VisualAge Pacbase 2.5

DSMS 2.5 BULL GCOS7, TDS ENVIRONMENT & INSTALLATION

DEDD7000251A

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.ibm.com/software/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.

First Edition (November 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.ibm.com/software/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.

DEDD7000251A

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

VISUALAGE PACBASE - OPERATIONS MANUAL
DSMS - INSTALLATION & OPERATIONS
FOREWORD 1

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 operations parameters.

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. SYSTEM PARAMETERS

SYSTEM PARAMETERS

The JCL lines supplied at installation include parameters which allow DSMS to conform to the codification rules that apply on-site. Also, parameters are used to assign the various files to the different disks used on the site.

A complete list of parameters is found in this chapter.

Parameters are formatted as follows: '$XXXXX'. The '$' sign is used in order to locate parameters in JCL module names. 'XXXXX' is the parameter code.

Parameter values can be globally substituted via the DSZZEXEC and DSZZJCL procedures. (For more information, Chapter "INSTALLATION", Subchapter 'INSTALLATION OF THE COMPLETE JCL'.)

Program libraries and files are referred to by their parameterized names.

```
COMM 'DSMS 2.5';
*****************************************
*        INSTALLATION PARAMETERS        *
*                                       *
*  REPLACE, IF NEEDED, THE DEFAULT      *
*  VALUE OF EACH PARAMETER.             *
*  EACH PARAMETER LINE IS FORMATTED AS  *
*  FOLLOWS:                             *
*    $NNNNN = VALUE                     *
*                                       *
*  THIS FILE IS PROCESSED BY AN EDITOR  *
*  PROGRAM WHICH CHANGES PARAMETER LINES*
*  INTO SUBSTITUTION COMMANDS.          *
*                                       *
*  SUBSEQUENTLY,                        *
*                                       *
*  - ALL LINES WHOSE FIRST NON-BLANK    *
*    CHARACTER IS NOT A DOLLAR SIGN     *
*    ARE CONSIDERED AS COMMENTS.        *
*                                       *
*  - THE EQUAL SIGN (DELIMITER) CANNOT  *
*    BE USED IN A PARAMETER VALUE.      *
*                                       *
*****************************************

******** DEFAULT  D S M S   USER        *
$USER   =  IBM
******** NAME  OF  INSTALLATION  CATALOG  *
$CTNM   =  PV
******** D S M S     TDS NAME           *
$NMTD   =  TDS
******** INSTALLATION TAPE    *************
$TAPE   =  XXXXXX
$DVTP   =  CT/M5
******** LANGUAGE E(NGLISH)/F(RENCH) ******
$LANG   =  E
************ SUFFIX OF LIBRARIES **********
******** BATCH CU
$LIBCUB =  CUBLIB
******** TP CU
$LIBCUT =  CUTLIB
******** PERMANENT CU
$LIBCUP =  CUPLIB
******** JCL
$LIBJCL =  JCLLIB
******** INVOKE JCLS SUBMISSION
$LIBINV =  INVLIB
******** PRINT OF JCLS CALLED BY INVOKE **
** BLANK NOT TO LIST OR &LIST TO LIST
$LIST   =  &LIST
******** LM
$LIBLM  =  LMLIB
******** SM  (SEE NOTE)
$LIBSM  =  SMLIB
******** TDS SL (SEE NOTE)
$LIBSL  =  SLLIB
******** USERS SL
$LIBSU  =  SULIB
******** DATABASE FILES' REFERNCES*********
******** ROOT OF DSMS FILES             *
$ROOT   =  DS
******** IDENTIFIER OF DSMS FILES       *
$FILE   =  250
******** CISIZE OF BATCH FILES          *
```

```
** DEFAULT VALUE FOR AN MS/D500 DISK TYPE **
$CISEQ  =  14336
***************** REFERENCES **************
*NMXX   =  FILE PREFIX
*DVXX   =  DISK TYPE
*MDXX   =  DISK NAME
*CTXX   =  CATALOGING (Y OR N)
******** TDS USER FILES
$NMTU   =  DSTU
$DVTU   =  MS/FSA
$MDTU   =  DISC01
$CTTU   =  Y
******** TDS FILES
$DVTD   =  MS/FSA
$MDTD   =  DISC02
$CTTD   =  Y
******** BATCH USER FILES
$NMBU   =  DSBU
$DVBU   =  MS/FSA
$MDBU   =  DISC03
$CTBU   =  Y
******** BATCH SYSTEM FILES
$NMBS   =  DSBS
$DVBS   =  MS/FSA
$MDBS   =  DISC04
$CTBS   =  Y
******** DSMS JOURNAL FILE
$NMDJ   =  DSDJ
$DVDJ   =  MS/FSA
$MDDJ   =  DISC05
$CTDJ   =  Y
******** TEMPORARY FILE
$DVTM   =  MS/FSA
$MDTM   =  DISC06
******** LIBRARIES
$NMLI   =  DSLI
$DVLI   =  MS/FSA
$MDLI   =  DISC07
$CTLI   =  Y
************ SM ENVIRONMENT   (SEE  NOTE) *
*** IF INSERTION IN AN EXISTING TDS       *
*** TAKE THE TPRS WHICH ARE NOT ALLOCATED *
******** DSMS  SM NAME                     *
$TPR0   =  TPR
******** DSMS  SM  NAME                    *
$TPR1   =  TPR1
*****************************************
* IMPORTANT NOTE :
*****************************************
* TO BE CONSISTENT WITH THE PROCEDURES    *
* SHIPPED BY THE VENDOR:                  *
* TP7PREP, TP7GEN, ETC..                  *
* IBM PROPOSES A CHOICE OF VALUES WHICH   *
* RESPECT THE COHERENCE OF THESE          *
* PROCEDURES' PARAMETERS.                 *
* SO IT IS HIGHLY RECOMMENDED TO USE      *
* THESE VALUES.                           *
*****************************************
```

FILE PARAMETERS

In order to set parameter values and to see how they affect DSMS System files names, the following charts list all files sorted by the first parameter in the external name.

LIBRARIES

BEFORE PARAMETERIZATION	WITH DEFAULT VALUES
$NMLI.$LIBJCL	DSLI.JCLLIB
$NMLI.$LIBSU	DSLI.SULIB
$NMLI.$LIBCUB	DSLI.CUBLIB
$NMLI.$LIBCUT	DSLI.CUTLIB
$NMLI.$LIBCUP	DSLI.CUPLIB
$NMLI.$LIBLM	DSLI.LMLIB
$NMLI.SOURCE	DSLI.SOURCE
$NMLI.ABOUT	DSLI.ABOUT
$NMTD.$LIBSM	TDS.SMLIB
$NMTD.$LIBSL	TDS.SLLIB

BATCH USER FILES

BEFORE PARAMETERIZATION	WITH DEFAULT VALUES
$NMBU.$ROOT$FILEBB	DSBU.DS200BB
$NMBU.$ROOT$FILEBJ	DSBU.DS200BJ
$NMBU.$ROOT$FILEBQ	DSBU.DS200BQ

BATCH SYSTEM FILES

BEFORE PARAMETERIZATION	WITH DEFAULT VALUES
$NMBS.$ROOT$ROOTDEO	DSBS.DSDSDEO

TDS USER FILES

BEFORE PARAMETERIZATION	WITH DEFAULT VALUES
$NMTU.$ROOT$FILEDA	DSTU.DS200DA
$NMDJ.$ROOT$FILEDJ	DSDJ.DS200DJ
$NMTU.$ROOT$FILEDX	DSTU.DS200DX
$NMTU.$ROOT$FILEDC	DSTU.DS200DC
$NMTU.$ROOT$FILEDE	DSTU.DSDSDE
$NMTU.$ROOT$FILEHE	DSTU.DS200HE
$NMTU.$ROOT$FILEJB	DSTU.DS200JB
$NMTU.$ROOT$FILEPA	DSTU.DS200PA

NOTE

In case of a CATALOG-type installation, the required directories must have been previously created.

The AUTOATTACH option must be specified for the $CTNM installation catalog.

If the $CTNM catalog does not exist OR if it cannot be 'autoattached', the system administrator must execute the DSINMPRE procedure (2nd step of the installation process).

THE $CISEQ PARAMETER

For files used in batch procedures, the $CISEQ parameter enables you to choose a value for the CISIZE so that use of disk space can be optimized. This parameter specifies the size of CI in bytes. It must be a multiple of 512.

The number of elements that can be stored in a CI depends on the total length of elements. An element cannot be shared by two CI. The maximum number of elements per CI is 255.

As a result, for files used by VA Pac, the recommended $CISEQ values by disk type are as follows:

```
MS/D500 ----> 14336

MS/FSA  ----> 14336

MS/M452 ----> 12800

MS/M500 ---->  9216

MS/D300 ---->  9216

MS/B10  ----> 12800
```

****
*** WARNING ***
****

For GCOS7 release V3A7, buffer size is equal to CI size. However for GCOS7 release V3B7 (4 Kb of paged memory) buffer size is equal to CI size plus 32 rounded up to the nearest muliple of 4 Kb.

For VA Pac TDS files, a CISIZE of 4096 requires two memory pages (with the loss of one page).

For GCOS7 release V3B7 a CISIZE equal to 3584 is recommended for all VA Pac TDS files.

2.3. ON-LINE PROGRAMS

ON-LINE PROGRAMS

PROGRAM CODE	ASSOCIATED CHOICE - COMMENTS
xxCHOI	Choice-decoding sub-program
xx0AAO	First TPR
PDTPDF	DAF extraction 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

On-line sub-programs

CODE	Function
DSCHOI	Choice decoding
ZAR980	Display management
ZAR985	Search of screen type
DSCUAM	On-line check on change definition
DSCUEV	On-line check on event definition
DSCUMQ	On-line check on report definition
DSCURQ	On-line check on query definition
DSCUSI	On-line check on site definition

2.4. BATCH PROGRAMS

BATCH PROGRAMS

CODE	PROC	COMMENTS
DAFD10	DPDF	DAF pre-processor
PACABE		Batch abend
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	Additional check sub-program for PDSUB1
PDSCEV	-	Additional check sub-program for PDSUE1
PDSCMQ	-	Additional check sub-program for PDSUQ6
PDSCRQ	-	Additional check sub-program for PDSUQ1
PDSCSI	-	Additional check sub-program for PDSUS1
PDTPDF	-	DAF extraction sub-program
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 queries
PDSRQ1	-	Selects and extracts queries
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
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
PDS381	-	Verifies the journal (English version)
PDS400	-	Restores or initializes the files
PDS450	-	Retrieves the archived journal
PDS500	DSAV	Saves data/elements/references
PDS600	DEXP	Extracts VA Pac journal
PDS610	-	
REP2PJ	DEXQ	Extracts VA Pac journal < 2.0
PDS700	DXBJ	Extracts journalized transactions
PDS900	DUPD	Update of DSMS from DAF tables
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

2.5. THE OTHER LIBRARIES

OTHER LIBRARIES

LIBRARY OF TDS SOURCE ($NMTD.$LIBSL)

It requires two tracks (MS/FSA). This library includes the TDS source, the TPR source of beginning and end of DSMS conversation.

Member	Contents
STDS	TDS source
DS0AA0	TPR source of beginning/end of conversation

NOTE: This library is created using the TDS preparation procedure: TP7PREP (or MTPREP for GCOS7 V5).

THE USER SOURCES LIBRARY: $NMLI.SOURCE

It requires one cylinder (MS/FSA). The library contains the French and English DAF dictionaries (VisualAge Pacbase update transactions) and the user control sub-programs on the definitions of changes, events, sites, queries and reports.

Member	Contents
DAFDICF	French DAF dictionary
DAFDICA	English DAF dictionary
DSCUAM	TP control on change definition
DSCUEV	TP control on event definition
DSCUMQ	TP control on report definition
DSCURQ	TP control on query definition
DSCUSI	TP 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

LIBRARY OF USER TRANSACTIONS ($NMLI.$LIBSU)

It requires one cylinder (MS/FSA).

This library includes the transactions resulting from the database entity extraction.

LIBRARY OF BATCH COMPILE-UNITS ($NMLI.$LIBCUB)

It requires approximately 3 cylinders (MS/FSA).

This library includes the compile-units of all batch programs as well as the sub-programs listed in the previous subchapter.

Its presence on the disk unit is required only when installing DSMS since this library is used to build the $NMLI.$LIBLM library.

LIBRARY OF ON-LINE COMPILE-UNITS ($NMLI.$LIBCUT)

It requires approximately 6 cylinders (MS/FSA).

This library includes the compile-units of all on-line programs. It also includes the sub-programs listed hereafter.

Its presence on the disk unit is required only when performing the following operations:

. DSMS System installation,

. TPRs LINK in an SMLIB (for instance, when DSMS is being inserted in one of the site's TDS).

LIBRARY OF PERMANENT ON-LINE COMPILE-UNITS ($NMLI.$LIBCUP)

It requires one cylinder (MS/FSA).

This library only includes the sub-programs: DSCHOI, ZAR980 and ZAR985.

