

.Sort files:

.Output file:
 -Extracted batch transactions
  MVDEXT file in INPUT directory
```

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name
$2	Radical of 'input' and 'tmp' files
$3	Suffix of 'input' and 'tmp' directories

13.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DEXT BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "----------------------------------------------------------"
echo "                         DEXT PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "----------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# ***************************************************************
# * DSMS : EXTRACTION OF BATCH TRANSACTIONS FOR DUPT
# ***************************************************************
# * INPUT TRANSACTION FORMAT :
# * .ONE USER AND DSMS PASSWORD LINE
# *  COL  2    : *
# *  COL  3    : DSMS USER CODE
# *  COL 11    : PASSWORD
# *  COL 19-21 : PRODUCT CODE
# *  COL 22-23 : SUBSIDIARY CODE
# * .LINE(S)  EXTRACTION COMMAND(S)
# *  COL  2-4  : EXTRACTION TYPE
# *            : 'Txx' TABLE EXTRACTION    (Txx = TABLE CODE)
# *            : 'QC ' DSMS CHANGE EXTRACTION
# *            : 'QE ' DSMS EVENT EXTRACTION
# *            : 'QS ' DSMS SITE EXTRACTION
# * .IF QUERY 'QC' OR 'QE' OR 'QS'
# *  COL  5-10 : USER QUERY CODE            (REQUIRED)
# *  COL 17    : DELIMITER                  (OPTIONAL)
# *  COL 18    : SYMBOL                     (    "    )
# *  COL 19    : SEPARATOR                  (    "    )
# *  COL 20-73 : VALUES OF PARAMETERS       (    "    )
# ***************************************************************
. $DSMSDIR/assign/$1/PACDDA.ini
. $DSMSDIR/assign/$1/PACDDC.ini
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDX.ini
PACDMB=$PACINPUT'MBDEXT'
export PACDMB
PACDIM=$PACINPUT'MVDEXT'
export PACDIM
PACDIA=$PACTMP'DEXT.IA'
export PACDIA
PACDRU=$PACTMP'DEXT.RU'
export PACDRU
PACDKQ=$PACTMP'WKQ'
export PACDKQ
PACDW0=$PACTMP'W0'
export PACDW0
PACDW1=$PACTMP'W1'
export PACDW1
PACDW2=$PACTMP'W2'
export PACDW2
```

```
PACDW3=$PACTMP'W3'
export PACDW3
PACDW4=$PACTMP'W4'
export PACDW4
PACDW5=$PACTMP'W5'
export PACDW5
PACDWI=$PACTMP'WI'
export PACDWI
echo "Execution : PDSEXE"
cobrun PDSEXE
RETURN=$?
case $RETURN in
 0)
   echo "End of procedure"
   echo ""
   echo "Deletion of the temporary files"
   rm -f $PACTMP'W'*
   ;;
 *)
   echo "Error in executing PDSEXE"
   echo "Error $RETURN"
   ;;
esac
if [ "$RETURN" != '0' ]
then
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
fi
exit $RETURN
```

14. EXTRACTION OF TABLES FOR EXTERNAL LISTS (DEXH)

14.1. INTRODUCTION

DEXH: INTRODUCTION

The DEXH procedure extracts all the information contained in DSMS tables in order to create a file that can be used by a developer's workstation.

With the resulting file, the developer can create 'Lists of external values', used by the 'revamped' (using the PAW function) DSMS workstations.

For further details, see the PAW OPERATOR'S HANDBOOK, chapter 'REVAMPING OF IBM PRODUCTS'.

EXECUTION CONDITIONS

None.

ABNORMAL EXECUTION

Refer to Subchapter 'Abnormal Execution' in Chapter THE BATCH PROCEDURES.

Whatever the cause of the abend, the procedure can be restarted as it is, once the problem has been solved.

14.2. INPUT - PROCESSING - RESULTS

USER INPUT

Pos.	Len.	Value	Meaning
2	1	'*'	Line code
3	8	uuuuuuuu	DSMS User code
11	8	pppppppp	Password
19	3	ppp	Product code
22	2	su	Subsidiary code

REPORT

Extraction report showing the list of extracted tables.

RESULT

All general tables (not linked to a specific product) as well as the OPTIONS, PHASES and VERSIONS tables of the product specified in the user input.

14.3. DESCRIPTION OF STEPS

DEXH: DESCRIPTION OF STEPS

EXTRACTION FOR PAW WORKSTATIONS: PDSXTH

This program extracts the values contained in tables: TST TSU, TGR, TPR, TRE, TTY, TUD, TAT, TLA, TPH, and TOP to be read on 'revamped' DSMS workstations.

.Permanent input files:
 -Data file
   DA file
 -Error message file

.Input transaction file:
 -User check
   MBDEXH file in temporary directory

.Output file:
 -Extracted tables

.Output report:
 -Extraction report

.Sort files:

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name
$2	Radical of 'input' and 'tmp' files
$3	Suffix of 'input' and 'tmp' directories

14.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DEXH BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "--------------------------------------------------------"
echo "                         DEXH PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "--------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# ************************************************************
# * DSMS : EXTRACTION OF TABLES FOR LOCAL LISTS
# ************************************************************
# * INPUT TRANSACTIONS FORMAT :
# * .ONE LINE FOR USER AND DSMS PASSWORD
# *  COL  2    : *
# *  COL  3    : DSMS USER CODE
# *  COL 11    : PASSWORD
# *  COL 19-21 : PRODUCT CODE
# *  COL 22-23 : SUBSIDIARY CODE
# ************************************************************
. $DSMSDIR/assign/$1/PACDDA.ini
. $DSMSDIR/assign/$1/PACDDE.ini
PACDMB=$PACINPUT'MBDEXH'
export PACDMB
PACDMV=$PACINPUT'MVDEXH'
export PACDMV
PACDRH=$PACTMP'DEXHRH.XTH'
export PACDRH
echo "Execution : PDSXTH"
cobrun PDSXTH
RETURN=$?
case $RETURN in
0)
  echo "End of procedure"
  ;;
*)
  echo "Error in executing PDSXTH"
  echo "Error $RETURN"
  ;;
esac
if [ "$RETURN" != '0' ]
then
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
fi
exit $RETURN
```

15. BATCH UPDATE OF ENTITIES (DUPT)

15.1. INTRODUCTION

DUPT: INTRODUCTION

The Batch Update of Entities procedure (DUPT) updates the DSMS entities with transactions from the DEXT, DEXP and/or DXBJ procedures.

Transactions can also be entered directly in a file, using an editor. For a complete description of the batch transactions, see the 'BATCH TRANSACTIONS STRUCTURE', in the appendix of the DSMS Reference Manual.

EXECUTION CONDITION

The DSMS files must be closed to on-line use.

ABNORMAL EXECUTION

Refer to Subchapter 'Abnormal Execution' in Chapter THE BATCH PROCEDURES.

Whatever the cause of the abend, the procedure can be restarted as it is after the problem has been solved.

CAUTION:

This procedure performs a GLOBAL update. Therefore, make sure that all the data fields have been filled in. The data fields that are not filled in will automatically be set to blank.

The Change, Event and Site definition screens require two update lines, and both lines must be filled.

DSMS automatically allocates numbers to Events or Changes when they are created. However, for its creation, an Event or Change must be allocated a temporary number. For example, to create a Change: C000001, where 000001 is the temporary number that DSMS will automatically replace with a unique number.

You must set the action code to 'C', since the system does not provide for implicit creation.

Several Changes or Events can be created simultaneously. In this case, each Change or Event being created must be allocated a different temporary number. For exemple, to create 3 Changes simultaneously: C000001, C000002 and C000003.

NOTE: Each transaction stream can only contain 2,520 changes and 2,520 events maximum (internal limit of the program).

15.2. INPUT - PROCESSING - RESULTS

USER INPUT

. One Parameter line (optional).

. One Identification line per Product/Subsidiary concerned by the updates (required).

. Update transactions extracted and formatted by the DEXT, DEXP or DXBJ procedures.

. The user must add at least one identification line in front of update transactions.

Parameter line (optional)

Col	Len	Value	Description
2	1	$	LINE CODE
3	1		UPDATE MODE / SORT ORDER Defines the update or processing mode to be used by ALL userids for this execution the DSMS batch procedure.
		A	NORMAL UPDATE MODE - Transactions sorted in ascending order before any update is applied (i.e entity definitions are processed before sub-screen records.) - Update mode specified for each sign-on record.
		D	DELETE MODE - Transactions sorted in descending order before any update is applied. - All transactions processed as Deletions - Action Code D'. - Sign-on records must specify 'NORMAL' mode - all other modes are considered as errors.

Col	Len	Value	Description
4	1		REPORT FORMAT INDICATOR
		1	SINGLE REPORT FORMAT - One 'END OF REPORT' line is produced. - The transaction 'INPUT NUMBER' is simply incremented by one for each transaction.
		2	SIGN-ON / USERID FORMAT 1 - An 'END OF REPORT' line is produced for each userid / sign-on record. - The transaction 'INPUT NUMBER' is reset to one for each sign-on record. The sign-on record will appear as transaction number one.
		3	SIGN-ON / USERID FORMAT 2 - An 'END OF REPORT' line is produced for each userid / sign-on record. - The transaction 'INPUT NUMBER' is reset to zero for each sign-on record. The sign-on record will appear as transaction number zero.

If the parameter line is not entered, ' $A1' is assumed.

Sign-on line format (required)

Col	Len	Value	Description
1	1		ACTION CODE / UPDATE MODE This field defines the update mode processing to be used for this userid.
		blank	NORMAL UPDATE MODE. - Works like DSMS on-line. - If an Event or Change is created, all following sub-screen transactions will be modified accordingly.
		V	VERSION CONTROL MODE. - All batch transactions will be processed with Action Code 'C' (create). - The external reference fields on Event and Change Definitions will be filled in. - The associated change fields on Event Definitions will be converted to the 'new' Change Number - the number assigned when the Change is created.
		R	REORGANIZATION MODE. - The same as 'V' except that the external reference fields' content will not be altered.
2	1	*	SIGN-ON RECORD CODE
3	8	...	DSMS USER
11	8	...	DSMS uSER PASSWORD
19	3	ppp	PRODUCT CODE to which updates apply.
22	2	ss	SUBSIDIARY CODE to which batch updates apply.
24	1	blank	Unused
25	9		EXTERNAL REFERENCE VALUES The value of the next three fields is used to create Event and Change external references if the update mode is 'V'.
25	4	dddA	- DSMS external Database code
29	3	ppp	- DSMS external Product code
32	2	ss	- DSMS external Subsidiary code

Col	Len	Value	Description
34	1		BLANK LINE AFTER ERROR INDICATOR
		blank	A blank line is printed after each error message on the report.
		N	Blank lines are not printed after error messages on the report.
35	1		REPORT PAGE BREAK INDICATOR
		blank	A new page begins only when the number of lines per page exceeds the maximum number
		T	A page skip for each new type of transaction
		E	A new page for each transaction type of each entity
36	1		TRANSACTION SORT INDICATOR
		blank	The transactions are sorted by type before they are processed.
		N	The transactions are processed in their arrival order.

REPORT

The printout generated by this procedure is an update report, with comments about irregularities or inconsistencies encountered during execution.

RESULT

The result of this procedure is:

. A DSMS database ready for on-line or batch processing,

. A Journal file of the transactions which have modified the database, if 'journalization' was not inhibited during the last restoration.

NOTE: This procedure increments the session number if it is the first access to the database for the current day.

15.3. DESCRIPTION OF STEPS

DUPT: DESCRIPTION OF STEPS

```
UPDATE OF THE DSMS DATABASE:  PDSUP0

.Permanent input-output files
 -Data file
   DA file
 -Va Pac element file
   DC file
 -Cross-reference file
   DX file

.Permanent input file
 -Error message file
   DE file

.Input transaction file
 -Update transactions obtained via
  the DEXP procedure
  MBDUPT file in INPUT directory

.Output file
 -Journal file
   DJ file in JOURNAL directory

.Output report
 -Update review
  in temporary directory

.Return codes
 - 0:  No error
 -08:  Error on the user line or the parameter input line
 -12:  I/O error on a file.
```

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name
$2	Radical of 'input' and 'tmp' files
$3	Suffix of 'input' and 'tmp' directories

15.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DUPT BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "--------------------------------------------------------"
echo "                         DUPT PROCEDURE"
echo "                         =============="
echo "Repertoire 'assign'              : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACDDJ.ini
echo "Directory 'journal'              : `dirname $PACDDJ.`"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "--------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# ************************************************************
# * DSMS : DSMS DATABASE UPDATE
# ************************************************************
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDA.ini
PACDRS=$PACTMP'DUPDRS.BAS'
export PACDRS
echo "Execution : PDSBAS"
cobrun PDSBAS
RETURN=$?
case $RETURN in
 0)
   ;;
 4)
   echo "Error in executing PDSBAS"
   echo "Database unavailable"
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
   exit $RETURN
   ;;
 *)
   echo "Error in executing PDSBAS"
   echo "Error $RETURN"
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
   exit $RETURN
   ;;
esac
# ************************************************************
. $DSMSDIR/assign/$1/PACDDA.ini
. $DSMSDIR/assign/$1/PACDDC.ini
. $DSMSDIR/assign/$1/PACDDX.ini
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDJ.ini
PACDIM=$PACINPUT'MBDUPT'
export PACDIM
PACDRP=$PACTMP'DUPTRP.UP0'
export PACDRP
echo "Execution : PDSUP0"
cobrun PDSUP0
RETURN=$?
case $RETURN in
 0)
```

```
      echo "End of procedure"
      ;;
    *)
      echo "Error in executing PDSUP0"
      echo "Error $RETURN"
      sh $DSMSDIR/batch/proc/ERRPAUSE.ini
      exit $RETURN
      ;;
   esac
   exit $RETURN
```

16. FILE INITIALIZATION (DINI)

16.1. INTRODUCTION

DINI: INTRODUCTION

The DINI procedure initializes the files needed for the installation of a new DSMS database.