It is used when generating the TDS that controls DSMS. The compile-units of the sub-programs DSCHOI, ZAR980 and ZAR985 are linked to the TDS during this operation.

VERSIONS LIBRARY ($NMLI.ABOUT)

It contains the technical characteristics of the installed version: new changes, elements' dates, installation specificities.

2.6. 'SYSTEM' FILES

THE SYSTEM FILES

They constitute the actual system. They are not affected by the daily transactions and must be reloaded whenever the system is reinstalled.

They are the LIBRARIES described in the previous subchapters:

.The library of on-line executable modules,

.The library of batch executable modules,

.The library of COBOL sources,

.The libraries of compile units,

.The library of versions,

as well as the file containing the ERROR MESSAGES and the HELP DOCUMENTATION of the DSMS function (DE):

.Size : Approximately 30,000 records
.External name: $NMTU.$ROOT$ROOTDE
.Organization : UIND
.Recsize : 90
.Key : 17 (position 1)
.CI size : 2,048

2.7. 'USER' FILES

USER FILES

These files contain user information and are managed by the module.

The first four files contain the data directly managed by this module. They are:

.The DSMS Data file (DA)

```
     .External name: $NMTU.$ROOT$FILEDA
     .Organization : UIND
     .Recsize      : 80 minimum, 340 maximum
     .CI size      : 2,048
     .Key          : 40 (position 3)
```

.The Cross-References file (DX)

```
     .External name: $NMTU.$ROOT$FILEDX
     .Organization : UIND
     .Recsize      : 80
     .CI size      : 2,048
     .Key          : 50 (position 1)
```

.The VisualAge Pacbase Elements file (DC)

```
     .External name: $NMTU.$ROOT$FILEDC
     .Organization : UIND
     .Recsize      : 50 minimum, 168 maximum
     .CI size      : 4,096
     .Key          : 31 (position 3)
```

.The DSMS Journal file (DJ)

```
     .External name: $NMTU.$ROOT$FILEDJ
     .Organization : UREL
     .Recsize      : 180
     .CI size      : 2,048
```

.The DSMS help file (HE)

     .External name: $NMTU.$ROOT$FILEHE
     .Organization : UIND
     .Recsize   : 1,932
     .CI size   : 4,096
     .Key   : 12 (position 1)

.The Job submission file (JB)

     .External name: $NMTU.$ROOT$FILEJB
     .Organization : UREL
     .Recsize   : 80
     .CI size   : 2,048

.The On-line DAF work file (PA)

     .External name: $NMTU.$ROOT$FILEPA
     .Organization : UIND
     .Recsize   : 100 minimum, 450 maximum
     .CI size   : 4,096
     .Key   : 37 (position 2)

Three other sequential files make up the back-up. These are:

.The Backup file (BB)

     .External name: $NMBU.$ROOT$FILEBB(n)
     .Organization : USEQ
     .Recsize   : 61 minimum, 350 maximum
     .CI size   : 2,048

.The Archived Journal file (BJ)

     .External name: $NMBU.$ROOT$FILEBJ(n)
     .Organization : USEQ
     .Recsize   : 180
     .CI size   : 2,048

.The Deactivated Archived file (BQ)

     .External name: $NMBU.$ROOT$FILEBQ
     .Organization : USEQ
     .Recsize   : 180
     .CI size   : 2,048

3. ENVIRONMENT

3.1. DEVELOPMENT ENVIRONMENT

OPERATING SITE CONFIGURATION

The system running the VA Pac system must have the following characteristics:

```
Operating system    :  GCOS-7 V3 A7 or higher
TDS level           :  GCOS-7 V3 A7 or higher
Supported messages  :  VIP7700, QUESTAR
                    :  VIP7800, IBM3270
```

3.2. ON-LINE ENVIRONMENT

ON-LINE ENVIRONMENT

The monitor in use is TDS/GCOS 7.

All user files are updated on-line. As such, they must be protected by the TDS journalization option (journal before).

The same applies to the screen backup file (DS80HE) and the relative file (DS80JB) required for the JOB function.

GENERAL INFORMATION - HOW THE SYSTEM RUNS

The general characteristics are:

. Two transaction codes are used. They call for the first TPR of the DSMS System (DS0AA0). This program does not directly manage any screen.

Both transaction code values are set by the user. However, the fourth character determines the type of terminal which may be used:

'1' : Only VIP7700 screens may be used

'2' : Only IBM3270 screens may be used

'3' : Only VIP7800 screens may be used

Other: No restriction as to the type of terminal. The display mode is automatically managed when the user accesses the transaction. The third character determines the language (F for French; E for English).

Each conversation starts and ends with a TPR execution, DS0AA0. Its source is supplied to allow for the insertion of standard conversation beginning and ending processings for on-site use, or to modifiy the standard transaction codes DSFO and DSEO.

. When an anomaly is managed by the system, a MAP ABEND is displayed. The display program of this screen is called by an ABORT call.

. Updates are serialized: the system manages simultaneous accesses to the database by placing all update TPRs in a wait queue.

. 'FT' entered in the OPERATION field on the DSMS initial screen ensures a correct exit. The following message is displayed: 'END OF CONVERSATION'.

UPPER AND LOWER-CASE PROCESSING

DSMS has its own character processing: all lower-case characters are automatically transformed into upper-case.

To use this automatic processing, the terminal must be configured for lower-case and the printer must print in lower case too. Before beginning on-line activity, the command '$*$LC ON' must be entered to specify that lower-case characters are to be used.

NOTE In batch processing, the transcoding of lower-case characters into upper-case is not possible. For example, an extraction request entered in lower-case is not recognized.

3.3. BATCH ENVIRONMENT

BATCH ENVIRONMENT

In batch mode, the system uses standard functions of the operating system, and UFAS access method.

The memory size required for the execution of batch procedures varies mainly according to the sizes of the buffers allocated to the files they use.

3.4. ACCESS METHOD

ACCESS METHOD

DSMS manages its files with relative and indexed without secondary index access methods.

All batch procedures include a DEALLOC PREALLOC step and, if necessary, a cataloging step if files are reloaded.

All files are protected against concurrent write accesses.

3.5. FILE SIZE

FILE SIZE

The total file size depends on the size of the applications managed by the system.

The global definition of an event or change is considered as:

- 20 records in the DA data file (1 record for screen definition + 19 records for the description lines)

  (NDA = 20)

- 10 records in the DX reference file,

  (NDX = 10)

- 10 records in the DC PAC element file (if VA Pac update is under DSMS control),

  (NDC = 10)

- each of these records creates 4 journal transactions in DJ. (NDJ = 4)

Example of calculation for a monthly growth of:

```
NBE:  300 events
NBA:  100 changes
NBT:  1000 table records (fixed)
Sites:  not significant.
```

Let:

```
NBDA be the DA records number
NBDX be the DX records number
NBDC be the DC records number
NBDJ be the DJ records number.
```

For six months:

```
NBDA = NBT + 6 x(NDA x(NBE + NBA))    --> 49,000 records

NBDX = 6 x (NDX x(NBE + NBA))         --> 24,000 records

NBDC = 6 x (NDC x NBA)                -->  6,000 records

For a 15-day journal:

NBDJ = ((NBDA+NBDX+NBDC) / 12) x NDJ  --> 26,340 records
```

FILE SIZE

File Descriptions:

(DA) DSMS data file: 80 to 340 bytes (average length of 220)

(DX) Cross-references file: its volume varies according to the use of the fields which manage these references when events and changes are created (phases, options, keywords, etc.). It contains 80 bytes per record.

(DC) VA Pac Elements file: its volume varies, whether DSMS is integrated in VA Pac or not. It it is integrated, it contains the references to the VA Pac elements entered on the -C and -M screens of changes (50 to 168 bytes per record, average length of 100).

(DJ) Journal file: it must be able to contain all the batch and on-line transactions occurred between two reinitializations. A DSMS transaction must correspond to a record in this journal file (180 bytes per record).

Example of Estimation for six months:

```
Volume of DA data file
     NBDA x 220 = 10,780,000 bytes.

Volume of DX cross-references file
     NBDX x  80 =  1,920,000 bytes.

Volume of DC VA Pac elements file
     NBDC x 100 =  600,000 bytes.

Volume of DJ journal file (for 15 days)
     NBDJ x 180 =  4,741,200 bytes.
```

SYSTEM SIZE

To estimate the disk space required when installing DSMS, the following tables show all the libraries and files needed, with their sizes (default values at installation).

The examples given for the installation represent a 30-million-byte environment.

SYSTEM FILES

Parameterized names	Contents	Millions of bytes
$NMLI.$LIBLM	Load-modules	3.5
$NMTD.$LIBSM	Sharable-modules	4.8
$NMTD.$LIBSL	Sources library	0.7
$NMLI.$LIBSU	Extracted trans. library	0.7
$NMBS.$ROOT$ROOTDE0	DSMS error messages	5.5
$NMLI.$LIBJCL	JCLs library	0.7
$NMLI.$LIBINV	INVOKEs library	0.7
$NMLI.SOURCE	Controls sources	0.7
$NMLI.ABOUT	Versions library	0.7
	TOTAL	18

EVOLVING FILES

Parameterized names	Contents		Millions of bytes
Databases:			
$NMTU.$ROOT$FILEDA	DSMS data		0.7
$NMTU.$ROOT$FILEDX	DSMS cross-references		0.7
$NMTU.$ROOT$FILEDC	VA Pac elements		0.7
$NMTU.$ROOT$ROOTDE	DSMS error messages		5.5
$NMTU.$ROOT$FILEDJ	DSMS journal		0.7
$NMTU.$ROOT$FILEHE	screen memorization		0.7
$NMTU.$ROOT$FILEJB			0.7
		TOTAL	9.7
Back-ups:			
$NMBU.$ROOT$FILEBB	Sequential images of DSMS database		0.7
$NMBU.$ROOT$FILEBJ	File of archived transactions		0.7
$NMBU.$ROOT$FILEBQ	deactivated transactions		0.7
		TOTAL	2.1
		TOTAL	11.8

ENVIRONMENT 3
FILE SIZE 5

4. BATCH PROCEDURES

4.1. 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.

4.2. 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).

JCL

Procedure names can be devided into two parts.

The first four characters represent the procedure type:

. DSEX operations procedures,

. DSIN intallation procedures,

. DSUS procédures utilisateurs,

. DSZZ JCL interpretation,

. DSIV call of procedures via INVOKE.

The last four characters represent the procedure.

examples :
. DSAV backup of DSMS database,
. ALLO file allocation,
. DEXT entity extraction.

All JCLs are parameterized. Some parameters, which exist in all procedures, do not appear in all the procedures' symbolics in use. Such is the case of the physical support names.

All batch procedures are started up by INVOKE from a DSIVXXXX member ($NMLI.$LIBINV library). There is one member of such type in each procedure.

example : . DSIVDRST call of the restoration procedure.

These members contain a call of the JCL to be submitted (INVOKE) and, for the procedures with user input, the contents of the user input in data lines (80 characters bounded by $INPUT and $ENDINPUT).

4.3. ABNORMAL EXECUTION

ABNORMAL EXECUTIONS

Input-output errors on the system or database files trigger the output of an error report (PAC7EI), the setting of switch-20 and the branching to the end of the procedure.

In most cases, the switch-20 and the type of operation help determine the cause of the abnormal ending (resources not available, file too small...).

If there is no PAC7EI report and if the ABEND type indicate a problem with DSMS programs, it is necessary to contact the IBM AD-TOOLS technical team and to keep all listings.

The PAC7EI report is printed via the PACABE sub-program. The user has the possibility to implement, at the end of each procedure, a specific error processing conditioned by the value of switch-20.

Since they are systematic, the setting of switch-20 and the branching are not indicated in the procedures' descriptions or in the flowcharts.

EXECUTION REPORT MESSAGE

Some DSMS programs output the following message in the execution report.

WARNING EX02. DECIMAL DATA OVERFLOW IN TASK xxxx AT ADDRESS
x.xx.xxxx

This message is output when numeric fields are truncated on the left of significant digits. This warning is not an execution anomaly and, in DSMS, some programming methods necessarily trigger this message output.

It is therefore advised not to take this message into account when executing DSMS programs. You should contact IBM if the warning counter is exceeded.

ERROR MANAGEMENT IN THE PROCEDURES

At the end of each procedure, the detection of an error stops the procedure with a return code different from zero. This code can be retrieved via the $? variable, right after the procedure submission command.

This prevents another procedure from being executed if there is a sequence of procedures.

5. JOURNAL ARCHIVING (DARC)

5.1. 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.

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).

CAUTION

With systems using generation files (MVS for instance), the +1 version of the archived transaction file might have been cataloged even if the procedure ended abnormally. In this case, the procedure must be executed again with the -1 version of the archived transaction file (not the 0 version) as input.

5.2. 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.

5.3. DESCRIPTION OF STEPS

DARC: DESCRIPTION OF STEPS

This procedure includes the following steps:

- Recognition of input : CREATE
- Journal archiving : PDS300
- Journal file definition : DEALLOC/PREALLOC
- Journal re-initialization : PDS320
- BJ generations shift : SHIFT

RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file defined under IOF. It is found at the beginning of all procedures which include user input.

. Input file : $INPUT

. Output file (TEMPRY) : TMBDARC
(recsize 80, Cisize 2048)

ARCHIVAL OF JOURNAL FILE: PDS300

This step executes the following:

. Updates the file of archived update transactions,

. Positions a flag in the Data file which represents the journal archiving,

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

.Input files:
-User transaction
TMBDARC temporary file
-Already archived transactions
PACDJB: External name $NMBU.$ROOT$FILEBJ
-Journal file to reinitialize
PACDDJ: External name $NMTU.$ROOT$FILEDJ
-Error message file

```
  PACDDE: External name $NMTU.$ROOT$ROOTDE

.Input-Output file:
 -Data file
  PACDDA: External name $NMTU.$ROOT$FILEDA

.Output files:
 -Archived update transactions
  PACDBJ: External name $NMBU.$ROOT$FILEBJ/G+1
 -Deactivated archived trans.
  PACDBQ: External name $NMBU.$ROOT$FILEBQ

.Output report:
 -Review of archival

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

CREATION OF JOURNAL FILE: DEALLOC/PREALLOC

This step performs a DEALLOC/PREALLOC of the journal (DJ).

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
  TMBDARC temporary file
 -Error-message file
  PACDDE: External name $NMTU.$ROOT$ROOTDE


.Input-Output file:
 -Data file
  PACDDA: External name $NMTU.$ROOT$FILEDA

.Output file:
 -Journal file to be reinitialized
  PACDDJ: External name $NMTU.$ROOT$FILEDJ

.Output report:
 -Reinitialization report
```

SHIFT OF ARCHIVED FILE GENERATIONS: SHIFT

This step performs a shift of the generations of the archive file (BJ).

5.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*                    DSEXDARC                         *';
COMM '*                    ========                         *';
COMM '*  D S M S            :  ARCHIVAL OF DSMS JOURNAL      *';
COMM '*                                                     *';
COMM '*******************************************************';
MVL  SIZEWK=2,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DARC,
     OF=(TMBDARC,&RFTM,TEMPRY,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
STEP PDS300,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  100;
     ASG PACDMB,TMBDARC,TEMPRY,&RFTM,END=PASS;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU;
     ASG PACDDJ,$NMDJ.$ROOT$FILEDJ,&RFDJ;
     ASG PACDJB,$NMBU.$ROOT$FILEBJ,&RFBU;
     ASG PACDBJ,$NMBU.$ROOT$FILEBJ/G+1,&RFBU;
     ASG PACDBQ,$NMBU.$ROOT$FILEBQ,&RFBU;
     ASG PACDRU,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP ERR,SW30,EQ,1;
COMM '*** ALLOCATION : DJ ***';
IV   DSINALDJ ($NMLI.$LIBJCL,&RFLI);
STEP PDS320,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     ASG PACDMB,TMBDARC,TEMPRY,&RFTM;
     ASG PACDDJ,$NMDJ.$ROOT$FILEDJ,&RFDJ;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU;
     ASG PACDRU,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT $NMBU.$ROOT$FILEBJ;
JUMP END;
ERR:
SEND ' DSEXDARC - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

6. PRINTING OF QUERIES AND OUTPUT REPORTS (DPRT)

6.1. 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'.

6.2. 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	mmmmmm	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, ky, queries, reports print requests
8	OK but erroneous queries or requests
12	Fatal error
16	Sort error

6.3. 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:

```
SYMBOLICS IN USE

-------------------------------------------------------------
! SYMBOLICS! MEANING                                <EXAMPLE>!
!------------------------------------------------------------!
! &SIZEW1  ! reserved for temporary files              01  !
! &SIZEW2  ! reserved for temporary files              01  !
! &SIZEW3  ! reserved for temporary files              01  !
! &SIZEW4  ! reserved for temporary files              01  !
! &SIZEWK  ! reserved for internal sorts               01  !
-------------------------------------------------------------


DESCRIPTION OF STEPS

This procedure includes the following steps:

   . Recognition of input             : CREATE

   . Printing                         : PDSB


RECOGNITION OF INPUT: CREATE

This utility creates a temporary file using the
file defined under IOF. It is found at the beginning of
all procedures which include user input.

. Input file                                : $INPUT

. Output file (TEMPRY)                      : TMBDPRT
   (recsize 80, Cisize 2048)
```

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

```
PRINTING:  PDSB

.Permanent input files:
 -Data file
  DS80IA  : External name : $NMTU.$ROOT$FILEDA
 -VA Pac element file
  PB80DC  : External name : $NMTU.$ROOT$FILEDC
 -Error message file
  DS80IE  : External name : $NMTU.$ROOT$ROOTDE
```

.Input file:
 -User Queries
  PACDMB :   TMBDPRT

.Work files:
 -Print requests
  PACDKD
 -Queries
  PACDKQ
 -Temporary files
  PACDQR PACDQJ

  PACDW1 PACDW2
  PACDW3 PACDW4

.Output reports:
 -Flow report
  PACDIA
 -List of Queries and requests
  PACDIB
 -Print of tables and keywords
  PACDID
 -Report of Query extractions
  PACDIQ
 -Print of Query extractions
  PACDQI
 -Print of Queries/Reports
  PACDRQ
 -Print of control cards
  PACDJQ

6.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*                                                     *';
COMM '*  D S M S       : PRINTING                           *';
COMM '*                                                     *';
COMM '*                    DSUSDPRT                         *';
COMM '*                                                     *';
COMM '*******************************************************';
MVL  SIZEWK=5,
     SIZEW1=01,SIZEW2=01,SIZEW3=01,SIZEW4=01,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     RFTM='DVC=$DVTM,MD=$MDTM',
     PGE='E',PGLNG=&PG$LANG;
CR   IF=*DPRT,
     OF=(TMBDPRT,&RFTM,TEMPRY,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
COMM '*** PDSB   ***';
STEP PDSB&PGLNG,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  500;
     ASG PACDMB,TMBDPRT,TEMPRY,&RFTM;
     ASG DS80IA,$NMTU.$ROOT$FILEDA,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF DS80IA,READLOCK=STAT;
     ASG PB80DC,$NMTU.$ROOT$FILEDC,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PB80DC,READLOCK=STAT;
     ASG DS80IE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF DS80IE,READLOCK=STAT;
     ASG PACDW1,TPACDW1,TEMPRY,&RFTM,END=PASS;
     ALC PACDW1,SZ=&SIZEW1,INCRSZ=1,UNIT=CYL;
     DEF PACDW1,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW2,TPACDW2,TEMPRY,&RFTM,END=PASS;
     ALC PACDW2,SZ=&SIZEW2,INCRSZ=1,UNIT=CYL;
     DEF PACDW2,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW3,TPACDW3,TEMPRY,&RFTM,END=PASS;
     ALC PACDW3,SZ=&SIZEW3,INCRSZ=1,UNIT=CYL;
     DEF PACDW3,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW4,TPACDW4,TEMPRY,&RFTM,END=PASS;
     ALC PACDW4,SZ=&SIZEW4,INCRSZ=1,UNIT=CYL;
     DEF PACDW4,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDKD,TPACDKD,TEMPRY,&RFTM,END=PASS;
     ASG PACDKQ,TPACDKQ,TEMPRY,&RFTM,END=PASS;
     ASG PACDQR,TPACDQR,TEMPRY,&RFTM,END=PASS;
     ASG PACDQJ,TPACDQJ,TEMPRY,&RFTM,END=PASS;
     ASG PACDIA,SYS.OUT;
     ASG PACDIB,SYS.OUT;
     ASG PACDID,SYS.OUT;
     ASG PACDIQ,SYS.OUT;
     ASG PACDJQ,SYS.OUT;
     ASG PACDQI,SYS.OUT;
     ASG PACDRQ,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
```

```
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP END;
ERR:
SEND ' DSUSDPRT - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

6.5. ON-LINE SUBMISSION

USE OF THE JOB FUCTION

The submission of the JCL stream triggered by the 'JOB' command is indirect. It first updates a relative UFAS buffer file (DS80JB) and executes a parameterized 'submitter'.

The first record of the buffer file is constituted of the submitter identification (library sub-file) and of the number of the last record written in the buffer file.

When the 'JOB' is sumbitted, DSMS reads the first record of this file in order to find the address of the last record written; the JCL stream is built and is written on continuation in the buffer file. The first record of the buffer file is updated with the address of the last record written.

DSMS then calls the "SUBJOB" system routine and sends the submitter with the following parameters:

START = number of the first record of the stream in the buffer.

HALT = number of records which make up the stream.

USER = DSMS userid of the user.

SYSUSER = IOF code of the user.

At this stage, DSMS branches off to the user.

The submitter performs two steps.

1) With the CREATE utility, it re-builds the steam via the START and HALT parameters and creates a temporary file which only contains the JCL.

2) With the RUN command, it triggers the execution of the JOB by passing on the USER parameter to it.

EXAMPLE OF JCL ENTERED BY A USER UNDER DSMS

DSMS SCREEN OPTION : C1 CHOICE : LJQ

```
T LIN  : V

A 050  : V   $JOB DSMSDPRT;
A 060  : V   VL DV='DVC=$DVLI',MD='$MDLI';
A 100  : V   SEND '==> BEGINNING OF DSMS QUERY <==' &USER;
Z 120  : V   IV DSUSDPRT ($NMLI.$LIBJCL,&MD,&DV);
A 900  : V   $INPUT DPRT TYPE=DATA;
Z 100  : V   $ENDINPUT DPRT;
Z 200  : V   SEND '==>   END OF DSMS QUERY   <==' &USER;
Z 300  : V   $ENDJOB;
```

7. DATABASE RESTORATION (DRST)

7.1. 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.

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.

7.2. 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.

7.3. DESCRIPTION OF STEPS

DRST: DESCRIPTION OF STEPS

```
DESCRIPTION OF STEPS

This procedure includes the following steps:

     . Recognition of input             : CREATE

     . Validation of journal contents   : PDS380

     . Definition of files              : DEALLOC/PREALLOC

     . Restoration of files             : PDS400

     . Retrieval of archived journal    : PDS450


RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file defined
under IOF. It is found at the beginning of all
procedures which include user input.

. Input file                              : $INPUT

. Output file (TEMPRY)                    : TMBDRST
   (recsize 80, Cisize 2048)


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
  PACDDJ  :  External name &INDUV..&ROOTD.$ROOT2.0
 -Error message file
  PACCDE  :  External name $NMTU.$ROOT$ROOTDE

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

.Return codes:
 .0: the journal file has been archived
 .4: the journal file has not been archived
     (no DRST step is executed).
```

DEFINITION OF FILES: DEALLOC/PREALLOC

This step is executed only if the journal file has been archived. It performs a DEALLOC/PREALLOC of the files.

. Defined files:
- Data file
  External name : $NMTU.$ROOT$FILEDA
- VA PAC elements file
  External name : $NMTU.$ROOT$FILEDC
- Journal file
  External name : $NMTU.$ROOT$FILEDJ
- Cross-references file
  External name : $NMTU.$ROOT$FILEDX

DATABASE RESTORATION: PDS400

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

.Permanent input files:
-Backup of the files
 PACDBB : External name $NMBU.$ROOT$FILEBB
-Error message file
 PACDDE : External name $NMTU.$ROOT$ROOTDE

.Permanent output files:
-Data file
 PACDDA : External name $NMTU.$ROOT$FILEDA
-VA Pac element file
 PACDDC : External name $NMTU.$ROOT$FILEDC
-Journal file
 PACDDJ : External name $NMTU.$ROOT$FILEDJ
-Cross-reference file
 PACDDX : External name $NMTU.$ROOT$FILEDX

.Input transaction file:
-User transactions
 PACDMB : TMBDRST temporary file

.Output file:
-Work file (2 records)
 PACDMS : TPACDMS temporary file (recsize 80)

.Output report:
-Restoration report
 PACDRU

RETRIEVAL OF ARCHIVED JOURNAL: PDS450

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

.Permanent input-output files:
-Data file
 PACDDA : External name $NMTU.$ROOT$FILEDA
-VA Pac element file
 PACDDC : External name $NMTU.$ROOT$FILEDC
-Cross-reference file
 PACDDX : External name $NMTU.$ROOT$FILEDX

```
.Input files:
 -Work file      (2 records)
  PACDMS  :   TPACDMS temporary file (recsize 80)
 -Error message file
  PACDDE  :   External name $NMTU.$ROOT$ROOTDE

.Input archived file:
 -Archiving of the journal to retrieve
  PACDBJ  :    External name $NMBU.$ROOT$FILEBJ

.Output report:
 -Update report
  PACDRU
```