It provides an initial backup of the DSMS files, which must be loaded by the Database Restoration (DRST) procedure.

EXECUTION CONDITIONS

None.

However, the parameters of the new DSMS database must have been previously defined, and must be different from the parameters in any other existing DSMS database.

The initial allocation and loading of DSMS components must have been executed (see the Installation Process).

ABNORMAL EXECUTION

Refer to Subchapter 'Abnormal Execution' in Chapter THE BATCH PROCEDURES.

Whatever the cause of the abend, the procedure can be restarted as it is after the problem has been solved.

16.2. INPUT - PROCESSING - RESULTS

USER INPUT

The structure of the input is as follows:

POS.	LEN.	VALUE	MEANING
2	1	'I'	Line code
3	1	'l'	Initial language code (E by default: English)
4	1		This field is ONLY used with DOS/VSE
		'I'	Default option for all hardware
		'N'	If CURRENT-DATE = DD/MM/YY in DOS/VSE

REPORT

This procedure prints a report listing the memorized options and the number of initial records of the DSMS database files.

RESULT

The result is an initial backup including:

- an initial user, whose userid is '********' and whose password is '********' (See the paragraph that follows: INITIAL CONNECTION.)
- a record in the Language Table corresponding to the language code indicated in the user input.

****
* IMPORTANT NOTE *
****

INITIAL CONNECTION:

The Database Restoration (DRST) procedure must be executed after the DINI procedure. After a successful execution of the DRST procedure, the DSMS database is installed.

Verify that the on-line access to the new DSMS database is operational.

The initial connection to the DSMS database is executed as follows:

- Access the DSMS database.

- On the Sign-on screen, enter '********' as the user code and '********' as the password, then press the ENTER key.

- Among the choices listed on the menu, only those marked with a '*' may be accessed. They correspond to the Tables which must be updated for a proper operation of DSMS. The information must be entered in the Tables in the following order:

. In the Languages Table (CHOICE: 'TLA'): the codes and labels of the languages used.

. In the Products Table (CHOICE: 'TPR'): the product codes and labels.

. In the Subsidiaries Table (CHOICE: 'TSU'): the subsidi- ary codes and labels.

. In the User Parameters Tables (CHOICES: 'TUD', 'TUG', 'TUP' and 'TUS'): user codes and authorizations.

(For more details on the management of these tables, see the DSMS Reference Manual).

The '********' user code cannot be deleted: after the User Parameters Tables are updated, the DSMS Database Manager should change passwords in order to prevent the use of this code by others.

16.3. DESCRIPTION OF STEPS

DINI: DESCRIPTION OF STEPS

This procedure includes the following steps:

```
INITIAL DATABASE BACKUP:  PDSINI

.Input transaction file:
 -Initialization transaction
   MBDINI file in temporary directory

.Permanent input file:
 -Error messages

.Output file:
 -Sequential images of files
   BB.NEW file in SAVE directory

.Output file:
 -Backup report
  in temporary directory
```

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

16.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DINI BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "----------------------------------------------------------"
echo "                         DINI PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "----------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# ***************************************************************
# * DSMS : INITIALIZATION OF DSMS FILES
# ***************************************************************
# * INPUT TRANSACTION FORMAT :
# * .INITIALIZATION LINE
# *  COL  2    : I
# *  COL  3    : INITIAL LANGUAGE CODE (E=ENGLISH, F=FRENCH)
# *  COL  4    : SYSTEM DATE FORMAT (I FOR MM/DD/YY)
# *            :                    (N FOR DD/MM/YY)
# ***************************************************************
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACSAVBB.ini
PACDBB=$PACSAVBBNEW
export PACDBB
PACDMB=$PACINPUT'MBDINI'
export PACDMB
PACDRU=$PACTMP'DINIRU.INI'
export PACDRU
echo "Execution : PDSINI"
cobrun PDSINI
RETURN=$?
case $RETURN in
 0)
   echo "End of procedure"
   echo ""
   echo "Call the file BBBACKUP.ini"
   sh $DSMSDIR/assign/$1/BBBACKUP.ini
   ;;
 *)
   echo "Error in executing PDSINI"
   echo "Error $RETURN"
   ;;
esac
if [ "$RETURN" != '0' ]
then
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
fi
exit $RETURN
```

17. JOURNAL EXTRACTION FOR UPDATE (DXBJ)

17.1. INTRODUCTION

DXBJ: INTRODUCTION

The DXBJ procedure extracts, from the DSMS journal file, all the transactions corresponding to a date/time interval or to a given user, and transforms them into update transactions.

EXECUTION CONDITIONS

None.

ABNORMAL EXECUTION

Refer to Chapter THE BATCH PROCEDURES, Subchapter 'Abnormal Execution'.

Whatever the cause of the abend, the procedure can be restarted as it is once the problem has been solved.

17.2. INPUT - PROCESSING - RESULTS

USER INPUT

One '*'-line is required:

Pos.	Len.	Value	Meaning
2	1	'*'	line code
3	8	uuuuuuuu	DSMS User code
11	8	pppppppp	User password
Optional			
19	3	ppp	Product code
22	2	su	Subsidiary code
24	1	'F' or 'E'	Language code
		'	USERS/PASSWORDS IN OUTPUT TRANSAC.

One line per extraction request:

Pos.	Len.	Value	Meaning
2	1	'K'	Line code
3	1	' '	List of selected transactions
		'N'	No list
4	8	CCYYMMDD	Starting date for selection
12	8	CCYYMMDD	Ending date for selection
20	6	HHMMSS	Starting time for selection
26	6	HHMMSS	Ending time for selection
32	8	uuuuuuuu	Selected user code
40	1	' '	User codes present in journal file without password.
		'T'	User codes present in journal file with passwords if sufficient authorization.
		'1'	User code and password, detailed in following columns.
41	8	uuuuuuuu	User code for output transactions (if column 40 = 1)
48	8	mmmmmmmm	Password for output transactions (if column 40 = 1)

REPORT

Extraction report and, upon request, the list of formatted transactions.

RESULT

A DSMS update transactions file to be used as input to the DUPT procedure. An 'N' is entered in column 36 of the user line for DUPT not to sort these transactions. .

17.3. DESCRIPTION OF STEPS

DXBJ: DESCRIPTION OF STEPS

```
TRANSACTION EXTRACTION AND FORMATTING:  PDS700

.Permanent input files:
 -Data file
   DA file
 -Error message file


 -Archived DSMS journal
  BJ file in SAVE directory

.Input transaction file :
 -User transactions
   MBDXBJ file in INPUT directory

.Output file:
 -Update transaction file for DUPT
   MVDXBJ file in INPUT directory

.Output reports:
 -Extraction review
  DXBJRK.700 in temporary directory
 -Transaction printout
  DXBJSK.700 in temporary directory

.Return codes:
 - 0:  No error
 - 8:  Error on the user line or the parameter input line
       The environment definition is missing.
 -12:  File access error
       The technical record is missing.
```

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name
$2	Radical of 'input' and 'tmp' files
$3	Suffix of 'input' and 'tmp' directories

17.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DXBJ BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo ""
echo "----------------------------------------------------------"
echo "                         DXBJ PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "----------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : EXTRACTION OF DSMS JOURNAL FOR DSMS UPDATE
# **************************************************************
# * INPUT TRANSACTION FORMAT:
# * .ONE LINE FOR USER AND DSMS PASSWORD
# *  COL 02    : *
# *  COL 03    : DSMS USER CODE
# *  COL 11    : PASSWORD
# *  COL 19-21 : PRODUCT CODE    (OPTIONAL)
# *  COL 22-23 : SUBSIDIARY CODE (OPTIONAL)
# *  COL 24    : LANGAGE CODE    (OPTIONAL)
# * .EXTRACTION COMMAND LINE
# *  COL 02    : 'K'
# *  COL 03    : ' ' LIST OF SELECTED TRANSACTIONS
# *            : 'N' NO LIST OF SELECTED TRANSACTIONS
# *  COL 04-11 : DATE FOR BEGINING OF SELECTION (SSYYMMDD)
# *  COL 12-19 : DATE FOR END      OF SELECTION (SSYYMMDD)
# *  COL 20-25 : HOUR FOR BEGINING OF SELECTION (HHMMSS)
# *  COL 26-31 : HOUR FOR END      OF SELECTION (HHMMSS)
# *  COL 32-39 : USER CODE
# **************************************************************
. $DSMSDIR/assign/$1/PACDDA.ini
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACSAVBJ.ini
PACDBJ=$PACSAVBJ
export PACDBJ
PACDMB=$PACINPUT'MBDXBJ'
export PACDMB
PACDIM=$PACINPUT'MVDXBJ'
export PACDIM
PACDRK=$PACTMP'DXBJRK.700'
export PACDRK
PACDSK=$PACTMP'DXBJSK.700'
export PACDSK
echo "Execution : PDS700"
cobrun PDS700
RETURN=$?
case $RETURN in
 0)
   echo "End of procedure"
   ;;
 *)
```

```
      echo "Error execution PDS700"
      echo "Error $RETURN"
      ;;
   esac
   if [ "$RETURN" != '0' ]
   then
      sh $DSMSDIR/batch/proc/ERRPAUSE.ini
   fi
   exit $RETURN
```

18. CODE AND KEYWORD UPDATE (DREN)

18.1. INTRODUCTION

DREN: INTRODUCTION

The Code and Keyword Update procedure (DREN) is used to define new codes (table or site) or new keywords to replace those defined and used until then in the tables, thesaurus, and entities.

EXECUTION CONDITION

This procedure works from a sequential backup and/or an archived journal, and must therefore be preceded by a backup and/or an archiving.

ABNORMAL EXECUTION

See Subchapter 'Abnormal Execution', in Chapter THE BATCH PROCEDURES.

Whatever the cause of the abend, the procedure can be restarted as it is once the problem has been solved.

18.2. INPUT - PROCESSING - RESULTS

USER INPUT

One '*' line (required):

Col.	Len.	Value	Meaning
2	1	'*'	Line code
3	8	uuuuuuuu	DSMS User Code
11	8	pppppppp	Password
Optional			
19	3	ppp	Changes made on the entities which depend on the product code 'ppp'
		'***'	Changes made on the entities which depend on all the product codes
22	2	ss	Changes made on the entities which depend on the subsidary code 'ss'
		'**'	Changes made on the entities which depend on all the subsidary codes
24	1	'E' or 'F'	Language code
REQUIRED: AT LEAST ONE OF THESE AREAS SET TO '1'			
25	1	' '	No change concerning the backup
		'1'	Changes concerning the backup
26	1	' '	No change concerning the archiving
		'1'	Changes concerning the archiving

Command lines (500 maxi)

Col.	Len.	Value	Meaning
2	3	'Txx'	table choice (idem TP)
		'Kxx'	keyword choice (with xx = 'T ' for technical keywords, xx = 'E ' for native keywords and xx = 'Cl' for keywords of change l language)
		'S '	site choice
5	13		old code
18	13		new code

NOTES:

- The codes (old and new) must be preceded by 'C', 'E' or 'S' for the TST table, by 'C' or 'E' for the TGR and TTY tables, and by 'F' or 'R' for the TAT table.

- It is not possible to invert two codes (for example, change 'AA' to 'BB', and 'BB' to 'AA'). However, it is possible to rename a code (with an unknown one), and to reuse the old code to transform other codes (for example: 'AA' becomes 'BB' while 'CC' and 'DD' become 'AA'; in this case the command AA/BB must be written before CC/AA and DD/AA).

- The products, subsidaries or sites new codes must not already exist (in the same subsidary for a site).

- The two parts of the site code (9 and 3 characters) cannot be modified separately.

- For the TVE table, it is possible to ask for the following updates:

  - Technical release alone
  - Technical release and release
  - Technical release, release and hardware
  - Technical release, release, hardware and version (with or without language code)
  - Release alone
  - Hardware alone
  - Version number (with or without language code)

  Isolated parts should be aligned as if the other parts were there.

  Ascending consistency checks are performed. The changes requested on the preceding lines must be taken into account.

- The label associated to the new code can either be that of the old code or that of the new code if it already existed. This choice is made while the file is sorted and is therefore impredictable.

- For tables depending on a product (TOP, TPH and TVE), the product's code must be clearly specified on the '*' line.

PRINTED OUTPUT

Report on changes concerning the backup and/or the archiving.

Note on counters:

They count the total number of updates but not the number of modified records (there can be several modifications on the same record).

RESULT

If the change was made on the archive (1 in column 26), a new version of the Journal's sequential backup is produced.

If the change was made on the Database backup (1 in column 25), the result is a new version of the Database sequential backup which should be reorganized via the DREO procedure before being restored.