7.4. EXECUTION JCL

```
COMM '*****************************************************';
COMM '*                   DSEXDRST                        *';
COMM '*                   ========                        *';
COMM '*  D S M S     : LOADING-RESTORATION OF DSMS DATABASE*';
COMM '*                                                   *';
COMM '*                                                   *';
COMM '*****************************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM',
     PGE='1',PGF='0',PGLNG=&PG$LANG;
CR   IF=*DRST,
     OF=(TMBDRST,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
COMM '*** CTL SUR DJ ***';
FILLIST INFILE=($NMDJ.$ROOT$FILEDJ,&RFDJ);
JUMP CRE SEV GE 3;
COMM '*** PDS380 ***';
STEP PDS38&PGLNG,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  30;
     ASG PACDDJ,$NMDJ.$ROOT$FILEDJ,&RFDJ;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PACDDE,READLOCK=STAT;
     ASG PACDRU,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP END,SW30,EQ,1;
CRE:
COMM '*** ALLOCATION : DA,DC,DJ,DX ***';
IV   DSINALDA ($NMLI.$LIBJCL,&RFLI);
IV   DSINALDC ($NMLI.$LIBJCL,&RFLI);
IV   DSINALDJ ($NMLI.$LIBJCL,&RFLI);
IV   DSINALDX ($NMLI.$LIBJCL,&RFLI);
COMM '*** PDS400 ***';
STEP PDS400,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDMB,TMBDRST,TEMPRY,&RFTM;
     ASG PACDDJ,$NMDJ.$ROOT$FILEDJ,&RFDJ;
     ASG PACDBB,$NMBU.$ROOT$FILEBB,&RFBU;
     ASG PACDDC,$NMTU.$ROOT$FILEDC,&RFTU;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU;
     ASG PACDDX,$NMTU.$ROOT$FILEDX,&RFTU;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PACDDE,READLOCK=STAT;
     ASG PACDMS,TPACDMS,TEMPRY,&RFTM,END=PASS;
     ASG PACDRU,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
```

```
    JUMP END,SW30,EQ,1;
    COMM '*** PDS450 ***';
    STEP PDS450,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
         SZ 60;
         ASG PACDMS,TPACDMS,TEMPRY,&RFTM;
         ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU;
         ASG PACDDC,$NMTU.$ROOT$FILEDC,&RFTU;
         ASG PACDDX,$NMTU.$ROOT$FILEDX,&RFTU;
         ASG PACDBJ,$NMBU.$ROOT$FILEBJ,&RFBU;
         ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU,
             ACCESS=READ,SHARE=MONITOR;
         DEF PACDDE,READLOCK=STAT;
         ASG PACDRU,SYS.OUT;
         ASG PAC7EI,SYS.OUT;
    ESTP;
    JUMP ERR,SW20,EQ,1;
    JUMP END;
    ERR:
    SEND ' DSEXDRST - ABNORMAL END OF RUN (I/O ERROR) ';
    LET  SEV 3;
    END:
```

8. DATABASE BACKUP (DSAV)

8.1. 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.

8.2. INPUT - PROCESSING - RESULTS

USER INPUT

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

REPORT RESULTS

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

OUTPUT RESULT

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.

NOTE:

The DSAV procedure increments the current session number.

8.3. DESCRIPTION OF STEPS

DSAV: DESCRIPTION OF STEPS

This procedure includes the following steps:

- Recognition of input : CREATE
- Check of data integrity : PDSBAS
- DSMS database backup : PDS500
- Shift of BJ generations : SHIFT

RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file defined under IOF. It is found at the beginning of all procedures which include user input.

. Input file : $INPUT

. Output file (TEMPRY) : TMBDSAV
(recsize 80, Cisize 2048)

DATABASE INTEGRITY CHECK: PDSBAS

.Permanent input files:
-Data file
PACDDA : External name $NMTU.$ROOT$FILEDA
-Error message file
PACDDE : External name $NMTU.$ROOT$ROOTDE

.Output report
-Validity report
PACDRS

.Return code
Switch 30 0: valid Database
1: Invalid Database. The procedure stops.

DATABASE BACKUP: PDS500

.Input-Output file:
-Data file
PACDDA : External name $NMTU.$ROOT$FILEDA

```
.Permanent Input files:
 -VA Pac element file
  PACDDC  : External name $NMTU.$ROOT$FILEDC
 -Cross-reference file
  PACDDX  : External name $NMTU.$ROOT$FILEDX
 -Error message file
  PACDDE  : External name $NMTU.$ROOT$ROOTDE

.Input transaction file:
 -User transactions
  PACDMB  : TMBDSAV temporary file

.Output file:
 -Sequential image of files
  External name $NMBU.$ROOT$FILEBB/G+1

.Output report:
 -Backup report
```

SHIFT OF ARCHIVED FILE GENERATIONS: SHIFT

This step performs a shift of the archive file generations (BB).

8.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*                    DSEXDSAV                         *';
COMM '*                    ========                         *';
COMM '*  D S M S        : BACKUP OF THE DSMS DATABASE       *';
COMM '*                                                     *';
COMM '*                                                     *';
COMM '*******************************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DSAV,
     OF=(TMBDSAV,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
COMM '*** PDSBAS ***';
STEP PDSBAS,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ 30;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU;
     ASG PACDRS,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP END,SW30,EQ,1;
COMM '*** PDS500 ***';
STEP PDS500,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU;
     ASG PACDDC,$NMTU.$ROOT$FILEDC,&RFTU;
     ASG PACDDX,$NMTU.$ROOT$FILEDX,&RFTU;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU;
     ASG PACDBB,$NMBU.$ROOT$FILEBB/G+1,&RFBU;
     ASG PACDMB,TMBDSAV,TEMPRY,&RFTM;
     ASG PACDRU,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT $NMBU.$ROOT$FILEBB;
JUMP END;
ERR:
SEND ' DSEXDSAV - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

9. REORGANIZATION OF CROSS-REFERENCE FILE (DREO)

9.1. 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.

9.2. 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.

9.3. DESCRIPTION OF STEPS

SYMBOLICS IN USE

SYMBOLICS	MEANING	<EXAMPLE>
&SIZEW1	reserved for temporary files	01
&SIZEW2	reserved for temporary files	01
&SIZEW3	reserved for temporary files	01
&SIZEW4	reserved for temporary files	01
&SIZEW5	reserved for temporary files	01
&SIZEW6	reserved for temporary files	01
&SIZEWK	reserved for internal sorts	01

DESCRIPTION OF STEPS

This procedure includes the following steps:

- Recognition of input : CREATE
- Building of indexes (except keywords) : PDSR10
- Building of keyword indexes : PDSR20
- Merging of indexes : PDSR30
- General merging for backup : PDSR40
- Shift of BB generations : SHIFT

RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file defined under IOF. It is found at the beginning of all procedures which include user input.

. Input file : $INPUT

. Output file (TEMPRY) : TMBDREO
(recsize 80, Cisize 2048)

BUILDING OF INDEXES (not keywords): PDSR10

.Input file:
-Transactions
CARTE : TMBDREO temporary file

.Permanent Input files:
-DSMS database backup
PACDBB : External name $NMBU.$ROOT$FILEBB
-Error message file
PACDDE : External name $NMTU.$ROOT$ROOTDE

.Work files:
-Data and VA Pac elements
PACDW1

```
 -Keywords and keyword references
  PACDW2
 -Cross-references (not keywords)
  PACDW3
 -Sort files

.Output reports:
 -Inconsistencies in DSMS data
  PACDRH
 -Reorganization report
  PACDRK

BUILDING OF KEYWORD INDEXES:  PDSR20

.Work files:
 -Keywords and keyword references
  TPACDW2
 -Keywords
  TPACDW4
 -Keyword references
  TPACDW5
 -Sort files

MERGE OF INDEXES:  PDSR30

.Work files:
 -Cross-references (except keywords)
  TPACDW3
 -Keyword references
  TPACDW5
 -Keyword references
  TPACDW6
 -Sort files

GENERAL MERGE FOR BACKUP:  PDSR40

.Work files:
 -Data and VA Pac elements
  TPACDW1
 -Keywords
  TPACDW4
 -Keyword references
  TPACDW6
 -Sort files

.Permanent input file:
 -Error message file
  PACDDE  :   External name $NMTU.$ROOT$ROOTDE

.Permanent output file:
 -Reorganized DSMS database backup
  PACDBB  :   External name $NMBU.$ROOT$FILEBB/G+1

.Output report:
 -Reorganization report
  PACDRR


SHIFT OF ARCHIVED FILE GENERATIONS: SHIFT

This step performs a shift of the archive file generations
(BB).
```

9.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*                    DSEXDREO                         *';
COMM '*                    ========                         *';
COMM '*  D S M S        : DSMS DATABASE REORGANIZATION      *';
COMM '*                                                     *';
COMM '*                                                     *';
COMM '*******************************************************';
MVL  SIZEW1=01,SIZEW2=01,SIZEW3=01,SIZEW4=02,SIZEW5=01,SIZEW6=01,
     SIZEWK=01,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DREO,
     OF=(TMBDREO,&RFTM,TEMPRY,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
COMM '*** PDSR10 ***';
STEP PDSR10,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG CARTE,TMBDREO,TEMPRY,&RFTM;
     ASG PACDBB,$NMBU.$ROOT$FILEBB,&RFBU;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PACDDE,READLOCK=STAT;
     ASG PACDW1,TPACDW1,TEMPRY,&RFTM,END=PASS;
     ALC PACDW1,SZ=&SIZEW1,UNIT=CYL;
     DEF PACDW1,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW2,TPACDW2,TEMPRY,&RFTM,END=PASS;
     ALC PACDW2,SZ=&SIZEW2,UNIT=CYL;
     DEF PACDW2,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW3,TPACDW3,TEMPRY,&RFTM,END=PASS;
     ALC PACDW3,SZ=&SIZEW3,UNIT=CYL;
     DEF PACDW3,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDRK,SYS.OUT;
     ASG PACDRH,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
COMM '*** PDSR20 ***';
STEP PDSR20,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDW2,TPACDW2,TEMPRY,&RFTM,END=PASS;
     DEF PACDW2,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW4,TPACDW4,TEMPRY,&RFTM,END=PASS;
     ALC PACDW4,SZ=&SIZEW4,UNIT=CYL;
     DEF PACDW4,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW5,TPACDW5,TEMPRY,&RFTM,END=PASS;
     ALC PACDW5,SZ=&SIZEW5,UNIT=CYL;
     DEF PACDW5,CISIZE=$CISEQ,NBBUF=1;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
COMM '*** PDSR30 ***';
```

```
    STEP PDSR30,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
         SZ  60;
         ASG PACDW3,TPACDW3,TEMPRY,&RFTM,END=PASS;
         DEF PACDW3,CISIZE=$CISEQ,NBBUF=1;
         ASG PACDW5,TPACDW5,TEMPRY,&RFTM,END=PASS;
         DEF PACDW5,CISIZE=$CISEQ,NBBUF=1;
         ASG PACDW6,TPACDW6,TEMPRY,&RFTM,END=PASS;
         ALC PACDW6,SZ=&SIZEW6,UNIT=CYL;
         DEF PACDW6,CISIZE=$CISEQ,NBBUF=1;
         ASG PAC7EI,SYS.OUT;
         SWK WKDISK=(SZ=&SIZEWK,&RFTM);
    ESTP;
    JUMP ERR,SW20,EQ,1;
    COMM '*** PDSR40 ***';
    STEP PDSR40,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
         SZ  60;
         ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU,
             ACCESS=READ,SHARE=MONITOR;
         DEF PACDDE,READLOCK=STAT;
         ASG PACDBB,$NMBU.$ROOT$FILEBB/G+1,&RFBU;
         ASG PACDW1,TPACDW1,TEMPRY,&RFTM;
         DEF PACDW1,CISIZE=$CISEQ,NBBUF=1;
         ASG PACDW4,TPACDW4,TEMPRY,&RFTM;
         DEF PACDW4,CISIZE=$CISEQ,NBBUF=1;
         ASG PACDW6,TPACDW6,TEMPRY,&RFTM;
         DEF PACDW6,CISIZE=$CISEQ,NBBUF=1;
         ASG PACDRR,SYS.OUT;
         ASG PAC7EI,SYS.OUT;
         SWK WKDISK=(SZ=&SIZEWK,&RFTM);
    ESTP;
    JUMP ERR,SW20,EQ,1;
    SHIFT $NMBU.$ROOT$FILEBB;
    JUMP END;
    ERR:
    SEND ' DSEXDREO - ABNORMAL END OF RUN (I/O ERROR) ';
    LET  SEV 3;
    END:
```

10. EXTRACTION FROM VA PAC ARCHIVED JOURNAL (DEXP)

10.1. 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.

10.2. 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.

10.3. DESCRIPTION OF STEPS

DEXP: DESCRIPTION OF STEPS

SYMBOLICS IN USE

SYMBOLICS	MEANING	<EXAMPLE>
&PAC7PJ	VA Pac archived journal file	
&SIZEWK	reserved for internal sort	01
&USER	DSMS userid	IBM

DESCRIPTION OF STPEPS

This procedure includes the following steps:

. Recognition of input : CREATE

. Printing : PDS600

. Printing of update transactions : PDS610

RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file defined under IOF. It is found at the beginning of all procedures which include user input.

. Input file : $INPUT

. Output file (TEMPRY) : TMBDEXP
(recsize 80, Cisize 2048)

TRANSACTION EXTRACTION AND FORMATTING: PDS600

.Permanent imput files:
-Data file
PACDDA : External name $NMTU.$ROOT$FILEDA

-Error message file
PACDDE : External name $NMTU.$ROOT$ROOTDE
-VA Pac archived journal
PAC7PJ External name &PAC7PJ

.Input transaction file:
-User transactions
PACDMB : TMBDEXP temporary file

.Sort files:

.Output file:

```
 -DUPT update transaction file
 TPACDMV temporary file (recsize 250)

.Output report:
 -Report on selection request
  PACDRU

.Return codes:
 . 0 : No error and no list requested
 .04 : No error and printing of transactions list
 .08 : Error on the user identification or parameter line
 .12 : Input/outpout error on a file

Storing of the result in source library

PRINTING OF DSMS UPDATE TRANSACTIONS:  PDS610

.Permanent input files:
 -Data file
  PACDDA  :    External name $NMTU.$ROOT$FILEDA


 -Error message file
  PACDDE  :    External name $NMTU.$ROOT$ROOTDE

.Input File:
 -DSMS update transactions file
  PACDMV  :    TPACDMV temporary file (recsize 250)

.Output report:
 -List of update transactions
  PACDRU

.Return codes:
   0 : No error
  12 : Input/output error on a file
```

10.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*  D S M S     : EXTRACTION OF TRANSACTIONS FROM       *';
COMM '*                JOURNAL FILE (DJ)                    *';
COMM '*                DSUSDEXP                             *';
COMM '*                                                     *';
COMM '*                                                     *';
COMM '*  OUTPUT :                                           *';
COMM '*    OUTPUT TRANSACTIONS ARE STORED IN THE LIBRARY :  *';
COMM '*    $NMLI.$LIBSU                                     *';
COMM '*******************************************************';
MVL  PAC7PJ='PACPJ20',USER='$USER',SIZEWK=1,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DEXP,
     OF=(TMBDEXP,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
STEP PDS600,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDMB,TMBDEXP,TEMPRY,&RFTM;
     ASG PACDMV,TPACDMV,TEMPRY,&RFTM,END=PASS;
     DEF PACDMV,CISZ=4096,RECSZ=250,RECFORM=FB;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PACDDE,READLOCK=STAT;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PACDDA,READLOCK=STAT;
     ASG PAC7PJ,&PAC7PJ,&RFBU;
     ASG PACDRU,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP END,SW30,EQ,0;
STEP PDS610,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ 60;
     ASG PACDMV,TPACDMV,TEMPRY,&RFTM,END=PASS;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PACDDE,READLOCK=STAT;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PACDDA,READLOCK=STAT;
     ASG PAC7EI,SYS.OUT;
     ASG PACDRU,SYS.OUT;
ESTP;
LMN  SL INFILE=(TPACDMV,TEMPRY,&RFTM),
     LIB=($NMLI.$LIBSU,&RFLI),
     COM='MV INFILE:MBDUPD_DEXP'&USER',INFORM=SARF,
          TYPE=DAT,NUMBER=(1,1),REPLACE;';
JUMP ERR,SW20,EQ,1;
```

```
    JUMP END;
    ERR:
    SEND ' DSUSDEXP - ABNORMAL END OF RUN (I/O ERROR) ';
    LET  SEV 3;
    END:
```

11. EXTRACTION OF ENTITIES (DEXT)

11.1. 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.

11.2. 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 a '*' line or on a command line
12	Input/output error or inconsistent DSMS Database
16	Sort error

11.3. 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:

```
DESCRIPTION OF STEPS

This procedure includes the following steps:

      . Recognition of input              : CREATE

      . Extraction of entities            : PDSEX

      . Integration of the result file in source library


RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file defined
under IOF. It is found at the beginning of all procedures
which include user input.

. Input file                              : $INPUT

. Output file (TEMPRY)                    : TMBDEXT
   (recsize 80, Cisize 2048)
 - Data file                              : PACDDA

EXTRACTIONS:  PDSEX

.Permanent input files:
 -Data file
  DS80IA      : $NMTU.$ROOT$FILEDA
 -VA Pac element file
  PB80DC      : $NMTU.$ROOT$FILEDC
 -Error message file
  DS80IE      : $NMTU.$ROOT$FILEDE

.Input transaction file:
 -Extraction requests
  TMBDEXT temporary file

.Work files:
 -Queries
  TPACDKQ
```

-Temporary files
TPACDW0 TPACDW1 TPACDW2 TPACDW3 TPACDW4 TPACDW5 TPACDWI

.Output reports:
-Flow report
PACDIA
-Extraction request report
PACDRU

.Sort files:

.Output file:
-Extracted batch transactions
TPACDIM temporary file

11.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*  D S M S     : EXTRACTION OF DSMS ENTITIES          *';
COMM '*                DSUSDEXT                             *';
COMM '*                                                     *';
COMM '*                                                     *';
COMM '*  OUTPUT :                                           *';
COMM '*    OUTPUT TRANSACTIONS ARE STORED IN THE LIBRARY :  *';
COMM '*    $NMLI.$LIBSU                                     *';
COMM '*******************************************************';
MVL  USER='$USER',SIZEWK=1,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM',
     PGE='E',PGLNG=&PG$LANG;
CR   IF=*DEXT,
     OF=(TMBDEXT,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
STEP PDSEX&PGLNG,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  100;
     ASG PACDMB,TMBDEXT,TEMPRY,&RFTM;
     ASG PACDIM,TPACDIM,TEMPRY,&RFTM,END=PASS;
     DEF PACDIM,CISZ=4096,RECSZ=250,RECFORM=FB;
     ASG PB80DC,$NMTU.$ROOT$FILEDC,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PB80DC,READLOCK=STAT;
     ASG DS80IA,$NMTU.$ROOT$FILEDA,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF DS80IA,READLOCK=STAT;
     ASG DS80IE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF DS80IE,READLOCK=STAT;
     ASG PACDKQ,TPACDKQ,TEMPRY,&RFTM,END=PASS;
     ASG PACDWI,TPACDWI,TEMPRY,&RFTM,END=PASS;
     ALC PACDWI,SZ=&SIZEWK,UNIT=CYL;
     DEF PACDWI,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW0,TPACDW0,TEMPRY,&RFTM,END=PASS;
     ALC PACDW0,SZ=&SIZEWK,UNIT=CYL;
     DEF PACDW0,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW1,TPACDW1,TEMPRY,&RFTM,END=PASS;
     ALC PACDW1,SZ=&SIZEWK,UNIT=CYL;
     DEF PACDW1,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW2,TPACDW2,TEMPRY,&RFTM,END=PASS;
     ALC PACDW2,SZ=&SIZEWK,UNIT=CYL;
     DEF PACDW2,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW3,TPACDW3,TEMPRY,&RFTM,END=PASS;
     ALC PACDW3,SZ=&SIZEWK,UNIT=CYL;
     DEF PACDW3,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW4,TPACDW4,TEMPRY,&RFTM,END=PASS;
     ALC PACDW4,SZ=&SIZEWK,UNIT=CYL;
     DEF PACDW4,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW5,TPACDW5,TEMPRY,&RFTM,END=PASS;
     ALC PACDW5,SZ=&SIZEWK,UNIT=CYL;
```

```
         DEF PACDW5,CISIZE=$CISEQ,NBBUF=1;
         ASG PACDRU,SYS.OUT;
         ASG PACDIA,SYS.OUT;
         ASG PAC7EI,SYS.OUT;
         SWK WKDISK=(SZ=&SIZEWK,&RFTM);
    ESTP;
    JUMP ERR,SW20,EQ,1;
    JUMP ERR,SW30,EQ,1;
    LMN  SL INFILE=(TPACDIM,TEMPRY,&RFTM),
         LIB=($NMLI.$LIBSU,&RFLI),
         COM='MV INFILE:MBDUPD_DEXT'&USER',INFORM=SARF,
              TYPE=DAT,NUMBER=(1,1),REPLACE;';
    JUMP ERR,SW20,EQ,1;
    JUMP END;
    ERR:
    SEND ' DSUSDEXT - ABNORMAL END OF RUN (I/O ERROR) ';
    LET  SEV 3;
    END:
```

EXTRACTION OF ENTITIES (DEXT) 11
EXECUTION JCL 4

12. EXTRACTION OF TABLES FOR EXTERNAL LISTS (DEXH)

12.1. 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.

12.2. 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.

12.3. DESCRIPTION OF STEPS

DEXH: DESCRIPTION OF STEPS

SYMBOLICS IN USE

SYMBOLICS	MEANING	<EXAMPLE>
&SIZEWK	reserved for internal sort	01
&USER	DSMS userid	IBM

DESCRIPTION OF STEPS

This procedure includes the following steps:

. Recognition of input : CREATE

. Extraction-formatting of transactions : PDSXTH

RECOGNITION OF INPUT: CREATE

This utitity creates a temporary file from a file defined under IOF. It is found at the beginning of all procedures which include user input.

EXTRACTION FOR LOCAL LISTS: 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
  DS80IA : External name $NMTU.$ROOT$FILEDA
 -Error message file
  DS80IE : External name $NMTU.$ROOT$ROOTDE

.Input transaction file:
 -User check
  TMBDEXH temporary file

.Output file:
 -Extracted tables
  PACDMV

.Output report:
 -Extraction report
  PACDRH

.Sort files:

Storing of the result in source library

12.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*  D S M S     : EXTRACTION OF DSMS TABLES             *';
COMM '*                DSUSDEXH                             *';
COMM '*                                                     *';
COMM '*  OUTPUT :                                           *';
COMM '*    OUTPUT TRANSACTIONS ARE STORED IN THE LIBRARY :  *';
COMM '*    $NMLI.$LIBSU                                     *';
COMM '*******************************************************';
MVL  USER='$USER',SIZEWK=1,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DEXH,
     OF=(TMBDEXH,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
STEP PDSXTH,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDMB,TMBDEXH,TEMPRY,&RFTM;
     ASG PACDMV,TPACDMV,TEMPRY,&RFTM,END=PASS;
     DEF PACDMV,CISZ=2048,RECSZ=100,RECFORM=FB;
     ASG DS80IE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF DS80IE,READLOCK=STAT;
     ASG DS80IA,$NMTU.$ROOT$FILEDA,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF DS80IA,READLOCK=STAT;
     ASG PACDRH,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
LMN  SL INFILE=(TPACDMV,TEMPRY,&RFTM),
     LIB=($NMLI.$LIBSU,&RFLI),
     COM='MV INFILE:MBPAW_DEXH'&USER',INFORM=SARF,
          TYPE=DAT,NUMBER=(1,1),REPLACE;';
JUMP ERR,SW20,EQ,1;
JUMP END;
ERR:
SEND ' DSUSDEXH - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

13. BATCH UPDATE OF ENTITIES (DUPT)

13.1. 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).

13.2. 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 enti-ty 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 trans- action
		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.

13.3. DESCRIPTION OF STEPS

DUPT: DESCRIPTION OF STEPS

SMBOLICS IN USE

SYMBOLICS	MEANING	<EXAMPLE>
&USER	USER CODE	<$USER>

PROCEDURE SUBMISSION

This procedure can be submitted in two different ways:

EJ DSIVDUPT to take as user input the transactions entered
on the $INPUT line of DSIVDUPT

EJ DSIVDUPT VL=(DEXP, DEXT ou DXBJ) to take as user input
the transactions producted by DEXP, DEXT or DXBJ
External name : $NMTU.$ROOT$ROOTDE

DESCRIPTION OF STEPS

This procedure includes the following steps:

.Recognition of input : CREATE

.Database integrity check : PDSBAS

.Update : PDSUP0

RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from the DATA lines
defined in the DSIVDUPT call procedure, or from a file
output from the extraction procedures.

.Input file : $INPUT
or MBDUPD_&1&USER
.Output file : TMBDUPT

DATABASE INTEGRITY CHECK: PDSBAS

. Permanent input file:
- Data file : PACDDA
External name : $NMTU.$ROOT$FILEDA
- Error message file : PACDDE

. Output reports:

```
 - Update report                          : PACDRS
 - Anomaly report                         : PAC7EI

. Return code : Switch-30
  - 0 : valid database.
  - 1 : invalid database. In this case, no other step is
        executed.