RETURN CODE

```
+-------------------------------------------------------+
!  0  ! OK                                               !
!  8  ! Error on the user line code or parameter line  !
! 10  ! Invalid absence of save/archiving flag          !
! 12  ! I-O error or inconsistent DSMS database         !
! 16  ! Sort error                                       !
+-------------------------------------------------------+
```

18.3. DESCRIPTION OF STEPS

DREN: DESCRIPTION OF STEPS

This procedure calls a single program (PDSMS) which is used as a flow monitor for various programs considered as sub-routines of this monitor. It includes the following steps:

```
UPDATES:  PDSMS

.Permanent input files:
 -Data file
   DA file
 -Error messages


 -Cross-references
   DX file
 -DSMS backup
   BB file in SAVE directory
 -DSMS archiving
   BJ file in SAVE directory

.Input file:
 -User queries
   MBDREN file in INPUT directory

.Work files:
 -Update requests


 -Partial backup (sorted)

 -Partial backup (not sorted)

.Output files:
 -Modified backup
  PACDB3
 -Modified archive
  PACDJB

.Output reports:
 -Branching report
  PACDIA
 -List of commands on the backup
  PACDIK
 -Update report (backup)
  PACDJK
 -Merging report (backup)
  PACDIS
 -List of commands on archiving
  PACDKK
 -Update report (archive)
  PACDLK

.Sort files:


SYMBOLICS IN USE

+---------+-----------------------------------------------+
!SYMBOLICS! MEANING                                       !
+---------+-----------------------------------------------+
! $1      ! Database name                                 !
! $2      ! Radical of 'input' and 'tmp' files            !
! $3      ! Suffix of 'input' and 'tmp' directories       !
+---------+-----------------------------------------------+
```

18.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--         Release xxx Version xxx         --
#@(#)
#@(#)DSMS (R) DREN BATCH Procedure
#@(#)
# Parameter control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                         PROCEDURE DREN"
echo "                         =============="
echo "Directory 'assign'             : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'              : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files: $2"
fi
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : CODE OR KEYWORD UPDATE
# **************************************************************
# * INPUT TRANSACTION FORMAT:
# * .ONE LINE FOR USER AND DSMS PASSWORD
# *  COL  2    : *
# *  COL  3-10 : DSMS USER CODE
# *  COL 11-18 : PASSWORD
# *              OPTIONAL
# *  COL 19-21 : 'ppp' CHANGES MADE ON THE ENTITIES WHICH DEPEND
# *            :       ON THE PRODUCT CODE 'ppp'
# *            : '***' CHANGES MADE ON THE ENTITIES WHICH DEPEND
# *            :       ON ALL THE PRODUCT CODES
# *  COL 22-23 : 'ss'  CHANGES MADE ON THE ENTITIES WHICH DEPEND
# *            :       ON THE SUBSIDIARY CODE 'ss'
# *            : '**'  CHANGES MADE ON THE ENTITIES WHICH DEPEND
# *            :       ON ALL THE SUBSIDIARY CODES
# *  COL 24    : 'E'   LANGUAGE CODE
# *            : 'F'
# *              MANDATORY (AT LEAST ONE ZONE WITH '1')
# *  COL 25    : ' '   NO CHANGE CONCERNING THE BACKUP
# *            : '1'   CHANGES CONCERNING THE BACKUP
# *  COL 26    : ' '   NO CHANGE CONCERNING THE ARCHIVING
# *            : '1'   CHANGES CONCERNING THE ARCHIVING
# * .COMMAND LINES (500 MAXI)
# *  COL  2-4  : 'Txx' TABLE CHOICE (IDEM TP)
# *            : 'Kxx' KEYWORD CHOICE (WITH xx = 'T ' FOR
# *            :       TECHNICAL KEYWORDS, xx = 'E ' FOR NATIVE
# *            :       KEYWORDS AND xx = 'Cl' FOR KEYWORDS OF
# *            :       CHANGE l LANGUAGE
# *            : 'S  ' SITE CHOICE
# *  COL  6-18 : OLD CODE
# *  COL 19-31 : NEW CODE
# *
# *  NOTE : THE CODES (OLD AND NEW) MUST BE PRECEDED BY 'C', 'E'
# *         OR 'S' FOR THE TST TABLE, BY 'C' OR 'E' FOR THE TGR
# *         TABLE AND BY 'F' OR 'R' FOR THE TAT TABLE.
# *
# **************************************************************
. $DSMSDIR/assign/$1/PACDDA.ini
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDX.ini
. $DSMSDIR/assign/$1/PACSAVBB.ini
PACDBB=$PACSAVBB
export PACDBB
. $DSMSDIR/assign/$1/PACSAVBJ.ini
PACDBJ=$PACSAVBJ
export PACDBJ
PACDMB=$PACINPUT'MBDREN'
export PACDMB
PACDW0=$PACTMP'W0'
export PACDW0
PACDW1=$PACTMP'W1'
export PACDW1
PACDW2=$PACTMP'W2'
export PACDW2
PACDB3=$PACSAVBBNEW
export PACDB3
PACDJB=$PACSAVBJNEW
export PACDJB
PACDIA=$PACTMP'DRENIA.MS'
export PACDIA
PACDIK=$PACTMP'DRENIK.MS'
export PACDIK
PACDJK=$PACTMP'DRENJK.MS'
export PACDJK
PACDIS=$PACTMP'DRENIS.MS'
export PACDIS
PACDKK=$PACTMP'DRENKK.MS'
export PACDKK
PACDLK=$PACTMP'DRENLK.MS'
export PACDLK
echo "Execution : PDSMSE"
cobrun PDSMSE
RETURN=$?
case $RETURN in
0)
 ;;
8)
 echo "Error in executing PDSMSE"
 echo "Error 8 : error on '*' line or on command line"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
10)
 echo "Error in executing PDSMSE"
 echo "Error 10 : database not saved or not archived"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
12)
 echo "Error in executing PDSMSE"
 echo "Error 12 : input/output error or inconsistent DSMS base"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
*)
 echo "Error in executing PDSMSE"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# ***************************************************************
echo "End of procedure"
if [ -f "$PACSAVBJNEW" ]
then
   echo ""
   echo "Calling the file BJBACKUP.CMD"
   . $DSMSDIR/assign/$1/BJBACKUP.ini
fi
if [ -f "$PACSAVBBNEW" ]
then
   echo ""
   echo "Calling the file BBBACKUP.CMD"
   . $DSMSDIR/assign/$1/BBBACKUP.ini
```

```
      echo ""
      echo "BB file must be reorganized"
   fi
   echo ""
   echo "Deletion of the temporary files"
   rm -f  $PACTMP'W*'
   exit $RETURN
```

19. PRE-PROCESSING OF GENERATED DAF PROGRAMS (DPDF)

19.1. INTRODUCTION

DPDF: DAF PRE-PROCESSOR FOR GENERATED PROGRAMS

The DPDF procedure processes user generated programs that contain SQL requests for Database access through DAF operators.

EXECUTION CONDITION

None.

IMPLEMENTATION

The DPDF procedure may be executed in several ways:

- Either after a program generation via GPRT, whose generated output is used as input for DPDF, before being passed on for compilation or storing in a source-program library.

- Or by a call in the optional generated program before/after control cards. In this case, the correct JCL must have been entered in the selected options, which are updated by the user-parameter update transaction or the PARM batch procedure.

19.2. INPUT - PROCESSING - RESULTS

USER INPUT

It is the COBOL source of the programs containing DAF operators which must be solved by the pre-processor before being compiled.

Each program contains, after the IDENTIFICATION DIVISION line, a command line for the pre-processor:

Pos.	Len.	Value	Meaning
1	6	nnnnnn	COBOL line number
7	1	'*'	Comments
8	5	'TP '	On-line program, or
		'BATCH'	Batch program
13	6	'LIB:'	Fixed label
19	3	bbb	Library code
22	1	blank	Not used
23	5	nnnns	Session number - Session status
28	1	blank	Not used
29	2	--	Generation variant(s)
31	5	'AR:'	Fixed label
36	1	l	Database language code
37	5	'SC:'	Batch language program skeleton
		'SG:'	OLSD program skeleton
		'SR:'	COBOL Generator program skeleton
42	1	l	Skeleton language
43	1	blank	Not used
44	6	'SINGLE'	Single quotes, or
		'DOUBLE'	Double quotes

```
Examples:

000020*TP    LIB: APP 2345  00 AR: F SG: F SINGLE

000020*BATCH LIB: APP 2300T 4  AR: F SC: F DOUBLE
```

This line is automatically generated by the GPRT procedure.

PRINTED OUTPUT

The procedure prints the list of errors, if any.

RESULT

The result of the execution is a COBOL source file in which all DAF operators have been solved, and all the calls to Database batch or on-line access routines have been generated.

19.3. DESCRIPTION OF STEPS

DPDF: DESCRIPTION OF STEPS

The DPDF procedure calls a single program which acts as a flow monitor for various programs, considered as sub-programs of this monitor. It includes the following step:

```
GENERATED PROGRAM'S PRE-PROCESSOR:  DAFD10

.Permanent input files:
 -Data file
  DA file
 -Error message file
  DE file

.Input file:
 -Generated programs
  DAF : cobol source in input in temporary directory

.Output files:
 -Generated programs to be compiled
  COB : precompiled cobol in temporary directory

.Output reports:
 -Execution report
  in temporary directory
```

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name
$2	Radical of 'input' and 'tmp' files
$3	Suffix of 'input' and 'tmp' directories

19.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--         Release xxx Version xxx         --
#@(#)
#@(#)DSMS (R) DPDF BATCH PROCEDURE
#@(#)
# Parameter control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                        DPDF PROCEDURE"
echo "                        =============="
echo "Directory 'assign'                : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                   : `dirname $PACTMP.`"
echo "  (DAF = input COBOL; COB = pre-compilation COBOL)"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                 : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files   : $2"
fi
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS ACCESS FACILITY PRE-PROCESSING
# **************************************************************
# * COMMAND LINE FOR PREPROCESSOR
# **************************************************************
# * COL 1-5   : COBOL LINE NUMBER
# * COL 7     : '*'  COMMENT
# * COL 8-12  : 'TP   ' OR 'BATCH'
# * COL 14-17 : 'LIB:'
# * COL 19-21 : LIBRARY CODE
# * COL 23-27 : SESSION NUMBER AND VERSION
# * COL 29-30 : GENERATION VARIANT
# * COL 32-34 : 'AR:'
# * COL 36    : DATABASE LANGUAGE CODE
# * COL 38-40 : 'SC:' BATCH PROGRAM SKELETON
# *           : 'SG:' ON-LINE PROGRAM SKELETON
# *           : 'SR:' COBOL PROGRAM SKELETON
# * COL 42    : SKELETON LANGUAGE CODE
# * COL 44-49 : 'SINGLE' OR 'DOUBLE'  (QUOTES)
# **************************************************************
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDA.ini
DAF80=$PACTMP'DAF'
export DAF80
COB80=$PACTMP'COB'
export COB80
DAFREP=$PACTMP'DAFREP'
export DAFREP
echo "Execution: DAFD10"
cobrun DAFD10
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing DAFD10"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# **************************************************************
```

```
echo "End of procedure"
exit $RETURN
```

20. BATCH UPDATE FROM DAF TABLES (DUPD)

20.1. INTRODUCTION

DUPD: INTRODUCTION

The DUPD procedure performs a batch update on the DSMS Database from a sequential file mirroring the DAF tables.

Its operating principle is quite similar to that of the DUPT procedure, except for the format of the input transactions.

EXECUTION CONDITION

Refer to the chapter dedicated to DUPT.

ABNORMAL EXECUTION

Refer to the chapter dedicated to DUPT.

20.2. INPUT - PROCESSING - RESULT

DUPD: INPUT-PROCESSING-RESULTS

USER INPUT

The sequential file of input transactions is produced by a DAF extractor program. Its records mirror the DAF tables (described in the DAF TABLES Manual).

Pos.	Length	Meaning
1	1	Transaction code (C, M, X, D or A, B)
2	10	DAF table code
12	299	DAF table contents (described in the DAF tables Manual).

UPDATE RULES

Update transactions are not sorted.

Each set of transactions impacting a library or session must be preceded by an ASSIGN table code line.

Pos.	Length	Value	Meaning
2	10	'ASSIGN'	Table code
12	8	uuuuuuuu	User code
20	8	pppppppp	Password
28	3	ppp	Product code
31	2	ss	Subsidiary code

PRINTED OUTPUT

Refer to the description of the DUPT output.

RESULT

Refer to the description of the DUPT result.