UPDATE OF THE DSMS DATABASE:  PDSUP0

.Permanent input-output files
 -Data file
 PACDDA  :    External name $NMTU.$ROOT$FILEDA
 -Va Pac element file
 PACDDC  :    External name $NMTU.$ROOT$FILEDC
 -Cross-reference file
 PACDDX  :    External name $NMTU.$ROOT$FILEDX

.Permanent input file
 -Error message file
 PACDDE  :    External name $NMTU.$ROOT$ROOTDE

.Input transaction file
 -Update transactions obtained via
  the DEXP procedure
  PACDIM  :   External name TMBDUPT

.Output file
 -Journal file
  PACDDJ  :    External name $NMTU.$ROOT$FILEDJ

.Output report
 -Update review
  PACDRP and PACDEI for anomalies

.Return codes
 . 0 : No error on files
 .08 : Error on user identification or parameter line
 .12 : Input/output error on a file
```

13.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*  D S M S     : DSMS DATABASE UPDATE                 *';
COMM '*                DSUSDUPT                             *';
COMM '*                                                     *';
COMM '*******************************************************';
MVL  USER='$USER',
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
JUMP CR&1;
CRJCL:
CR   IF=*DUPT,
     OF=(TMBDUPT,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=250,RECFORM=FB);
JUMP CREND;
CRDEXP:CRDEXT:CRDXBJ:
CR   IF=($NMLI.$LIBSU,&RFLI,SUBFILE=MBDUPD_&1&USER),
     OF=(TMBDUPT,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=250,RECFORM=FB),
     COMFILE=*SELREC,START=2;
CREND:
COMM '*** PDSBAS ***';
STEP PDSBAS,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ 30;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU;
     ASG PACDRS,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP END,SW30,EQ,1;
COMM '*** PDSUP0 ***';
STEP PDSUP0,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ 100;
     ASG DS80IA,$NMTU.$ROOT$FILEDA,&RFTU;
     ASG DS80DJ,$NMDJ.$ROOT$FILEDJ,&RFDJ;
     ASG DS80IX,$NMTU.$ROOT$FILEDX,&RFTU;
     ASG PB80DC,$NMTU.$ROOT$FILEDC,&RFTU;
     ASG DS80IE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF DS80IE,READLOCK=STAT;
     ASG PACDIM,TMBDUPT,TEMPRY,&RFTM;
     ASG PACDRP,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP END;
ERR:
SEND ' DSUSDUPT - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

14. FILE INITIALIZATION (DINI)

14.1. 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.

14.2. 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.

14.3. DESCRIPTION OF STEPS

DINI: DESCRIPTION OF STEPS

This procedure includes the following steps:

```
DESCRIPTION OF STEPS

This procedure includes the following steps:

      . Recognition of input              : CREATE

      . Initial backup of the database    : PDSINI

      . Shift of BB generations           : SHIFT


RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file
defined under IOF. It is found at the beginning of all
procedures which include user input.

. Input file                              : $INPUT

. Output file (TEMPRY)                    : TMBDINI
   (recsize 80, Cisize 2048)


INITIAL DATABASE BACKUP:  PDSINI

.Input transaction file:
 -Initialization transaction
  PACDMB  :    TMBDINI temporary file

.Permanent input file:
 -Error messages
  PACDDE  :    Extenal name $NMTU.$ROOT$ROOTDE

.Output file:
 -Sequential image of files
  PACDBB  :    External name $NMBU.$ROOT$FILEBB/G+1

.Output file:
 -Backup report
  PACDRU
```

14.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*  D S M S        : INITIALIZATION OF DSMS DATABASE  *';
COMM '*                                                     *';
COMM '*                    DSEXDINI                         *';
COMM '*                   ========                          *';
COMM '*                                                     *';
COMM '*******************************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DINI,
     OF=(TMBDINI,&RFTM,TEMPRY,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
STEP PDSINI,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  40;
     ASG PACDMB,TMBDINI,TEMPRY,&RFTM;
     ASG PACDBB,$NMBU.$ROOT$FILEBB/G+1,&RFBU;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF PACDDE,READLOCK=STAT;
     ASG PACDRU,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT $NMBU.$ROOT$FILEBB;
JUMP END;
ERR:
SEND ' DSEXDINI - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

15. JOURNAL EXTRACTION FOR UPDATE (DXBJ)

15.1. 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.

15.2. 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. .

15.3. DESCRIPTION OF STEPS

DXBJ: DESCRIPTION OF STEPS

This procedure includes te following steps:

. Recognition of input : CREATE

. Journal extraction : PDS700

. Storing of the result in source library

RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file defined under IOF. It is found at the beginning of all procedures which include user input.

. Input file : $INPUT

. Output file (TEMPRY) : TMBDXBJ
(recsize 80, Cisize 2048)

TRANSACTION EXTRACTION AND FORMATTING: PDS700

```
.Permanent input files:
 -Data file
  PACDDA  :  External name $NMTU.$ROOT$FILEDA
 -Error message file
  PACDDE  :  External name $NMTU.$ROOT$ROOTDE
 -Archived DSMS journal
  PACDBJ  :  External name $NMBU.$ROOT$FILEBJ

.Input transaction file :
 -User transactions
  PACDMB  :  TMBDXBJ temporary file

.Output file:
 -Update transaction file for DUPT
  PACDIM  :  TPACDIM temporary file (recsize 250)

.Output reports:
 -Extraction review
  PACDRK
 -Transaction printout
  PACDSK

.Return codes:
.0 : No error
.8 : Error on user identification or parameter line.
     Missing of environment definition
.12: File access error
     Missing of technical environment
```

15.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*  D S M S     : EXTRACTION FROM DSMS JOURNAL          *';
COMM '*                DSUSDXBJ                             *';
COMM '*                                                     *';
COMM '*                                                     *';
COMM '*  OUTPUT :                                           *';
COMM '*    OUTPUT TRANSACTIONS ARE STORED IN THE LIBRARY :  *';
COMM '*    $NMLI.$LIBSU                                     *';
COMM '*******************************************************';
MVL  USER='$USER',SIZEWK=1,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DXBJ,
     OF=(TMBDXBJ,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
STEP PDS700,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  100;
     ASG PACDMB,TMBDXBJ,TEMPRY,&RFTM;
     ASG PACDIM,TPACDIM,TEMPRY,&RFTM,END=PASS;
     DEF PACDIM,CISZ=4096,RECSZ=250,RECFORM=FB;
     ASG DS80IA,$NMTU.$ROOT$FILEDA,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF DS80IA,READLOCK=STAT;
     ASG DS80IE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACCESS=READ,SHARE=MONITOR;
     DEF DS80IE,READLOCK=STAT;
     ASG PACDBJ,$NMBU.$ROOT$FILEBJ,&RFBU;
     ASG PACDRK,SYS.OUT;
     ASG PACDSK,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP END,SW30,EQ,1;
LMN  SL INFILE=(TPACDIM,TEMPRY,&RFTM),
     LIB=($NMLI.$LIBSU,&RFLI),
     COM='MV INFILE:MBDUPD_DXBJ'&USER',INFORM=SARF,
          TYPE=DAT,NUMBER=(1,1),REPLACE;';
JUMP ERR,SW20,EQ,1;
JUMP END;
ERR:
SEND ' DSUSDXBJ - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

16. CODE AND KEYWORD UPDATE (DREN)

16.1. 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.

16.2. 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 '*' line or on a command line
10	Absence or invalid value for backup 'top'
11	Absence or invalid value for archive 'top'
12	Input/Output error or inconsistent DSMS base
	Invalid absence of backup/archive 'tops'

16.3. 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:

```
SYMBOLICS IN USE

-------------------------------------------------------------
! SYMBOLICS! MEANING                                <EXAMPLE>!
!------------------------------------------------------------!
! &SIZEWK  ! size of intermediary files              01    !
-------------------------------------------------------------
```

```
DESCRIPTION OF STEPS

This procedure includes the following steps:

      .  Recognition of input             : CREATE

      .  Modifications                    : PDSMS

      .  Shift of backups                 : SHIFT
```

```
RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file defined
under IOF. It is found at the beginning of all procedures
which include user input.

. Input file                              : $INPUT

. Output file (TEMPRY)                    : TMBDREN
   (recsize 80, Cisize 2048)

UPDATES:  PDSMS

.Permanent input files:
 -Data file
  DS80IA  :   External name $NMTU.$ROOT$FILEDA
 -Error messages
  DS80IE  :   External name $NMTU.$ROOT$ROOTDE
 -Cross-references
```

```
      DS80IX  :   External name $NMTU.$ROOT$FILEDX
     -DSMS backup
      PACDBB  :   External name $NMBU.$ROOT$FILEBB
     -DSMS archiving
      PACDBJ  :   External name $NMBU.$ROOT$FILEBJ

    .Input file:
     -User queries
      PACDMB  :   TMBDREN temporary file

    .Work files:
     -Update requests
      PACDW0
     -Partial backup (sorted)
      PACDW1
     -Partial backup (not sorted)
      PACDW2

    .Output files:
     -Modified backup
      PACDB3  :   External name $NMBU.$ROOT$FILEBB/g+1
     -Modified archive
      PACDJB  :   External name $NMBU.$ROOT$FILEBJ/g+1

    .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:
```

16.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*                    DSEXDREN                         *';
COMM '*                    ========                         *';
COMM '*  D S M S           :  CHANGE CODES AND KEYWORDS     *';
COMM '*                                                     *';
COMM '*                                                     *';
COMM '*******************************************************';
MVL  SIZEWK=2,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM',
     PGE='E',PGLNG=&PG$LANG;
CR   IF=*DREN,
     OF=(TMBDREN,&RFTM,TEMPRY,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
STEP PDSMS&PGLNG,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  100;
     ASG PACDMB,TMBDREN,TEMPRY,&RFTM,END=PASS;
     ASG DS80IA,$NMTU.$ROOT$FILEDA,&RFTU,
         ACC=READ,SHARE=MONITOR;
     DEF DS80IA,READLOCK=STAT;
     ASG DS80IE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACC=READ,SHARE=MONITOR;
     DEF DS80IE,READLOCK=STAT;
     ASG DS80IX,$NMTU.$ROOT$FILEDX,&RFTU,
         ACC=READ,SHARE=MONITOR;
     DEF DS80IX,READLOCK=STAT;
     ASG PACDBB,$NMBU.$ROOT$FILEBB,&RFBU;
     ASG PACDBJ,$NMBU.$ROOT$FILEBJ,&RFBU;
     ASG PACDW0,TPACDW0,TEMPRY,&RFTM;
     ALC PACDW0,SZ=&SIZEWK,INCRSZ=1,UNIT=CYL;
     DEF PACDW0,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW1,TPACDW1,TEMPRY,&RFTM;
     ALC PACDW1,SZ=&SIZEWK,INCRSZ=1,UNIT=CYL;
     DEF PACDW1,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDW2,TPACDW2,TEMPRY,&RFTM;
     ALC PACDW2,SZ=&SIZEWK,INCRSZ=1,UNIT=CYL;
     DEF PACDW2,CISIZE=$CISEQ,NBBUF=1;
     ASG PACDB3,$NMBU.$ROOT$FILEBB/G+1,&RFBU;
     ASG PACDJB,$NMBU.$ROOT$FILEBJ/G+1,&RFBU;
     ASG PACDIA,SYS.OUT;
     ASG PACDIK,SYS.OUT;
     ASG PACDJK,SYS.OUT;
     ASG PACDIS,SYS.OUT;
     ASG PACDKK,SYS.OUT;
     ASG PACDLK,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP ERR,SW30,EQ,1;
COMM '*** CTL FOR SHIFT ***';
```

```
    FILLIST INFILE=$NMBU.$ROOT$FILEBB/G+1 SPACE PRTFILE=DUMMY;
    JUMP SHBBNO SEV GE 3;
    JUMP SHBBNO SW21 EQ 1;
    JUMP SHBBNO SW23 EQ 1;
    SHIFT $NMBU.$ROOT$FILEBB;
    SHBBNO:
    FILLIST INFILE=$NMBU.$ROOT$FILEBJ/G+1 SPACE PRTFILE=DUMMY;
    JUMP SHBJNO SEV GE 3;
    JUMP SHBJNO SW22 EQ 1;
    JUMP SHBJNO SW23 EQ 1;
    SHIFT $NMBU.$ROOT$FILEBJ;
    SHBJNO:
    JUMP END;
    ERR:
    SEND ' DSEXDREN - ABNORMAL END OF RUN (I/O ERROR) ';
    LET  SEV 3;
    END:
```

CODE AND KEYWORD UPDATE (DREN) 16
EXECUTION JCL 4

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

17.1. 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.

17.2. 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.

17.3. 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:

```
SYMBOLICS IN USE

+----------+-----------------------------------------------+
! SYMBOLICS! MEANING                                <EXAMPLE>!
+----------+-----------------------------------------------+
! &USER    ! USER CODE                                <$USER>!
+----------+-----------------------------------------------+


DESCRIPTION OF STEPS

This procedure includes the following steps:

      .Recognition of input             : CREATE

      .Pre-processor                    : DAFD10

      .Storing the result in source library


RECOGNITION OF INPUT

This utility creates a temporary file from a user generated
program which contains SQL orders.

.Input file                               : &1
                                         (generated program)
.Output file                              : TMBDPDF

GENERATED PROGRAM'S PRE-PROCESSOR:  DAFD10

.Permanent input files:
 -Data file
  DS80IA  :   External name $NMTU.$ROOT$FILEDA
 -Error message file
  DS80IE  :   External name $NMTU.$ROOT$ROOTDE

.Input file:
 -Generated programs
  DAF80  :    TMBDPDF temporary file

.Output files:
 -Generated programs to be compiled
  COB80  :    TMBDPDF temporary file
```

```
.Output reports:
 -Execution report
  DAFREP
```

Note: if the generated stream contains the compiling control lines, if DPDF is executed after GPRT, this DSN (TMBPDF) can be replaced by the sending of the stream in the machine's Internal Reader.

17.4. EXECUTION JCL

```
COMM '*******************************************************';
COMM '*                   DSUSDPDF                          *';
COMM '*                   ========                          *';
COMM '*                                                     *';
COMM '*     PRE-PROCESSING OF A VA PAC DAF SOURCE           *';
COMM '*                                                     *';
COMM '*******************************************************';
MVL  DAF,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
MVT:
CR   IF=*&1,
     OF=(TMBDPDF,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80);
STEP DAFD10,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  130;
     ASG DS80IE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACC=READ,SHARE=MONITOR;
     DEF DS80IE,NBBUF=2,READLOCK=STAT;
     ASG DS80IA,$NMTU.$ROOT$FILEDA,&RFTU,
         ACC=READ,SHARE=MONITOR;
     DEF DS80IA,READLOCK=STAT;
     ASG DAF80,TMBDPDF,TEMPRY,&RFTM,END=PASS;
     DEF DAF80,NBBUF=1;
     ASG COB80,&1,TEMPRY,&RFTM,END=PASS;
     DEF COB80,NBBUF=1;
     ASG DAFREP,SYS.OUT;
ESTP;
LMN  SL INFILE=(&1,TEMPRY,&RFTM,END=PASS),
     LIB=($NMLI.$LIBSU,&RFLI),
     COM='MV INFILE:'&1',INFORM=SARF,
          TYPE=DAT,NUMBER=(1,1),REPLACE;';
JUMP END,SW30,EQ,0;
JUMP ERR,SW20,EQ,1;
JUMP END;
ERR:
SEND ' DSUSDPDF - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

18. I N S T A L L A T I O N

18.1. INTRODUCTION

INTRODUCTION

The installation procedure is executed in three main phases:

. Preparation for installation,

. Installation,

. On-line and batch tests.

A special installation tape is provided by IBM. The whole installation process is described in this chapter.

Before executing the actual installation, the user must be familiar with the technical characteristics of the new DSMS release, described in this manual. This information is necessary to prepare the environment required for the installation procedure (disk space, definition of the TDS and its users on the catalog, etc.).

Once the environment is prepared, the installation may be performed. The installation procedure is described in the following subchapters.

PREPARATION

Definition of a JCL library, loading of this library using a backup included in the tape, definition of the JCL parameters.

. Backup of the installation tape,

. Allocation and unloading of a library containing the complete DSMS installation and operation JCL.

. Adaptation of the JCL to the site's specific needs.

INSTALLATION

. Pre-allocation of the system files,

. Preparation of the TDS,

. File installation,

. TDS generation,

. Program link edit,

. Installation of a DSMS test database.

TESTING

. On-line testing,

. Batch procedure testing.

18.2. INSTALLATION TAPE

INSTALLATION TAPE

The installation tape (6250 BPI, standard labels) contains the following files:

RANK	LABEL	CONTENTS
1	SVF.JCL	Skeleton JCL for installation and operation in French.
2	SVE.JCL	Skeleton JCL for installation and operation in English.
3	SV.SL	TDS source, TPRs source.
4	SV.CUB	Batch compile-units.
5	SV.CUT	TPRs compile-units.
6	SV.DE0	Error message and documentation initial file.
7	SV.BB	DSMS database backup
8	SV.SOURCE	Sources of user sub-programs
9	SVF.ABOUT	Characteristics of the version in French
10	SVE.ABOUT	Characteristics of the version in English

18.3. JCL INSTALLATION

INSTALLATION OF THE COMPLETE JCL

The installation is executed in three steps:

1- Allocation of a JCL library ($NMLI.$LIBJCL) by BLIB (members=70,sz=1). The name of this library must conform to the values of the parameters constituting its name. The parameter values must be entered in the 'DSZZVALS' member during the JCL adaptation.

2- Loading of the library, using the first file on the tape (SV.JCL) by a LIBMAINT.

3- Adaptation of the JCL to the site specific needs. The adjustment is executed using IOF editor, modifying the 'DSZZVALS' member of the JCL library. (The default values of the parameters are replaced by their specific values on the site.) Subsequently, the DSZZEXEC procedure, which prepares the parameter substitution, must be started and the JCL must be executed. 1) EXEC DSZZEXEC VL=DSZZJCL BRIEF 2) SUBMIT DSZZJCL

NOTES :
Error messages such as 'SUBSTITUTION FAILED' are displayed during the DSZZEXEC procedure execution. These messages do not interfere with the procedure execution.

DSMS TDS is usually linked in the DSMS library of batch load-modules. For this reason, the value of $NMTD must be different from the values of DSMS standard load-modules. mo(See the list of Batch Load-Modules in the Chapter "DSMS COMPONENTS").

NOTES

Error messages such as 'SUBSTITUTION FAILED' are displayed during the DSZZEXEC procedure execution. These messages do not interfere with the procedure execution.

DSMS TDS is usually linked in the DSMS library of batch load-modules. For this reason, the value of $NMTD must be different from the values of DSMS standard load-modules. (See the list of Batch Load-Modules in the Chapter "DSMS COMPONENTS").

TABLE OF VA Pac JCL MODULES

These members are in the '$NMLI.$LIBJCL' library.

OPERATIONS PROCEDURES

Member	Contents	Nature
DSEXDARC	Journal archiving	JCL
DSEXDRST	Database restoration	JCL
DSEXDSAV	Database backup	JCL
DSEXPDSL	Technical member which transforms an IOF mb into 80-character UFAS	JCL
DSEXDREO	Database reorganization	JCL
DSEXDINI	Database initialization	JCL
DSEXJOBL	JOB submitter for GP	JCL
DSEXDREN	Change of entity codes	JCL
DSEXDLVB	Replacement of low value with blank	JCL

USER PROCEDURES

Member	Contents	Nature
DSUSDEXP	VA Pac journal extraction	JCL
DSUSDEXQ	VA Pac journal (< 2.0) extraction	JCL
DSUSDEXH	Entity extraction	JCL
DSUSDEXT	Table extraction for revamped DSMS	JCL
DSUSDPRT	Printing of queries	JCL
DSUSDUPT	DSMS database updating	JCL
DSUSDXBJ	Extraction from DSMS journal	JCL
DSUSDPDF	DAF pre-processor for generated pgms	JCL

INSTALLATION PROCEDURES

Member	Contents	Nature
DSINALxx	File allocation (*)	JCL
DSINALIV	Procedures' call library allocation	JCL
DSINALLI	Library allocation	JCL
DSINBLNK	Batch program link-edit	JCL
DSINBLSO	Batch program standard link-edit	JCL
DSINB1LN	'DPRT' program link-edit	JCL
DSINB2LN	'DEXT' program link-edit	JCL
DSINB3LN	'DUPT' program link-edit	JCL
DSINB4LN	'DEXH' program link-edit	JCL
DSINB5LN	'DXBJ' program link-edit	JCL
DSINB6LN	'DREN' program link-edit	JCL
DSINB7LN	DAF pre-processor link-edit	JCL
DSINDRS1	Restoration of the test database	JCL
DSININJB	PACDJB file initialization	JCL
DSINMAXI	Maximum index record	JCL
DSINMGEN	TDS generation procedure submission	JCL
DSINMPRE	TDS preparation procedure submission	JCL
DSIVDR80	8.0.1 retrieval	JCL
DSIVDR8Q	8.0.2 01/02 retrieval	JCL
DSIVDR8X	Retrieval of DSMS 8.0.2 compatible PB801	JCL
DSIVDR15	1.2 ou 1.5 retrieval	JCL
DSIVDR5J	Retrieval of 1.2 or 1.5 journal	JCL
DSINPRBS	Re-allocation of files	JCL
DSINPRDS	Allocation of files and libraries	JCL
DSINTLNK	TPRs link-edit in SMLIB	JCL
DSINTLSO	TPRs standard link-edit	JCL
DSINUNLD	Installation tape unloading	JCL
DSZZEDIT	JCL parameterization	JCL
DSZZEXEC	JCL parameterization	JCL
DSZZJCL	JCL parameterization	JCL
DSZZVALS	Parameter default values	DAT

(*) The last two characters (xx) represent the file name.
For example, for the error message file (DE) allocation,
the name of the procedure is DSINALDE.

PROCEDURE INPUT

These members contain the user input; they are contained in the '$NMLI.$LIBINV' library.
They call, via INVOKE, the procedures of the '$NMLI.$LIBJCL' library.

Member	Contents	Nature
DSIVDARC	Journal archiving	JCL
DSIVDEXP	Journal extraction	JCL
DSIVDEXT	Entity extraction	JCL
DSIVDEXH	Table extraction for revamped DSMS	JCL
DSIVDINI	Database initialization	JCL
DSIVDLVB	Replacement of low values with blanks	JCL
DSIVDPRT	Printing of queries	JCL
DSIVDREO	Database reorganization	JCL
DSIVDRST	Database restoration	JCL
DSIVDRS1	Test DSMS restoration	JCL
DSIVDR80	8.0.1 retrieval	JCL
DSIVDR8Q	8.0.2 01/02 retrieval	JCL
DSIVDR8X	Retrieval of 8.0.2 compatible PB801	JCL
DSIVDR15	1.2 or 1.5 retrieval	JCL
DSIVDR5J	Retrieval of 1.2 or 1.5 journal	JCL
DSIVDSAV	Database backup	JCL
DSIVDUPT	Batch update	JCL
DSIVDXBJ	Extraction from journal	JCL
DSIVDREN	Loading of entity codes	JCL
DSIVDPDF	DAF pre-processor of generated programs	JCL
DSUSDEXP	Extraction of VA Pac journal	JCL
DSUSDEXQ	Extr. of VA Pac journal (< 2.0)	JCL
DSUSDEXT	Entities extraction	JCL
DSUSDEXH	Tables extraction for DSMS revamping	JCL
DSUSDPRT	Printing of queries	JCL
DSUSDUPT	Update of DSMS database	JCL
DSUSDXBJ	Extraction of DSMS journal	JCL

```
" ***********************************************
" *                                             *
" * DSZZEDIT : THIS EDIT PROCEDURE IS CALLED BY *
" * THE JCL INTERPRETOR (DSZZEXEC).             *
" * ITS PURPOSE IS TO PREPARE THE USER SUBFILE  *
" * DSZZVALS FOR JCL INTERPRETATION.            *
" *                                             *
" ***********************************************
"  WEAKEN RETURN CODES
YW
"  LOADING DSZZVALS
RDSZZVALS
"  REMOVE ALL BLANKS IN THE LINES
GS/ //
"  DELETE LINES WHICH DO NOT BEGIN WITH $
VD/^°C$/
" INSERT STRING "GS=°C" AT THE BEGINNING OF EACH LINE
GS/^/GS=°C°°CC/
" INSERT STRING "°C" IN FRONT OF EACH "&" CHARACTER
GS/°C&/°C°°CC°C&/
" INSERT STRING "="     AT THE END OF EACH LINE
GS/$/=/
```

```
COMM 'DSMS 2.5';
COMM ******************************************;
COMM *                                        *;
COMM * DSZZEXEC : JCL INTERPRETATION EXEC PROC. *;
COMM * THIS PROCEDURE PREPARES THE USER SUBFILE *;
COMM * DSZZVALS WITH THE PROCEDURE DSZZEDIT.    *;
COMM * THEN IT REPLACES D S M S PARAMETERS BY   *;
COMM * USER VALUES IN TARGET SUBFILES SPECIFIED *;
COMM * IN PARAMETER 1, ACCORDING TO THE NAMING  *;
COMM * CONVENTIONS OF LIBRARY SUBFILES.         *;
COMM * EX : EXEC DSZZEXEC VL=DSZZJCL BRIEF      *;
COMM *                                        *;
COMM *                                        *;
COMM ******************************************;
ED;
YB
B1
RDSZZEDIT
B0
°E1
Z(JCL)DSZZTEMP
Q
STATUS RESET;
ED  LIB:&1;
YB
B1
RDSZZTEMP
B0
R &0
°E1
Z &0
Q
STATUS RESET;
DELETE DSZZTEMP;
```

```
COMM 'DSMS 2.5';
COMM '************************************************';
COMM '*                                              *';
COMM '*  DSZZJCL : THIS PROCEDURE MUST BE EXECUTED   *';
COMM '*  FOR THE INTERPRETATION OF D S M S JCL.      *';
COMM '*  BEFORE EXECUTION, DSZZJCL ITSELF HAS TO BE  *';
COMM '*  INTERPRETED BY THE EXEC PROCEDURE DSZZEXEC  *';
COMM '*  EX : EXEC DSZZEXEC VL=DSZZJCL BRIEF;        *';
COMM '*                                              *';
COMM '************************************************';
MVL  ^DSZZ*,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
LMN  SL LIB=($NMLI.$LIBJCL,&RFLI),
        COMFILE=*DS80A,PRTFILE=DUMMY;
$IN  DS80A PRINT JVL;
EXEC DSZZEXEC VL=&1;
$EIN DS80A;
LMN  SL LIB=($NMLI.$LIBJCL,&RFLI),
        COMFILE=*DS80B;
$IN  DS80B PRINT JVL;
PR   LIB:DSZZVALS;
PR   LIB:&1;
$EIN DS80B;
```