20.3. DESCRIPTION OF STEPS

DUPD: DESCRIPTION OF STEPS

TRANSACTION FORMATTING: PDS900

.Input transaction file:
 -Update transactions

.Output files:
 -Formatted transactions

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name
$2	Radical of 'input' and 'tmp' files
$3	Suffix of 'input' and 'tmp' directories

20.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DUPD BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "----------------------------------------------------------"
echo "                         DUPD PROCEDURE"
echo "                         =============="
echo "Repertoire 'assign'              : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACDDJ.ini
echo "Directory 'journal'              : `dirname $PACDDJ.`"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "----------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# ************************************************************
# * DSMS : DSMS DATABASE UPDATE
# ************************************************************
. $DSMSDIR/assign/$1/PACDDE.ini
. $DSMSDIR/assign/$1/PACDDA.ini
PACDRS=$PACTMP'DUPDRS.BAS'
export PACDRS
echo "Execution : PDSBAS"
cobrun PDSBAS
RETURN=$?
case $RETURN in
 0)
   ;;
 4)
   echo "Error in executing PDSBAS"
   echo "Database unavailable"
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
   exit $RETURN
   ;;
 *)
   echo "Error in executing PDSBAS"
   echo "Error $RETURN"
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
   exit $RETURN
   ;;
esac
# ************************************************************
PACDGY=$PACINPUT'MBDUPD'
export PACDGY
PACDIM=$PACTMP'IM'
export PACDIM
echo "Execution : PDS900"
cobrun PDS900
RETURN=$?
case $RETURN in
 0)
   ;;
 *)
   echo "Error in executing PDS900"
   echo "Error $RETURN"
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
```

```
      exit $RETURN
      ;;
   esac
   # *************************************************************
   . $DSMSDIR/assign/$1/PACDDA.ini
   . $DSMSDIR/assign/$1/PACDDC.ini
   . $DSMSDIR/assign/$1/PACDDX.ini
   . $DSMSDIR/assign/$1/PACDDE.ini
   . $DSMSDIR/assign/$1/PACDDJ.ini
   PACDIM=$PACTMP'IM'
   export PACDIM
   PACDRP=$PACTMP'DUPDRP.UP0'
   export PACDRP
   echo "Execution : PDSUP0"
   cobrun PDSUP0
   RETURN=$?
   case $RETURN in
    0)
      echo "End of procedure"
      ;;
    *)
      echo "Error in executing PDSUP0"
      echo "Error $RETURN"
      sh $DSMSDIR/batch/proc/ERRPAUSE.ini
      exit $RETURN
      ;;
   esac
   exit $RETURN
```

21. INSTALLATION

21.1. INTRODUCTION

INTRODUCTION

The UNIX version of DSMS is delivered in two packages:

1. The P.A.W. workstations (Windows or Win-OS/2), supplied on a CDROM.

   This installation is described in the P.A.W DEVELOPER'S GUIDE, Chapter 'Revamping of IBM Products'.

   See also Subchapter 'Installation of Rewamped Workstations PAW)' for additional information.

2. A package supplied on a magnetic medium or a CD-ROM which contains the following elements :

   . the server part of DSMS,
   . the 'dumb terminal' user workstations operating under UNIX (DSWST),
   . the PaclinK-emulated workstations (Windows, Win-OS/2).

The installation of these elements is described in this chapter.

This package includes the following files (described in the next two pages):

. DSMSINST.Vnn,

'Vnn' represents the installed version number
. DSMSBASE.3PC, DSMSBASE.3IS, DSMSBASE.4PC and DSMSBASE.4IS for AIX,

DSMSBASE.750, DSMSBASE.815 and DSMSBASE.875 for HPUX,
DSMSBASE.ISO for other hardware,
. README_E.Vnn and README_F.Vnn,
. AMELIO.Vnn and CHANGES.Vnn,
. and if relevant, a DSMSPATCH directory which contains the patch files which might be required in case of re-installation.

NOTE: Depending on the type of medium and the UNIX system, the name of the installation files can be written in upper or lower cases.

Description of DSMSINST.Vnn :

The DSMSINST.Vnn file contains the DSMS installation or re-installation commands .

These commands are written in shell language.

Description of DSMSBASE.xxx :

The DSMSBASE.xxx files are compressed in the tar format. They contain the DSMS execution and operating files in French and in English.

On RS6000 or DPX20, the operating system allows you to choose between character sequences 'pc850' (set of characters used on OS/2) and 'iso8859' (set of characters used on WINDOWS).
For compatibility purposes, you benefit from both installation options when installing the system for the first time and when re-installing it.

On HP9000, the software can be used with the Run-Time COBOL Micro Focus 3.0 (HP 7.50), 3.1 (HP 8.15) or 3.2 (HP 8.75 ou HP 9.xx).
On RS6000 or DPX/20, it can be used with Run-Time COBOL Micro Focus 3.2 ou 4.0.
During installation or re-installation, the Run-Time choice allows to install files which are compatible with the Run-Time used.

Description of README_F.Vnn and README_E.Vnn

The README currently contains information on the evolution of the software.

It is recommended to consult this file before any installation or re-installation operation.

The README_F.Vnn is directed at French users and the README_E.Vnn at English users.

Description of AMELIO.Vnn and CHANGES.Vnn

These files contains the list of changes brought by the patch.

The AMELIO.Vnn file is directed at French users and the CHANGES.Vnn file at the English users.

Description of the 'DSMSPATCH' directory:

The DSMSPATCH contains patches, when applicable. Each file's name is a patch number. The file contains :

- the list of changes brought by the patch,
- the name and access path of the program(s) affected by the patch,
- all the releases that can use the patch.

These files are used when a re-installation is done by DSMSINST.Vnn.

INSTALLATION PROCESS

A general system backup should be performed prior to proceeding with the installation of DSMS. Then, the following steps should be carefully carried out:

- Creation of a DSMS login

- Creation or update of the UNIX environment variables

- Unloading of the magnetic media in the $HOME connection directory

- loading of the CD-ROM in the 'release_code'. 'hardware_code' subdirectory in the loading directory,

- Submission of the dsmsinstall procedure,

- Creation of the batch and on-line servers,

- Creation of the users on the LAN if necessary,

- Installation of the workstations,

- Test of the installation.

WARNING

DSMS MUST be installed on DISKS WHICH ARE PHYSICALLY INSIDE the UNIX computer, NOT on NFS-mounted disks.

The present DSMS release requires that the VA Pac Database should be formatted to the 8.0.2 release (whatever the sub-release) or to an earlier release.

21.2. DSMS INSTALLATION

DSMS INSTALLATION

CREATION OF A SPECIFIC UNIX USER

The installation of most UNIX software requires the creation of a specific user account.

This allows the administrator to isolate, for security reasons, the different software which might be installed on a computer.

DO NOT INSTALL UNDER 'root' UNIX ACCOUNT.

The UNIX Administrator (root) should therefore create a UNIX login under which DSMS will be installed. This login will be the Administrator's login. If a VA Pac Database was already installed, the same login may be used.

MODIFICATION OF THE CONFIGURATION FILE

If the DSMS Database must monitor a VA Pac Database, and if this VA Pac Database has not been installed yet, you will have to install it before the DSMS Database.

Log in under the DSMS administrator login, and update the shell configuration files (.profile, .kshrc or .login according to the case).

The environment variables COBDIR, DSMSDIR, COBPATH, and PATH will be created or modified:

```
COBDIR=<path to COBOL Run-Time directory>
export COBDIR

DSMSDIR=$HOME/dsmsx
export DSMSDIR

COBPATH='$DSMSDIR/tp/gnt:$DSMSDIR/batch/gnt'
export COBPATH

PATH='$DSMSDIR/bin:$DSMSDIR/batch/proc:$COBDIR/bin:$PATH'
export PATH
```

For DIGITAL, SUN or TANDEM releases, add the following variable:

```
     LD_LIBRARY_PATH='$COBDIR/coblib'
     export LD_LIBRARY_PATH
```

For the RS6000, create or modify the LIBPATH variable:

```
     LIBPATH=$COBDIR/coblib
     export LIBPATH
```

The COBOL run-time directory is the one which contains the executable programs and libraries necessary for the execution of COBOL programs.

Usually, COBDIR is set to the /usr/lib/cobol value. For further details, refer to your COBOL Compiler's Installation Manual and to other COBOL manuals describing the COBDIR environment variable.

After this operation, start up the configuration initialization again by typing '. .profile' and check with the UNIX command 'set' that the modifications have been taken into account in the configuration files. If the environment variables are not correctly initialized, log off then log on again with the DSMS administrator login.

LOADING THE MAGNETIC MEDIUM

When DSMS is shipped on a magnetic medium, the content of the medium has to be loaded on the UNIX machine, in the administrator $HOME directory.

This is done via the following commands:

```
   cd $HOME
   tar xvf /dev/...
```

('/dev/...' being the path to this support reading unit).

LOADING THE CDROM

Load the driver of the CDROM reader in a system directory,

Connect, in the CDROM 'Loading' directory, in the subdirectory:

```
   ...'version-code'.'hardware_code'
```

Example on RS6000 :

```
cd/cdrom/ID200V00.RS6
```

SUBMISSION OF THE INSTALLATION PROCEDURE:

WARNING

To ensure a correct installation of the software, check that the disk space is equal to five times the DSMSBASE.xxx file (or is twice the DSMSBASE.xxx file) in the installation directory.

Once the files have been copied onto the hard disk drive, type the following commands according to your language:

- 'sh dsmsinst* -e' for an installation in English,

- 'sh dsmsinst* -f' for an installation in French,

and press the Enter or Return key.

Description of the installation procedure steps:

- Consistency check on the environment variables

  (.profile, .kshrc, .login),

- Display of 'readme' file contents,

- Display of the installation menu,

- Selection of a character set (RS6000 or DPX20 only),

- Selection of the COBOL run-time used (HP9000, RS6000 or

  DPX20 only),

- Creation of the installation directory,

- Distribution of files issued from DSMSBASExxx,

- Test Database creation and naming,

- Specification of the DC file access path,

- Execution of the test Database initialization batch procedures,

- Creation of the 'journal_dsmsinstall' file.

Consistency check of environment variables:

Before running dsmsinst, the DSMS administrator must check the modifications made in the configuration file(s), even though the dsmsinst procedure performs some consistency checks.

Display of readme contents:

To scroll through the readme file, use the space bar.

At the end of the display, you can:

- Either re-run this step,

- Quit the dsmsinst procedure,

- Or go to the next step.

Display of the installation menu:

The installation menu looks like this:

```
************************************************************
                           DSMS
************************************************************

 readme           :  display of the readme file
 inst             :  installation of this version

************************************************************

Type the command name, or 'x' to quit dsmsinst:
```

Description of commands:

- The 'readme' command edits the readme file for reading.

- The 'inst' command starts the installation process. Type the command and follow the 'dsmsinst' instructions.

Choice of a character set (RS6000 - DPX20))

Depending on the machine's configuration, type 'pc850' or 'iso8859'. If you do not know what configuration your machine is in, ask your UNIX administrator.

Choice of COBOL run-time used (HP9000, RS6000 or DPX20):

On HP9000, the COBOL run-time can be '750' if you are using a 3.0 Micro Focus COBOL run-time (HP release 7.50) or '815' if you are using a 3.1 Micro Focus COBOL run-time (HP release 8.15). (consult the UNIX administrator about the run-time)

On RS6000 or DPX20, the COBOL run-time can be '3.2' if you are using a 3.2 Micro Focus COBOL run-time or '4.0' if you are using a 4.x. Micro Focus COBOL run-time. (consult the UNIX administrator about the run-time)

Creation of the installation directory:

Dsmsinst imposes the name (dsmsx) to the installation directory for the two following reasons:

- to isolate the DSMS software from any other software installed on the computer, and to monitor changes to it,

- to allow dsmsinst to monitor the type of processing (installation or re-installation) to be done.

Distribution of files:

During this step dsmsinst extracts the group of DSMSBASExxx files according to the selected language.
It also creates the directories described further in this manual (paragraph 'Description of created Directories').

The -e or -f installation option allows you to benefit from the following features in English or in French when running DSMS:

- Error messages
- Batch procedures

in the language chosen at installation.

Test Database creation and naming:

The test Database must be given a name at the time of installation. You enter this name in an interactive manner, by answering questions about the Database (4 alphanumeric characters maximum).

This name allows you to create the Database's specific sub-directories:

```
- $DSMSDIR/assign/'db_name'
- $DSMSDIR/bases/'db_name'
- $DSMSDIR/input/'db_name'
- $DSMSDIR/journal/'db_name'
- $DSMSDIR/save/'db_name'
- $DSMSDIR/tmp/'db_name'
```

Selection of the DC file access path

For sites which operate a VA Pac Database, it allows to assign the DC file in this Database (currently in $PACDIR/bases)

Execution of the batch initialization procedures

The following batch procedure initializes the test Database:

. DRST: Loading of the test database.

NOTE: The Micro Focus COBOL run-time is required to run the batch procedures, as well as to start the batch and on-line servers.

Update of the journal_dsmsinstall file:

The journal_dsmsinstall file contains some information resulting from the use of dsmsinst. After an installation, it reads the following information:

- Installation date,
- Release number
- Version number
- Name of the character set used
- Test Database creation date.

21.3. COMPLEMENT: INSTALLATION OF DAF ENVIRONMENT

COMPLEMENT : INSTALLATION OF DAF ENVIRONMENT

The DSMS ACCESS FACILITY (DAF) implies the translation of SQL access queries to the DSMS database, written in user programs, by the generation of data and calls of sub-programs in the generated COBOL source of these programs.

The pre-processor processes the programs generated by VA Pac to make this translation.

The pre-processor is made up of a program, DAFD10.gnt, installed in the batch programs directory.

The user can use the DPDF procedure to process the generated programs, using DAF (see chapter dedicated to the DPDF procedure).

The work file required to operate DAF is described in paragraph 'DAF TP Work file' in subchapter 'The User Files', chapter THE DSMS COMPONENTS.

EXTRACTION SUB-PROGRAMS

For user programs generated with a variant 3 (adaptation to COBOL Micro Focus), the same extraction sub-programs are used for batch and on-line programs. These extractors are compiled linked when delivered (.gnt files) and as COBOL source (.cbl files).

The extraction programs are:

. PDSBDF and PDTPDF, called by the user DAF programs.

. PDSDAC and PDSFAC, called by the extractors to access the DSMS Database and the DAF work file.

The extractors compiled files are delivered in the directory of the VA Pac servers' batch programs ($DSMSDIR/batch/gnt).

The COBOL source files are delivered in the SMSDIR/daf directory. These subprograms must be compiled on the site when the version of the site Micro Focus compiler is different from that used for DSMS.
The versions of the Micro Focus compilator used are the following:

. "3.0", "3.1" ou "4.0.5" for HP9000, depending on the installation,

. "3.1" for TANDEM,

. "4.1" for OSF1,

. "4.0.20" for SUN,

. "3.2.50" or 4.0.20 for DPX/20, RS6000, depending on the installation.

DAF DICTIONARY

Data Element, Data Structure and Segment entities can be used to write programs calling DAF facility. These entities are delivered as batch transactions in the MBUPDT.DAF file copied in $DSMSDIR/method at installation.

The insertion of the 'DAF dictionary' in VisualAge Pacbase via the UPDT batch update procedure is under the responsability of the Database Manager who must first:

. check the compatibilty of entity codes with the entities already existing on the site,

. duplicate the $DSMSDIR/method/MBUPDT.DAF file in MBUPDT, under $PACDIR/input/"db_name",

. in $PACDIR/input/"db_name", modify the user line "*" of the MBUPDT file.

In order to avoid problems of compatibility between the site dictionary and the entities provided for the DAF facility, it is recommended to create a network independent of libraries for the writing of the site DAF utilities.

COMPILING AND RUNNING DAF PROGRAMS

The $DSMSDIR/daf directory contains an example of compilation and running scripts for a DAF program ('dafcomp' and 'dafrun').

It is advisable to duplicate the directory in a working directory, for example dafuser, via the command:

```
cp -r $DSMSDIR/daf /dafuser
```

And then copy the DAF generated programs in the working directory.

Go to the working directory to modify and launch the compilation and running scripts.

The compilation script may be used to compile DAF sub-programs.

The compilation script may be used to compile the sub-programs of the DAF system.

CAUTION: the sources of generated DAF programs must not be modified under the editor after the operation of the pre-processor because the editor removes the 'low values' generated by the pre-processor.

EXECUTION OF DAF EXTRACTOR

The assignment of the following files must precede the execution of the DAF extractor:

```
. Permanent input files:

  - DSMS Data file                   : PACDDA
  - VA Pac elements file             : PACDDC
  - Cross-references files           : PACDDX
  - Error message file               : PACDDE

. DAF work file                      : SYSDAF

. User files if relevant.
```

21.4. DESCRIPTION OF CREATED DIRECTORIES

DESCRIPTION OF CREATED DIRECTORIES

The installation of DSMS creates a tree structure under the '$DSMSDIR' root.

```
'$DSMSDIR'
    !               +---- batch_server
    !               !
    !               +---- 'Base1'
    !               !
    !               !---- 'Base2'
    !               !
    +---assign   ---+---- ......
    !               !
    !               +---- monitors
    !               !
    !               +---- tp_server
    !
    !               +---- 'Base1'
    !               !
    +---bases    ---+---- 'Base2'
    !               !
    !               +----  ...
    !
    !               +----  gnt
    !               !
    +---batch    ---+----  int
    !               !
    !               +----  proc
    !
    +---bin
    !
    +---commun
    !
    +---daf
    !
    +---input
    !
    +---install
    !
    !
```

```
.../...
   !
   !               +---- 'Base1'
   !               !
   +---journal  ---+---- 'Base2'
   !               !
   !               +----  ...
   !
   +---log
   !
   +---method
   !
   +---paclink
   !
   !               +---- 'Base1'
   !               !
   +---save     ---+---- 'Base2'
   !               !
   !               +----  ...
   !
   !               +---- 'Base1'
   !               !
   +---tmp      ---+---- 'Base2'
   !               !
   !               +----  ...
   !
   !               +---- gnt
   !               !
   +---tp       ---+---- int
   !               !
   !               +---- save
   !
   !               +---- 'User1'
   !               !
   +---users    ---+---- 'User2'
   !               !
   !               +---- ...
   !
   !---zardlg
```

SYSTEM DIRECTORIES

The $DSMSDIR/bin directory

The 'bin' directory contains all C executable files: batch monitors (dsbatch), on-line monitors (dstp), the UNIX workstation (dswstsock), as well as some conversion utilities.

The $DSMSDIR/tp directory

The 'tp' directory contains the COBOL executables ('int' and 'gnt' directories), as well as the transaction backup files ('save' directory), which allow for the operation of the on-line server(s).

```
tp ----+--- gnt   - on-line programs (compiled code)
       !
       +--- int   - on-line programs (interpreted code)
       !
       +--- save  - 'server1'.ts (transaction back-up)
       !          - 'server2'.ts (transaction back-up)
                            .
                            .
```

The $DSMSDIR/batch directory

The 'batch' directory contains the COBOL executable programs ('int' and 'gnt' directories) and the DSMS shell operation commands ('proc' directory).

```
   batch -+--- gnt   - Batch programs (compiled code)
          !
          !--- int   - Batch programs (interpreted code)
          !
          +--- proc  - batch procedures
```

Batch procedures are described in this manual.

The $DSMSDIR/install directory

This directory contains the dsmsadmin procedure, which allows for the creation of a new Database and the creation or deletion of servers, the files that initialize new Databases and servers, and the journal_dsmsinstall file which records all the installation operations performed.

The $DSMSDIR/method directory

This directory contains all transaction files used in installation complement:

- DAF files (MBUPDT.DAF),

When being used, these files are copied in the ACDIR/input/"db_name" directory of the VA Pac installation.

DIRECTORIES OF THE SPECIFICATION DATABASES

The $DSMSDIR/assign/'db_name' directory

The 'assign' directory contains one subdirectory for each installed Database. It is home to all the command files that contain the assignments of files installed as a standard in the '$DSMSDIR/bases' and $DSMSDIR/bases/'db_name' directories. These command files are called PACDDA.ini, PACDDE.ini, etc. for the Database files.

The 'assign' directory' also contains 3 subdirectories:

- batch_server
- tp_server
- monitors

The 'batch_server' and 'tp_server' directories contain the initialization files of the server(s). The 'monitors' directory contains the initialization files of the monitors.

The $DSMSDIR/bases directory

This directory contains the DE error message file that can be used by all installed DSMS Databases.

The $DSMSDIR/bases/'db_name' directory

This directory has one subdirectory for each installed Database. These subdirectories contain the Database's indexed files (DA and DX). The DC file of VA Pac elements is not located in this directory; it is installed in the $PACDIR/bases directory of the VA Pac installation (can monitor several VA Pac Databases).

The $DSMSDIR/daf directory

This directory contains examples of compilation and execution scripts for DAF programs.

The $DSMSDIR/journal/'db_name' directory

This directory contains one subdirectory for each installed Database. These subdirectories contain the DJ journal file of the DSMS Database.

The $DSMSDIR/save/'db_name' directory

The 'save' directory contains one subdirectory for each installed Database; these subdirectories contain the sequential Database backups (BB and BJ).

The 'save' directory also contains the command files that ensure a shift on the two versions of the Database's sequential backups (BBBACKUP.ini and BJBACKUP.ini files).

The $DSMSDIR/input/'db_name' directory

The 'input' directory contains one subdirectory for each installed Database; these subdirectories contain the input transaction files for the MBxxxx batch procedures (MBDRST, MBDREO, MBDPRT,...) as well as the transaction files created by the extraction utilities, coded MVxxxx (MVDEXH, MVDEXP,...).

The $DSMSDIR/tmp/'db_name' directory

The tmp directory contains one subdirectory for each installed Database. It contains the temporary files, as well as the batch procedures' execution reports.

The $DSMSDIR/log directory

The 'log' directory contains one 'trace' file per server if the Trace function has been activated on the on-line monitor.

SHARED DIRECTORIES

They are common to the whole DSMS installation, and they are created by the dsmsinst procedure.

The $DSMSDIR/commun directory

The 'commun' directory contains communication files: DLB and DBD for batch servers; DCD and DCM for each on-line server.

The DLB file memorizes the contents of the batch requests, and the DBD file shows the status of these requests.

These files are prefixed by the name of the specific Database.

The DCM file contains the synchronization messages of the on-line server and the workstations.

The DCD file contains the screen information exchanged between the workstations and the on-line server.

These files are prefixed by the name of the specific server.

The $DSMSDIR/users directory

The 'users' directory groups the results of generation-print requests submitted on-line (LVQ screen). This information is distributed among the various (n) subdirectories, each corresponding to a different user.

```
example:   users  +---  JOHN  --- nnnnn.ia
                  !               nnnnn.iq
                  !               etc.
                  !---  PAUL  --- ppppp.ia
                                  ppppp.iq
                                  etc.
```

where nnnnn and ppppp are the job numbers (see batch monitor screen) and 'ia' and 'iq' are examples of files created by the batch server.

THE DIRECTORY OF PACLINK-EMULATED WORKSTATIONS

In the $DSMSDIR/Paclink installation directory, the following files are created:

```
. PACLINK.EXE:     user workstation monitor

. GSTCPIP.EXE:     communications manager via the Windows
                   Socket interface

. PACLINK.PRM:     workstation monitor's parameter file
                   xx = IR for TANDEM   (IRIX)

. GSTCPIP.PRM:     communications manager's parameter file,
                   communication via the Windows Socket.

. GSHPUNIX.TAB (for HPUX), GSAS850.TAB (for AIX/PC850),
  GSANSI.TAB (for AIX/ISO8859, SunOs, IRIX or OSFI):
                   character transcoding table

. VAPAC.FON:       character font used for the screen trace
                   activated in GSPACLAN.EXE.
```

21.5. CREATION/DELETION OF DSMS ON-LINE/BATCH SERVERS

CREATION AND DELETION OF DSMS ON-LINE AND BATCH SERVERS

The 'dsmsinsadmin' procedure, in $DSMSDIR/install, offers a menu for the creation and deletion of on-line and batch servers.

Creation of an on-line server

Before creating an on-line server, you must first update the UNIX system files. Perform this update BEFORE executing the dsmsadmin procedure.

Update of the system files:

- Even if you do not actually use it, you must declare the socket interface to the system. For this, you need to assign a communication port number in the UNIX '/etc/services' file.

- Choose a communication port number for each on-line server, among the available ports.
  Some port numbers are reserved for standard Internet network applications or for future applications. Others are dynamically allocated. It is then necessary to choose, in '/etc/services'a port number which is not reserved and which is not currently used, that is to say, which is dynamically allocated.

Implementation example:

For each on-line server, a port number must be chosen first;

Let 1500 be this port number:

- The 1500/tcp port number must not be declared in the '/etc/services' file.

- It must not be currently used (i.e. dynamically allocated). To check this, type the following UNIX command:

```
netstat -a \ grep 1500
```

- If both conditions are met, the UNIX administrator (root) must update the '/etc/services' file, by typing the following UNIX command:

```
dsmsx 1500/tcp
```

Make a note of the chosen port number since it will be required by the dsmsinstall procedure when creating the on-line server.

Submission of the 'dsmsadmin' procedure

To activate the dsmsadmin procedure, go to the $DSMSDIR/install directory by typing the following command:

```
cd $DSMSDIR/install
```

Then, type one of the following commands:

```
sh dsmsadmin -e  for an English-language installation

sh dsmsadmin -f  for an French-language installation
```

and press the Return or Enter key.

Description of steps:

The server creation procedure includes the following steps:

- Display of the menu (see below)
- Typing the command,
- Typing the Database name,(if creating)
- Typing the server's name,
- Specifying the number of workstations (if creating an on-line server)
- Specifying the communication port number (if creating an on-line server)
- Updating the journal_dsmsinstall file

Menu display:

The server creation or deletion menu looks like this:

```
************************************************************

                            DSMS

************************************************************

 cresertp       :  creation of an on-line server
 creserbatch    :  creation of a batch server
 delsertp       :  deletion of an on-line server
 delserbatch    :  deletion of a batch server
    .
    .

************************************************************

 Type the command name or 'x' to exit dsmsadmin.
```

Execute the command and follow the instructions given by the dsmsadmin procedure itself.

Update of the journal_dsmsinstall file

The following information is added to the journal_dsmsinstall file:

- Server name
- Creation/deletion date

21.6. CREATION OF NETWORK RESOURCES

CREATION OF NETWORK RESOURCES

COMMUNICATION VIA THE SOCKET INTERFACE

The socket communication interface is based on the TCP/IP network protocol. Besides the installation itself, the implementation of TCP/IP requires some essential files to be configured.

The 'hosts' file of each machine must contain the IP addresses of the machines with which communication is established.

The 'ping' command allows you to check whether the TCP/IP network is in operating order.

Example : ping "host_name"

If it is the case, the PC workstations and the DSMS on-line server are enabled to communicate via the socket interface.

To access the generation-print files, you must define some specific NFS resources or use FTP.

NFS and FTP are standard TCP/IP products for UNIX systems and are currently available on TCP/IP products for PC.

When using NFS, the principle is to declare the $DSMSDIR/users directory as a resource which can be 'exported' by the NFS server (UNIX machine), and to 'mount' this resource as a read-write network resource on the client NFS workstation (Windows PC).

When using FTP, you transfer the files generated from the UNIX machine to the workstation via FTP commands. These commands are integrated to the PACAGP procedure.

FTP transfer must be made in binary mode.

For further details, refer to the corresponding TCP/IP documentation.

21.7. INSTALLATION OF PACLINK-EMULATED WORKSTATIONS

INSTALLATION OF PACLINK-EMULATED WORKSTATIONS

Once the DSMS servers are installed, you may start installing the dumb Terminal workstation running under Windows.

1 COPYING THE PACLINK FILES

All the PACLINK specific files are found in the $DSMSDIR/paclink directory. These files are described in Subchapter 'Description of Created Directories'.

They must be copied on the hard disk of the MS-DOS workstation, via the local network, in an installation directory chosen by the user.

2 UPDATING THE PARAMETER FILES

Two parameter files should be modified according to the installation characteristics:

GSTCPIP.PRM

Specify the following elements :

. The name (UNIX command 'hostname') of the UNIX machine on which DSMS is installed (P0 line) or the IP address of this machine (P1 line).

Both lines are exclusive.

. The socket number assigned to the DSMS on-line server (P2 line);

. The file used when exiting DSMS with a backup of the conversation (SA line);

. The file used for the Trace function if this function is activated (D0 line).

Example:

On a UNIX system with the 'MYHOST' network name, and with the on-line server 'listening' to socket 1500, the conversation back-up file C:\DSBASE\SAVE.TXT and the trace file C:\DSBASE\GSPACLAN.DEB, we have:

```
            P0 MYHOST
            P2 1500
            SA C:\DSBASE\SAVE.TXT
            D0 0 1 0 1 0 0 0 C:\DSBASE\GSPACLAN.DEB
```

The other lines do not need updating.

PACLINK.PRM

Specify the disk unit and the directory under which GSTCPIP.EXE (G line), and GSTCPIP.PRM (E line) were installed.

```
Example:    G C:\DSBASE\GSTCPIP.EXE
            E C:\DSBASE\GSTCPIP.PRM
```

CREATION OF A WINDOWS ICON

Create a program: 'File' menu, 'New...' command, in a group of your choice. The 'command line' must read the complete access path to the PACLINK.EXE file, separated by a blank from the PACLINK.PRM file access path.

Example: C:\DSBASE\PACLINK.EXE C:\DSBASE\PACLINK.PRM

21.8. INSTALLATION OF REVAMPED WORKSTATIONS (PAW)

INSTALLATION OF REWAMPED WORKSTATIONS (PAW)

1. Assign the PACLAN option to the 'COM' parameter of the Communications Manager.

2. To ensure the connection of a workstation to its DSMS on-line server, the communication files must be correctly assigned in the file whose name is specified in the ENVIRON.PRM file.

These files are found on one of the network's shared disks, in the $DSMSDIR"/commun directory, and are called:

```
'on_line_server_name'.dcm
'on_line_server_name'.dcd
```

21.9. INSTALLATION OF UNIX DUMB TERMINALS

INSTALLATION OF A UNIX DUMB TERMINAL (DSWSTSOCK)

The 'dswstsock' executable program, installed in $DSMSDIR/bin, allows to start up a UNIX terminal:

'dswstsock' communicates with the on-line server via the socket interface.

CONDITION FOR USING A UNIX TERMINAL

In order to use a terminal, some conditions are required:

- create a user account and a UNIX login specific to the TP/batch server.

- check the userid bit activation of the 'dswstsock' executable with the following commands:

```
cd $DSMSDIR/bin
1s -1 dswstsock
```

The system displays one line. The first 4 characters of this line are '-rws'.

If this is not the case, the DSMS administrator, with his/her login, must execute the following command:

```
chmod u+s dswstsock
```

- Under the administrator login, modify the shell configuration files (.profile, .kshrc or .login according to the case).

The environment variables DSMSDIR and PATH will be created or modified:

```
DSMSDIR=<access path to the DSMS directory>
export DSMSDIR

PATH=$DSMSDIR/bin:$PATH
export PATH
```

After these modifications, restart the configuration initialization by typing '. .profile', and, using the UNIX command 'set', check that the modifications were taken into account in the configuration files.

If the environment variables are not correctly initialized, log off and then log in again in the user account.

STARTING THE UNIX TERMINAL

Refer to Chapter 'MONITOR START-UP', Subchapter 'UNIX Terminal Start-up'.

21.10. SETTING OF A VA PAC DATABASE UNDER DSMS CONTROL

SETTING A VA PAC DATABASE UNDER DSMS CONTROL

Implementation under VisualAge Pacbase

To put a VA Pac Database under DSMS control, access the PK screen of the PARM transaction (VA Pac user parameter update).

Indicate 'YES' in the 'DATABASE UNDER DSMS CONTROL' field in the bottom of the screen.

Authorization level 3 is required for this operation.

Implementation under DSMS

The screen accessed through the PL choice enables you to specify the libraries, sessions and entities that are to be controlled by DSMS.

NOTE:

In this screen, the VA Pac Database code that is required is the Database's logical code, displayed in the upper-right corner of the VA Pac screens. This code can be modified by a user input in the REST procedure (restoration procedure).

For more information, refer to the DSMS reference manual, Chapter 'VA Pac interface: DATABASE LOCK'.

One VA Pac Database and one DSMS Database

If a VA Pac Database is monitored by a DSMS Database, the implementation described above is quite adapted. Moreover, the standard installation of DSMS and VA Pac locates the DSMS DC file in the BASES directory of the VA Pac installation. It thus allows the control of the VA Pac Database if relevant.

However, if a VA Pac Database must be monitored by a DSMS Database, the implementation described above is not sufficient. The location of the DSMS Database DC file must be modified. The file must be moved in the BASES directory of the VA Pac Database to be monitored.

To relocate the DC file, modify its assignment file in both the DSMS and VA Pac installations. The assigment file is in the ASSIGN\'db_name' directories of the DSMS and VA Pac installations.

Multiple VA Pac and DSMS Databases

If several VA Pac Databases are monitored by only one DSMS Database, the adequate implementation procedure is the one described above. But the standard DSMS and VA Pac installations require that the DSMS DC file be located in the BASES directory of the VA Pac installation. Thus, it can monitor all Databases if necessary.

However, if several VA Pac Databases are to be monitored by several DSMS Databases, the implementation steps described above are not sufficient. Each DSMS Database's DC file must be relocated in the BASES\'db_name' directory of the VA Pac Database to be monitored.

To relocate the DC file, modify its assignment file in both the DSMS and VA Pac installations. The assigment file is in the ASSIGN\'db_name' directories of the DSMS and VA Pac installations.

21.11. USE TESTS

USE TESTS

DSMS INSTALLATION TESTS

These tests break down into the following phases:

. On-line Database use tests,

. Extraction utility test,

. Database management tests.

1. ON-LINE DATABASE USE TESTS

Start up an on-line server.

Connect a workstation to it: the user codes defined in the test Database are TEST or USER, the password is IBM (French is the TEST language and English is the USER language).

Work in the DSMS Database, in read-only or read-write mode.

2. EXTRACTION TEST

Run the DEXT procedure which extracts elements from the test Database.
For this test, the on-line server may remain active.

The transaction file MBDEXT, provided as a default in the 'release'\INPUT\'db_name' directory, extracts the status table and a query.

3. DATABASE MANAGEMENT TESTS

The objective of these tests is to run the Database management procedures.

These tests consist of the following steps (to be performed in this order):

. Archive the journal created during tests: run the DARC procedure which outputs the BJ.NEW file, then runs the BJBACKUP command file at the end of the archiving to create the BJ file (and the BJ-1 file if necessary).

. Back up the Database directly: run the DSAV procedure which outputs the BB.NEW file, then run the BBBACKUP command file at the end of the backup to create the BB file (and the BB-1 file if necessary).

. Restore the Database from the BJ archive and the BB Database backup: run the DRST procedure.

For all these tests, the on-line servers must be shut down.

After restoring the Database, perform another set of quick operational tests on-line (but make sure to start up the on-line server first).

21.12. MANAGEMENT OF SEVERAL DSMS DATABASES

MANAGEMENT OF SEVERAL DSMS DATABASES

Standard location of the files of a DSMS Database

In a standard installation, one of the parameters of the dsmsadmin procedure corresponds to the DSMS Database name (test, for example). It allows for the creation of the Database's specific subdirectories:

```
$DSMSDIR/assign/test
$DSMSDIR/bases/test
$DSMSDIR/tmp/test
$DSMSDIR/save/test
$DSMSDIR/input/test
```

The test Database's files are copied in each of the above directories.

The TE error message file is copied directly in $DSMSDIR/bases because it may be used by several DSMS Databases.

All servers and batch procedures assign the files located in 'bases' and 'bases/'db_name" via command files that are created at installation time in the 'assign/'db_name" directory. This facilitates the modifications that may be made on the standard installation.

The standard DSMS architecture is designed for the management of a multi-Database environment.

Creation of a new DSMS Database

To create a new Database, go to the $DSMSDIR/install directory by typing the 'cd $DSMSDIR/install' command.

Then type:

-'sh dsmsinstall -e' for an English creation

-'sh dsmsinstall -f' for a French creation

and press the Enter or Return key.

Description of steps:

The creation procedure contains the following steps:

- Menu display (below)
- Command input (crebase)
- Database name input
- Creation of the Database's specific directories
- Writing of information about the new Database in the journal_dsmsinstall file

Menu display

The dsmsinstall menu looks like this:

```
************************************************************

                           DSMS

************************************************************
    .
    .

 crebase        :  creation of a new Database
    .
    .

************************************************************

 Type the command name or 'x' to exit dsmsadmin.
```

To execute the command, follow the instructions given by the dsmsadmin procedure itself.

Update of the journal_dsmsinstall file

The following information is added to the journal_dsmsinstall file:

- Database name

- Creation date

Database initialization

The dsmsadmin procedure initializes the new Database with the PC file of the test Database.

21.13. STANDARD RE-INSTALLATION OF A DSMS VERSION

RE-INSTALLATION OF A DSMS RELEASE

DSMS must be re-installed when a new version comes out, with corrections and/or new specific developments to the current release.

The new version is identified by a number, and consists of:

. A re-installation cartridge (or tape),

. A list of corrected bugs

. Instructions.

In general, only the system files (AEO, transaction files in $DSMSDIR/method and generation features) and the programs (batch and and on-line) are affected by the new version.

General notes

.The re-installation procedure does not create directories that are identical to those created during the first installation of the release.

.It does not copy the batch procedures ('$DSMSDIR/batch/proc' directory) if they have been adapted to the site by the Database administrator, except in case the new version cannot operate with the old procedures.

Submission of the re-installation procedure:

Once the files have been copied from the support onto the hard disk drive, type the following command (same as for an installation):

'sh dsmsinst* -e' (re-installation in English),

'sh dsmsinst* -f' (re-installation in French),

and press the Enter or Return key.

Description of steps:

The steps of the re-installation procedure are:

- Consistency check on the environment variables (.profile, .kshrc, or .login file),
- Display of the contents of the readme file,
- Display of the re-installation menu
- Selection of a character set (RS6000 or DPX/20 only),
- Selection of the COBOL run-time used (on HP9000, RS6000 or DPX20 only),
- Distribution of the DSMSBASE.xxx files.
- Update of the journal_dsmsinstall file.

Consistency check on the environment variables:

Refer to Subchapter 'DSMS Installation'.

Display of the readme file:

Refer to Subchapter 'DSMS Installation'.

Display of re-installation menu

The re-installation menu looks like this:

```
************************************************************

                     DSMS

************************************************************

 readme       :  display the 'readme' file
 patch        :  selective re-installation of patches
 reinst       :  re-installation of the release

************************************************************

Type the name of the command or 'x' to exit dsmsinstall.
```

Description of commands:

The 'readme' command prompts the display of the readme information file.

The 'patch' command allows a selective reinstallation of the changes in the new version. The procedure displays the list of patches for this version, and for each patch the corresponding changes. Enter the patch names, separated by a space. For more details concerning the changes, the user may refer to the lists supplied with the version.

The 'reinst' command performs a complete re-installation of the new version.

Type the desired command, and follow the instructions given by dsmsinst.

Selection of the character set (RS6000 - DPX/20):

Refer to Subchapter 'DSMS Installation'.

Selection of the COBOL run-time used (HP9000):

Refer to Subchapter 'DSMS Installation'.

Update of the journal_dsmsinstall file:

The following information, related to the re-installation, is added to the journal_dsmsinstall file:

- Re-installation date
- Release number
- Version number
- Name of the character set used

Retrieval of the batch procedures

The general case is that the batch procedures affected by the new version must be updated by the DSMS administrator, either by editing them under an editor, or by copying them. All of the new version's procedures are unloaded in the following directory:

$DSMSDIR/batch/proc.version_number

In case the new version cannot operate with the old procedures, these are backed up in the $DSMSDIR/batch /proc.old directory, and the new procedures are unloaded in:

$DSMSDIR/batch/proc

The DSMS administrator must modify the new batch procedures in the $DSMSDIR/batch/proc directory in order to adapt them to the site's requirements.

22. RETRIEVAL OF DSMS 8.0.1 --> DSMS 2.5 (DR80)

22.1. OPERATIONS TO CARRY OUT

OPERATIONS TO CARRY OUT

The retrieval of a DSMS 8.0 (or 8.0.1) database and its adaptation to the new release requires the following operations:

. 8.0/8.0.1 archival of the Database (DARC procedure).

. 8.0/8.0.1 backup of the Database, producing a 8.0/8.0.1 file called BB (DSAV procedure).

Then, with the NEW INSTALLATION, execute the following procedures:

. Convert the 8.0/8.0.1 (BB) DSMS database backup to the new format (DR80 procedure).

. Reorganize the backup in order to rebuild the DX cross-references in the new format (DREO procedure).

. Restore the database (DRST procedure).

22.2. USER INPUT

USER INPUT

User input allows product codes to be changed from one character to three. It is composed of 1 to n lines with each line sub-divided into groups of four characters starting from column one.

POS.	LEN.	VALUE	MEANING
1	1	'P'	Old product code
2	3	'PRO'	New product code Each group of 4 characters can be repeated a maximum of 20 times per line.

22.3. BACKUP RETRIEVAL

DR80: DESCRIPTION OF STEPS

```
CONVERSION PREPARATION:  PDSR8B

.Permanent input file
 -Backup of DSMS 8.0.1 DSMS database
  BB.OLD file in SAVE directory

.Output work files:

  WIQ file in temporary directory
  WIT file in temporary directory
  WIW file in temporary directory
  WLA file in temporary directory

.Sort files:

.Input transaction file:
  MBDR80 file in INPUT directory

.Output report:
 -Retrieval report


8.0.2 BACKUP CONVERSION:  PDSR8C

.Input work files:
WIQ file in temporary directory
WIT file in temporary directory
WIW file in temporary directory
WLA file in temporary directory

.Permanent input file:
 -Backup of DSMS 8.0.1 database
  BB.OLD in SAVE directory

.Permanent output file:
 -Backup of converted DSMS database
  WB1 file in temporary directory

.Input transaction file:
  MBDR80 file in INPUT directory

.Sort files

.Output report:
 -Retrieval report

CONVERSION OF BACKUP 1.2 TO 2.5 FORMAT:  PDSR15

.Permanent input file:
 -Backup file of DSMS 1.2 or 1.5 Database
  WB1 file in temporary directory

.Permanent output file:
 -Backup of converted DSMS Database
  BB.NEW file in SAVE directory

.Output report:
 -Printing report
```

in temporary directory

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

22.4. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--         Release xxx Version xxx         --
#@(#)
#@(#)DSMS (R) DR80 BATCH PROCEDURE
#@(#)
# Parameter control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "----------------------------------------------------------"
echo "                         DR80 PROCEDURE"
echo "                         =============="
echo "Directory 'assign'                : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                   : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                 : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files   : $2"
fi
echo ""
. $DSMSDIR/assign/$1/PACSAVBB.ini
echo "NOTE:"
echo "Input BB file (8.0/8.0.1 format)  : $PACSAVBB.OLD"
echo "Output BB file                    : $PACSAVBB"
echo "----------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# ***************************************************************
# * DSMS : RETRIEVAL OF A DSMS DATABASE 8.0 OR 8.0.1
# ***************************************************************
# * INPUT :
# * 1 to N LINES WITH 4 CHARACTERS GROUPS
# *        (20 GROUPS MAX PER LINE)
# * COL 1   : 'P'   OLD PRODUCT CODE
# * COL 2-4 : 'PRO' NEW PRODUCT CODE
# ***************************************************************
PACDBB=$PACSAVBB.OLD
export PACDBB
PACDMB=$PACINPUT'MBDR80'
export PACDMB
PACDIQ=$PACTMP'WIQ'
export PACDIQ
PACDIT=$PACTMP'WIT'
export PACDIT
PACDIW=$PACTMP'WIW'
export PACDIW
PACDLA=$PACTMP'WLA'
export PACDLA
PACDIK=$PACTMP'DR80IK.R8B'
export PACDIK
echo "Execution: PDSR8B"
cobrun PDSR8B
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR8B"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# ***************************************************************
PACDBB=$PACSAVBB.OLD
```

```
export PACDBB
PACDB1=$PACTMP'WB1'
export PACDB1
PACDMB=$PACINPUT'MBDR80'
export PACDMB
PACDIQ=$PACTMP'WIQ'
export PACDIQ
PACDIT=$PACTMP'WIT'
export PACDIT
PACDIW=$PACTMP'WIW'
export PACDIW
PACDLA=$PACTMP'WLA'
export PACDLA
PACDIO=$PACTMP'DR80IO.R8C'
export PACDIO
echo "Execution: PDSR8C"
cobrun PDSR8C
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR8C"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# ***************************************************************
PACDBB=$PACTMP'WB1'
export PACDBB
PACDB1=$PACSAVBBNEW
export PACDB1
PACDIK=$PACTMP'DR80IK.R15'
export PACDIK
echo "Execution: PDSR15"
cobrun PDSR15
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR15"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# ***************************************************************
echo "End of procedure"
echo ""
echo "Deletion of the temporary files"
rm -f  $PACTMP/WB1
rm -f  $PACTMP/WIQ
rm -f  $PACTMP/WIT
rm -f  $PACTMP/WIW
rm -f  $PACTMP/WLA
echo ""
echo "Calling the file BBBACKUP.ini"
. $DSMSDIR/assign/$1/BBBACKUP.ini
exit $RETURN
```

23. RETRIEVAL OF DSMS 8.0.2 01/02 --> DSMS 2.5 (DR8Q)

23.1. OPERATIONS TO CARRY OUT

OPERATIONS TO CARRY OUT

NOTE: this chapter relates to databases already installed in 8.0.2 (version 01 or 02); if a DR8X or DR80 retrieval procedure was executed, do not perform this new retrieval.

Installing the new DSMS release requires the retrieval of the DSMS database queries, which includes the following steps:

Using the 8.0.2 01 or 02 procedures:

1. DSMS database archive (DARC)

2. DSMS database backup (DSAV)

Then, with the NEW INSTALLATION, execute the following procedures:

3. Retrieval of the BB file (DR8Q).

4. DSMS backup reorganization (DREO).

5. DSMS database restoration (DRST).

EXECUTION CONDITIONS

None.
However, to ensure the consistency of the retrieved database, it is recommended to close the access to on-line use.

USER INPUT

None.

23.2. BACKUP RETRIEVAL

DR8Q: DESCRIPTION OF STEPS

```
QUERY RETRIEVAL:  PDSR8Q

.Permanent input file:
 -Backup of DSMS 8.0.2 01/02 Database
  BB.OLD file in SAVE directory

.Output work file:
  WIQ file in temporary directory

.Sort files:



MERGE:  PDSR8R

.Input work file
  WIQ file in temporary directory

.Permanent input file:
 -Backup of DSMS Database 8.0.2 01/02
  BB.OLD file in SAVE directory

.Output Permanent file :
 -Backup of retrieved DSMS Database
  WB1 file in temporary directory

CONVERSION OF BACKUP 1.2 TO 2.5 FORMAT:  PDSR15

.Permanent input file:
 -Backup file of DSMS 1.2 or 1.5 Database
  WB1 file in temporary directory

.Permanent output file:
 -Backup of converted DSMS Database
  BB.NEW file in SAVE directory

.Output report:
 -Printing report
  in temporary directory



SYMBOLICS IN USE

+---------+-----------------------------------------------+
!SYMBOLICS! MEANING                                       !
+---------+-----------------------------------------------+
! $1      ! Database name                                 !
+---------+-----------------------------------------------+
```

23.3. EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--         Release xxx Version xxx         --
#@(#)
#@(#)DSMS (R) DR8Q BATCH PROCEDURE
#@(#)
# Parameter control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                         DR8Q PROCEDURE"
echo "                         =============="
echo "Directory 'assign'                : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                   : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                 : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files   : $2"
fi
echo ""
. $DSMSDIR/assign/$1/PACSAVBB.ini
echo "NOTE:"
echo "Input BB file (8.0.2 format)      : $PACSAVBB.OLD"
echo "Output BB file                    : $PACSAVBB"
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : RETRIEVAL OF A DATABASE 8.0.2 01 OR 02
# **************************************************************
PACDBB=$PACSAVBB.OLD
export PACDBB
PACDIQ=$PACTMP'WIQ'
export PACDIQ
echo "Execution: PDSR8Q"
cobrun PDSR8Q
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR8Q"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# **************************************************************
PACDBB=$PACSAVBB.OLD
export PACDBB
PACDB1=$PACTMP'WB1'
export PACDB1
PACDIQ=$PACTMP'WIQ'
export PACDIQ
echo "Execution: PDSR8R"
cobrun PDSR8R
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR8R"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# ***************************************************************
PACDBB=$PACTMP'WB1'
export PACDBB
PACDB1=$PACSAVBBNEW
export PACDB1
PACDIK=$PACTMP'DR8QIK.R15'
export PACDIK
echo "Execution: PDSR15"
cobrun PDSR15
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR15"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# ***************************************************************
echo "End of procedure"
echo ""
echo "Deletion of the temporary files"
rm -f  $PACTMP/WIQ
rm -f  $PACTMP/WB1
echo ""
echo "Calling the file BBBACKUP.ini"
. $DSMSDIR/assign/$1/BBBACKUP.ini
exit $RETURN
```

24. RETRIEVAL OF DSMS 8.0.2, COMPATIBLE VA PAC 8.0.1

24.1. OPERATIONS TO CARRY OUT

INTRODUCTION

The current release of DSMS is not compatible with VisualAge Pacbase release 8.0.1.

If DSMS 8.0.2 was used with VA Pac 8.0.1, the installation of the new version of DSMS requires the following operations to be carried out:

1. DSMS Database backup (DSAV),
2. Installation of the new DSMS version,
3. Installation of the new VA Pac version,
4. Retrieval of the BB backup file (DR8X procedure),
5. Reorganization of the DSMS Database to rebuild the DX file.
6. DSMS Database restoration.

EXECUTION CONDITIONS

None.
However, to ensure the integrity of the retrieved Database, it is recommended to close the Database to on-line use.

24.2. 'DR8X' PROCEDURE - USER INPUT

USER INPUT

User input allows product codes to be changed. It is composed of 1 to n lines with each line sub-divided into groups of six characters starting from column one.

POS.	LEN.	VALUE	MEANING
1	3	'PRO'	Old product code
4	3	'PRO'	New product code
			Each group of 6 characters can be repeated a maximum of 13 times per line.

24.3. 'DR8X' PROCEDURE

DR8X: DESCRIPTION OF STEPS

CONVERSION FOR VA PAC 8.0.2: PDSR8X

.Permanent input file:
-Backup of DSMS 8.0.1 Database
BB.-- file in SAVE directory

.Sort files:

.Input transaction file:
MBDR8X file in INPUT directory

.Permanent output file:
-Backup of converted DSMS Database
DSN=BBOLD

.Output report
-Retrieval report

CONVERSION OF BACKUP 1.2 TO 2.5 FORMAT: PDSR15

.Permanent input file:
-Backup file of DSMS 1.2 or 1.5 Database
WB1 file in temporary directory

.Permanent output file:
-Backup of converted DSMS Database
BB.NEW file in SAVE directory

.Output report:
-Printing report
in temporary directory

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

24.4. 'DR8X' PROCEDURE - EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--         Release xxx Version xxx         --
#@(#)
#@(#)VA Pac DSMS (R) DR8X BATCH Procedure
#@(#)
# Parameter control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "----------------------------------------------------------"
echo "                         DR8X PROCEDURE"
echo "                         =============="
echo "Directory 'assign'                : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                   : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                 : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files   : $2"
fi
echo ""
. $DSMSDIR/assign/$1/PACSAVBB.ini
echo "NOTE:"
echo "Input BB file (8.0.2 format)      : $PACSAVBB.OLD"
echo "Output BB file                    : $PACSAVBB"
echo "----------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# ***************************************************************
# * VA Pac DSMS : RETRIEVAL OF DSMS DATABASE CONNECTED TO
# *               VA Pac 8.0.1
# ***************************************************************
# * INPUT :
# * 1 to N LINES WITH 4 CHARACTERS GROUPS
# *        (20 GROUPS MAX PER LINE)
# * COL 1   : 'P'   OLD PRODUCT CODE
# * COL 2-4 : 'PRO' NEW PRODUCT CODE
# ***************************************************************
PACDBB=$PACSAVBB.OLD
export PACDBB
PACDB1=$PACTMP'WB1'
export PACDB1
PACDMB=$PACINPUT'MBDR8X'
export PACDMB
PACDIK=$PACTMP'DR8XIK.R8X'
export PACDIK
echo "Execution: PDSR8X"
cobrun PDSR8X
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR8X"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# ***************************************************************
PACDBB=$PACTMP'WB1'
export PACDBB
PACDB1=$PACSAVBBNEW
export PACDB1
PACDIK=$PACTMP'DR8XIK.R15'
export PACDIK
```

```
echo "Execution: PDSR15"
cobrun PDSR15
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR15"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# **************************************************************
echo "End of procedure"
echo ""
echo "Deletion of the temporary files"
rm -f  $PACTMP/WB1
echo ""
echo "Calling the file BBBACKUP.ini"
. $DSMSDIR/assign/$1/BBBACKUP.ini
exit $RETURN
```

25. RETRIEVAL OF DSMS 1.2 OR 1.5 --> DSMS 2.5

25.1. OPERATIONS TO CARRY OUT

OPERATIONS TO CARRY OUT

The retrieval of a DSMS 1.2 (or 1.5) database and its adaptation to the new release requires the following operations:

. 1.2/1.5 archival of the Database (DARC procedure).

. 1.2/1.5 backup of the Database, producing a 1.2/1.5 file called BB (DSAV procedure).

Then, with the NEW INSTALLATION, execute the following procedures:

. Convert the 1.2/1.5 (BB) DSMS database backup to the new format (DR15 procedure).

. Reorganize the backup in order to rebuild the DX cross-references (DX file) in the new version format (DREO procedure).

. Restore the database (DRST procedure).

NOTE:

It is possible to retrieve the sequential version of the journal (Rel. 1.2 or 1.5) with the DR5J procedure.

25.2. 'DR15' PROCEDURE - DESCRIPTION OF STEPS

DR15: DESCRIPTION OF STEPS

```
RETRIEVAL OF DSMS 1.2 / 1.5: PDSR15

.Permanent input file:
 -DSMS 1.2 or 1.5 database backup
  BB.OLD file in SAVE directory

.Permanent output file:
 -Retrieved DSMS database backup
  BB.NEW file in SAVE directory

.Output report:
 -Printing report
```

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

25.3. 'DR15' PROCEDURE - EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--         Release xxx Version xxx         --
#@(#)
#@(#)VA Pac DSMS (R) DR15 BATCH Procedure
#@(#)
# Parameter control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                         DR15 PROCEDURE"
echo "                         =============="
echo "Directory 'assign'                : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                   : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                 : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files   : $2"
fi
echo ""
. $DSMSDIR/assign/$1/PACSAVBB.ini
echo "NOTE:"
echo "Input BB file (1.2 / 1.5 format)  : $PACSAVBB.OLD"
echo "Output BB file                    : $PACSAVBB"
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * VA Pac DSMS : RETRIEVAL OF A DSMS DATABASE 1.2 OR 1.5
# **************************************************************
PACDBB=$PACSAVBB.OLD
export PACDBB
PACDB1=$PACSAVBBNEW
export PACDB1
PACDIK=$PACTMP'DR15IK.R15'
export PACDIK
echo "Execution: PDSR15"
cobrun PDSR15
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR15"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# **************************************************************
echo "End of procedure"
echo ""
echo "Calling the file BBBACKUP.ini"
. $DSMSDIR/assign/$1/BBBACKUP.ini
exit $RETURN
```

25.4. 'DR5J' PROCEDURE - DESCRIPTION OF STEPS

```
RETRIEVAL OF JOURNAL FILE: PDSR5J

.Permanent input file:
 -Sequential image of journal file, Rel. 1.2 or 1.5
  BJ.OLD file in SAVE directory

.Permanent output file:
 -Journal retrieved in the 2.5 format
  BJ.NEW file in SAVE directory

.Output report:
 -Printing report
  in temporary directory
```

25.5. 'DR5J' PROCEDURE - EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--         Release xxx Version xxx         --
#@(#)
#@(#)VA Pac DSMS (R) DR5J BATCH Procedure
#@(#)
# Parameter control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                         DR5J PROCEDURE"
echo "                         =============="
echo "Directory 'assign'                : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                   : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                 : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files   : $2"
fi
echo ""
. $DSMSDIR/assign/$1/PACSAVBJ.ini
echo "NOTE:"
echo "Input BJ file (1.2 / 1.5 format)  : $PACSAVBJ.OLD"
echo "Output BJ file                    : $PACSAVBJ"
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * VA Pac DSMS : RETRIEVAL OF A DSMS JOURNAL FILE 1.2 OR 1.5
# **************************************************************
PACDBJ=$PACSAVBJ.OLD
export PACDBJ
PACDJB=$PACSAVBJNEW
export PACDJB
echo "Execution: PDSR5J"
cobrun PDSR5J
RETURN=$?
case $RETURN in
0)
 ;;
*)
 echo "Error in executing PDSR5J"
 echo "Error $RETURN"
 sh $DSMSDIR/batch/proc/ERRPAUSE.ini
 exit $RETURN
 ;;
esac
# **************************************************************
echo "End of procedure"
echo ""
echo "Calling the file BJBACKUP.CMD"
. $DSMSDIR/assign/$1/BJBACKUP.ini
exit $RETURN
```

26. DSMS DATABASE MIGRATION (DRBB)

26.1. INTRODUCTION

INTRODUCTION

The purpose of this procedure is to retrieve DSMS Databases from other platforms (IBM, BULL...) to adapt them to a DSMS installation in an OS/2, WINDOWS/NT or UNIX environment.

The backup of the DSMS database coming from the other platform and converted into an ASCII format is sorted according to an ASCII sequence by the DRBB procedure.

If the DSMS version on the source site is the same as the target's version, the BB backup file produced after the execution of the DRBB procedure can be directly restored to make up the DSMS Database.

However if the version of the source Database is older (8.0, 8.0.1, 8.0.2 v01 or v02), or if the VA Pac Database of the source site (DSMS controlled) is an 8.0.1 version, you should refer to chapter 'RETRIEVAL' corresponding to the installation on the source site.

It should provide the description of the procedure to be executed (DR80, DR8Q or DR8X) before executing the reorganization procedure (DREO) and the ASCII format retrieval (DRBB).

The Database can then be restored (DRST procedure) to make up the DSMS Database on the Local Area Network.

EXECUTION CONDITION

None.

However, it is recommended to read the following notes carefully.

USER INPUT

None.

NOTES:

1. TRANSFER OF THE BB BACKUP FILE

The transfer of the BB backup file from the source site to the Local Area Network on which DSMS will be installed is the user's responsibility.
The parameters which must be specified for the transfer are: the data files (DATA) which must be converted into ASCII format and must contain the control characters for the end of records (generally, CRLF parameter).

2. SPECIAL CHARACTERS AND LOW-VALUES

The backup file found on the host may contain special characters that may cause problems when transferred onto the Local Area Network. In this case, the BB file should be processed on the host before being transferred: the DLVB batch procedure replaces all low-values by blanks in the BB backup. If other problems remain, contact your Technical Support to obtain another utility program.

3. DISK SPACE

The DRBB procedure described hereafter consists in sorting the BB backup file according to an ASCII sequence. The sort is performed in three separate programs so as to reduce the disk space requirements. However, the procedure requires 4 to 4,5 times as much space as the initial file. For example, if the BB backup file size is 10Mb, about 43Mb will be necessary to execute the procedure.

The procedure does not include the deletion of the BB.INI file (from the source site). You can delete it after the end of the first step if you need more disk space.

Finally, to reduce execution time, it is recommended to create the work and sort files on the DSMS server disk that executes the programs.

26.2. DRBB : PARAMETERS - DESCRIPTION OF STEPS

CAUTION:

The BB input file (produced on another platform) is called BB.INI.

The output BB file (ASCII-sorted) is called BB.NEW and is renamed BB by the BBBACKUP command file.

These files are located in the sequential file directory of the DSMS database.

DESCRIPTION OF STEPS

The DRBB procedure includes the following step:

```
    .Positioning the sort switch

    .Division of the backup file         : PDSRB1

    .Data sort                           : PDSRBA

    .Sort on the VA Pac elements         : PDSRBC

    .Sort on Cross-references            : PDSRBX

    .Backup reconstitution               : PDSRB2
```

DIVISION OF THE BACKUP FILE: PDSRB1

```
.Input backup file                          : PACDBB
  (BB.INI file in SAVE directory)

.Output work files:
 -data sequential image                     : PACDBA
  (BA file in the temp. directory)
 -sequential image of the VA Pac
  elements                                  : PACDBC
  (BC file in the temp. directory)
 -sequential image of cross-references      : PACDBX
  (BX file in the temp. directory)
```

SORT ON DATA: PDSRBA

```
.Input work file:
 -data sequential image                     : PACDBA
  (BA file in the temp. directory)

.Output work file:
 -sorted data                               : PACDAB
  (AB file in the temporary directory)
```

Error-free end of step:
Deletion of BA temporary file

SORT ON THE VA PAC ELEMENTS: PDSRBC

```
.Input work file:
 -sequential image of the VA Pac
  elements                                  : PACDBC
  (BC file in the temporary directory)

.Output work file:
 -sorted VA Pac elements                    : PACDCB
  (CB file in the temporary directory)
```

Error-free end of step:
Deletion of BC temporary file

SORT ON THE CROSS-REFERENCES: PDSRBX

```
.Input work file:
 -sequential image of cross-references   : PACDBX
  (BX file in the temporary directory)

.Output work file:
 -sorted cross-references                : PACDXB
  (XB file in the temporary directory)
```

Error-free end of step:
Deletion of the BX temporary file

RECONSTITUTION OF THE BB BACKUP: PDSRB2

```
.Permanent output file:
 -BB backup in ASCII format              : PACDBB
  (BB.NEW file in the SAVE directory)

.Input work files:
 -sequential image of data               : PACDBA
  (AB file in the temporary directory)
 -sequential image of VA Pac elements    : PACDBC
  (CB file in the temporary directory)
 -sequential image of cross-references   : PACDBX
  (XB file in the temporary directory)
```

Error-free end of step:
Deletion of AB, CB and XB temporary files

Error-free end of procedure

Call of the BBBACKUP command file.

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

26.3. EXECUTION PROCEDURE

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DRBB BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                         DRBB PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo ""
. $DSMSDIR/assign/$1/PACSAVBB.ini
echo "NOTE:"
echo "Input BB file (host)             : $PACSAVBB.INI"
echo "Output BB file (ASCII format)    : $PACSAVBB"
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : RETRIEVAL OF A MAINFRAME DATABASE
# **************************************************************
PACDBB=$PACSAVBB.INI
export PACDBB
PACDBA=$PACTMP'BA'
export PACDBA
PACDBC=$PACTMP'BC'
export PACDBC
PACDBX=$PACTMP'BX'
export PACDBX
echo "Execution : PDSRB1"
cobrun PDSRB1
RETURN=$?
case $RETURN in
 0)
   PACDBA=$PACTMP'BA'
   export PACDBA
   PACDAB=$PACTMP'AB'
   export PACDAB
   echo "Execution : PDSRBA"
   cobrun PDSRBA
   RETURN=$?
   case $RETURN in
    0)
      echo "Deletion of the temporary files"
      rm -f $PACTMP'BA'
      PACDBC=$PACTMP'BC'
      export PACDBC
      PACDCB=$PACTMP'CB'
      export PACDCB
      echo "Execution : PDSRBC"
      cobrun PDSRBC
      RETURN=$?
      case $RETURN in
       0)
         echo "Deletion of the temporary files"
         rm -f $PACTMP'BC'
         PACDBX=$PACTMP'BX'
         export PACDBX
         PACDXB=$PACTMP'XB'
         export PACDXB
         echo "Execution : PDSRBX"
         cobrun PDSRBX
         RETURN=$?
         case $RETURN in
          0)
            echo "Deletion of the temporary files"
            rm -f $PACTMP'BX'
            PACDBB=$PACSAVBBNEW
            export PACDBB
            PACDBA=$PACTMP'AB'
            export PACDBA
            PACDBC=$PACTMP'CB'
            export PACDBC
            PACDBX=$PACTMP'XB'
            export PACDBX
            echo "Execution : PDSRB2"
            cobrun PDSRB2
            RETURN=$?
            case $RETURN in
             0)
               echo ""
               echo "Deletion of the temporary files"
               rm -f $PACTMP'AB'
               rm -f $PACTMP'CB'
               rm -f $PACTMP'XB'
               echo ""
               echo "End of procedure"
               echo ""
               echo "Call the file BBBACKUP.ini"
               sh $DSMSDIR/assign/$1/BBBACKUP.ini
               ;;
             *)
               echo "Error in executing PDSRB2"
               echo "Error $RETURN"
               ;;
            esac
            ;;
          *)
            echo "Error in executing PDSRBX"
            echo "Error $RETURN"
            ;;
         esac
         ;;
       *)
         echo "Error in executing PDSRBC"
         echo "Error $RETURN"
         ;;
      esac
      ;;
    *)
      echo "Error in executing PDSRBA"
      echo "Error $RETURN"
      ;;
   esac
   ;;
 *)
   echo "Error in executing PDSRB1"
   echo "Error $RETURN"
   ;;
esac
if [ "$RETURN" != '0' ]
then
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
fi
exit $RETURN
```

27. REPLACEMENT OF LOW-VALUES BY BLANKS (DLVB)

27.1. DLVB: REPLACEMENT OF LOW-VALUES BY BLANKS

REPLACEMENT OF LOW-VALUES BY BLANKS IN A BB FILE

The DLVB procedure inserts a blank wherever a low-value is present in the BB Database backup file.

The purpose of this procedure is to make possible the transfer of the BB file onto various platforms, while avoiding problems due to the presence of low-values during these transfers.

Utilization option

The DLVB procedure gives the user the opportunity to produce a transfer file containing only the 'data'-type records (refer to next subchapter).

In this case, the backup file obtained on the target platform after transfer will have to be reorganized (DREO procedure) in order to rebuild the cross-references file (DX file).

EXECUTION CONDITIONS

None

27.2. DLVB: PARAMETERS-DESCRIPTION OF STEPS

DLVB: DESCRIPTION OF STEPS

```
REPLACEMENT OF LOW-VALUES BY BLANKS: PDSLVB

.EXEC line: specify PARM='DATA' to keep only DATA
 records in the output file.
 To keep both INDEX and DATA records, do not specify
 anything.

.Input file:
 -Database backup
   BB file in SAVE directory

.Output file:
 -New Database backup
   BB.NEW file in SAVE directory
```

SYMBOLICS IN USE

SYMBOLICS	MEANING
$1	Database name

27.3. DLVB: EXECUTION JCL

```
#!/bin/sh
#@(#)
#@(#)--   Release xxx Version xxx   --
#@(#)
#@(#)DSMS (R) DLVB BATCH PROCEDURE
#@(#)
# Parameters Control
. $DSMSDIR/batch/proc/USAGE.ini
clear
echo ""
echo "---------------------------------------------------------"
echo "                         DLVB PROCEDURE"
echo "                         =============="
echo "Directory 'assign'               : $DSMSDIR/assign/$1"
. $DSMSDIR/assign/$1/PACTMP.ini
echo "Directory 'tmp'                  : `dirname $PACTMP.`"
. $DSMSDIR/assign/$1/PACINPUT.ini
echo "Directory 'input'                : `dirname $PACINPUT.`"
if [ -n "$2" ]
then
   echo "Radical 'tmp' and 'input' files  : $2"
fi
echo "---------------------------------------------------------"
echo ""
sh $DSMSDIR/batch/proc/MSGPAUSE.ini
# **************************************************************
# * DSMS : REPLACEMENT OF LOW-VALUES BY BLANKS IN A BB FILE
# *
# **************************************************************
. $DSMSDIR/assign/$1/PACSAVBB.ini
PACDBB=$PACSAVBB
export PACDBB
PACDB1=$PACSAVBBNEW
export PACDB1
echo "Execution : PDSLVB"
cobrun PDSLVB
RETURN=$?
case $RETURN in
 0)
   echo "End of procedure"
   echo ""
   echo "Call the file BBBACKUP.ini"
   sh $DSMSDIR/assign/$1/BBBACKUP.ini
   ;;
 *)
   echo "Error in executing PDSLVB"
   echo "Error $RETURN"
   ;;
esac
if [ "$RETURN" != '0' ]
then
   sh $DSMSDIR/batch/proc/ERRPAUSE.ini
fi
exit $RETURN
```