```
COMM 'DSMS 2.5';
*****************************************
*        INSTALLATION PARAMETERS        *
*                                       *
*  REPLACE, IF NEEDED, THE DEFAULT      *
*  VALUE OF EACH PARAMETER.             *
*  EACH PARAMETER LINE IS FORMATTED AS  *
*  FOLLOWS:                             *
*    $NNNNN = VALUE                     *
*                                       *
*  THIS FILE IS PROCESSED BY AN EDITOR  *
*  PROGRAM WHICH CHANGES PARAMETER LINES*
*  INTO SUBSTITUTION COMMANDS.          *
*                                       *
*  SUBSEQUENTLY,                        *
*                                       *
*  - ALL LINES WHOSE FIRST NON-BLANK    *
*    CHARACTER IS NOT A DOLLAR SIGN     *
*    ARE CONSIDERED AS COMMENTS.        *
*                                       *
*  - THE EQUAL SIGN (DELIMITER) CANNOT  *
*    BE USED IN A PARAMETER VALUE.      *
*                                       *
*****************************************

******** DEFAULT  D S M S   USER        *
$USER   =  IBM
******** NAME  OF  INSTALLATION  CATALOG  *
$CTNM   =  PV
******** D S M S     TDS NAME           *
$NMTD   =  TDS
******** INSTALLATION TAPE    *************
$TAPE   =  XXXXXX
$DVTP   =  CT/M5
******** LANGUAGE E(NGLISH)/F(RENCH) ******
$LANG   =  E
************ SUFFIX OF LIBRARIES **********
******** BATCH CU
$LIBCUB =  CUBLIB
******** TP CU
$LIBCUT =  CUTLIB
******** PERMANENT CU
$LIBCUP =  CUPLIB
******** JCL
$LIBJCL =  JCLLIB
******** INVOKE JCLS SUBMISSION
$LIBINV =  INVLIB
******** PRINT OF JCLS CALLED BY INVOKE **
** BLANK NOT TO LIST OR &LIST TO LIST
$LIST   =  &LIST
******** LM
$LIBLM  =  LMLIB
******** SM  (SEE NOTE)
$LIBSM  =  SMLIB
******** TDS SL (SEE NOTE)
$LIBSL  =  SLLIB
******** USERS SL
$LIBSU  =  SULIB
******** DATABASE FILES' REFERNCES*********
******** ROOT OF DSMS FILES             *
$ROOT   =  DS
******** IDENTIFIER OF DSMS FILES       *
$FILE   =  250
******** CISIZE OF BATCH FILES          *
```

```
** DEFAULT VALUE FOR AN MS/D500 DISK TYPE **
$CISEQ  =  14336
**************** REFERENCES **************
*NMXX   =  FILE PREFIX
*DVXX   =  DISK TYPE
*MDXX   =  DISK NAME
*CTXX   =  CATALOGING (Y OR N)
******** TDS USER FILES
$NMTU   =  DSTU
$DVTU   =  MS/FSA
$MDTU   =  DISC01
$CTTU   =  Y
******** TDS FILES
$DVTD   =  MS/FSA
$MDTD   =  DISC02
$CTTD   =  Y
******** BATCH USER FILES
$NMBU   =  DSBU
$DVBU   =  MS/FSA
$MDBU   =  DISC03
$CTBU   =  Y
******** BATCH SYSTEM FILES
$NMBS   =  DSBS
$DVBS   =  MS/FSA
$MDBS   =  DISC04
$CTBS   =  Y
******** DSMS JOURNAL FILE
$NMDJ   =  DSDJ
$DVDJ   =  MS/FSA
$MDDJ   =  DISC05
$CTDJ   =  Y
******** TEMPORARY FILE
$DVTM   =  MS/FSA
$MDTM   =  DISC06
******** LIBRARIES
$NMLI   =  DSLI
$DVLI   =  MS/FSA
$MDLI   =  DISC07
$CTLI   =  Y
************ SM ENVIRONMENT   (SEE  NOTE) *
*** IF INSERTION IN AN EXISTING TDS       *
*** TAKE THE TPRS WHICH ARE NOT ALLOCATED *
******** DSMS  SM NAME                     *
$TPR0   =  TPR
******** DSMS  SM  NAME                    *
$TPR1   =  TPR1
******************************************
* IMPORTANT NOTE :
******************************************
* TO BE CONSISTENT WITH THE PROCEDURES    *
* SHIPPED BY THE VENDOR:                  *
* TP7PREP, TP7GEN, ETC..                  *
* IBM PROPOSES A CHOICE OF VALUES WHICH   *
* RESPECT THE COHERENCE OF THESE          *
* PROCEDURES' PARAMETERS.                 *
* SO IT IS HIGHLY RECOMMENDED TO USE      *
* THESE VALUES.                           *
******************************************
```

18.4. INSTALLATION PROCESS

INSTALLATION PROCESS

Once the JCL is obtained, DSMS System installation is executed in 7 main steps:

1. System file allocation
2. DSMS management TDS preparation
3. Unloading of files and programs
4. DSMS management TDS generation
5. Link-edit of generation-printing and batch program monitors, and link-edit of the TPRs in the SMLIB prepared for DSMS
6. Creation of a test DSMS database
7. TDS submission.

1. SYSTEM FILE ALLOCATION (DSINPRDS)

The system file allocation is executed by the 'DSINPRDS' member, which is included in the $NMLI.$LIBJCL library.

Although the allocation of some files is also performed by procedures mentioned later, this step is useful to check that the disk space needed is available.

The 'DSINPRDS' member is a sequence of PREALLOC/LIBALLOC, which may be logically split as follows:

```
.Allocation of permanent UFAS files:
      $NMBS.$ROOT$ROOTDE0

.Allocation of test database UFAS files:
      $NMBU.$ROOT$FILEBB
      $NMBU.$ROOT$FILEBJ
      $NMBU.$ROOT$FILEBQ
      $NMTU.$ROOT$FILEDA
      $NMTU.$ROOT$FILEDC
      $NMDJ.$ROOT$FILEDJ
      $NMTU.$ROOT$FILEDX
      $NMTU.$ROOT$ROOTDE

.Allocation of Program libraries:
       $NMLI.$LIBLM
       $NMLI.$LIBCUB
       $NMLI.$LIBCUT
       $NMLI.$LIBCUP
       $NMLI.$LIBSU
       $NMLI.$LIBINV
       $NMLI.SOURCE
       $NMLI.ABOUT
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '**********************************';
COMM '*    FILE AND LIBRARY             *';
COMM '*    ALLOCATION FOR               *';
COMM '*      D S M S                    *';
COMM '**********************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
OVL  HOLD;
COMM '*** BATCH SYSTEM ***';
IV   DSINALD0,($NMLI.$LIBJCL,&RFLI);
COMM '*** TDS    ***';
IV   DSINALDA,($NMLI.$LIBJCL,&RFLI);
IV   DSINALDC,($NMLI.$LIBJCL,&RFLI);
IV   DSINALDE,($NMLI.$LIBJCL,&RFLI);
IV   DSINALDJ,($NMLI.$LIBJCL,&RFLI);
IV   DSINALDX,($NMLI.$LIBJCL,&RFLI);
COMM '*** BATCH  ***';
IV   DSINALBB,($NMLI.$LIBJCL,&RFLI);
IV   DSINALBJ,($NMLI.$LIBJCL,&RFLI);
IV   DSINALBQ,($NMLI.$LIBJCL,&RFLI);
COMM '***  LIBRARIES  ***';
IV DSINALLI,($NMLI.$LIBJCL,&RFLI) VL=($NMLI.$LIBLM,LM,5,1,70);
IV DSINALLI,($NMLI.$LIBJCL,&RFLI) VL=($NMLI.$LIBCUB,CU,3,1,70);
IV DSINALLI,($NMLI.$LIBJCL,&RFLI) VL=($NMLI.$LIBCUT,CU,6,1,100);
IV DSINALLI,($NMLI.$LIBJCL,&RFLI) VL=($NMLI.$LIBCUP,CU,1,1,10);
IV DSINALLI,($NMLI.$LIBJCL,&RFLI) VL=($NMLI.$LIBSU,SL,1,1,100);
IV DSINALIV,($NMLI.$LIBJCL,&RFLI) VL=($NMLI.$LIBINV,SL,1,1,50);
IV DSINALLI,($NMLI.$LIBJCL,&RFLI) VL=($NMLI.SOURCE,SL,1,1,50);
IV DSINALLI,($NMLI.$LIBJCL,&RFLI) VL=($NMLI.ABOUT,SL,1,1,50);
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINALLI                    *';
COMM '*             ========                    *';
COMM '*  DSMS LIBRARY ALLOCATION                *';
COMM '*******************************************';
MVL    CTLI='FILESTAT=CAT,UNCATNOW',
       CLIY='FILESTAT=CAT,DVC=$DVLI,MD=$MDLI',
       DV='DVC=$DVLI',MD='MD=$MDLI',
       COMP&2=COMPACT;
OVL    HOLD;
JUMP   CAT$CTLI;
CATY:
DALC   &1,&CTLI;
JUMP   CONTINUE;
CAT    &1,TYPE=FILE,SHARE=UNSPEC;
LALC   &2,(&1,&CLIY,SZ=(&3,&4)),MEMBERS=&5,&COMPSL;
JUMP   END;
CATN:
DALC   &1,&DV,&MD;
JUMP   CONTINUE;
LALC   &2,(&1,&DV,&MD,SZ=(&3,&4)),MEMBERS=&5,&COMPSL;
END:
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINALIV                    *';
COMM '*             ========                    *';
COMM '*                                         *';
COMM '* ALLOCATION OF DSMS PROC INVOKE LIBRARY   *';
COMM '*                                         *';
COMM '*******************************************';
MVL    CTLI='FILESTAT=CAT,UNCATNOW',
       CLIY='FILESTAT=CAT,DVC=$DVLI,MD=$MDLI',
       CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
       RFLI=&CTLI$CTLI,
       COMP&2=COMPACT;
OVL    HOLD;
JUMP   CAT$CTLI;
CATY:
CAT    &1,TYPE=FILE,SHARE=UNSPEC;
JUMP   CONTINUE;
LALC   &2,(&1,&CLIY,SZ=(&3,&4)),MEMBERS=&5,&COMPSL;
JUMP   CONTINUE;
JUMP   CATEND;
CATN:
LALC   &2,(&1,&CTLIN,SZ=(&3,&4)),MEMBERS=&5,&COMPSL;
JUMP   CONTINUE;
CATEND:
LIB    SL,INLIB1=($NMLI.$LIBJCL,&RFLI);
LMN    SL,LIB=(&1,&RFLI),
       COM='MV IL1:DSIV*,REPLACE;';
END:
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINALD0                    *';
COMM '*             ========                    *';
COMM '*                                         *';
COMM '*  DEO FILE ALLOCATION                    *';
COMM '*******************************************';
MVL  CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     CTBSY='FILESTAT=CAT,UNCATNOW',RFBS=&CTBS$CTBS,
     CBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     CBSY='FILESTAT=CAT',RBS=&CBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     DV1='DVC=$DVBS',MD1='MD=$MDBS',
     CATFN='FILESTAT=UNCAT',
     CATFY='FILESTAT=CAT';
DALC $NMBS.$ROOT$ROOTDE0,&RFBS;
JUMP CONTINUE;
JUMP D0$CTBS;
D0Y:
CAT  $NMBS.$ROOT$ROOTDE0,TYPE=FILE,SHARE=UNSPEC;
D0N:
PALC $NMBS.$ROOT$ROOTDE0,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=8),
     UFAS=(SEQ=(CISZ=2048,RECSZ=90)),
     &CATF$CTBS;
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*            DSINALBB                     *';
COMM '*            ========                     *';
COMM '*   BB FILE ALLOCATION                    *';
COMM '*******************************************';
MVL  CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     CTBUY='FILESTAT=CAT,UNCATNOW',RFBU=&CTBU$CTBU,
     CBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     CBUY='FILESTAT=CAT',RBU=&CBU$CTBU,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     DV1='DVC=$DVBU',MD1='MD=$MDBU',
     CATFN='FILESTAT=UNCAT',
     CATFY='FILESTAT=CAT';
DALC $NMBU.$ROOT$FILEBB,&RFBU;
JUMP CONTINUE;
DALC $NMBU.$ROOT$FILEBB/G-1,&RFBU;
JUMP CONTINUE;
UNCAT $NMBU.$ROOT$FILEBB,TYPE=FILE;
JUMP CONTINUE;
JUMP BB$CTBU;
BBY:
CAT  $NMBU.$ROOT$FILEBB,TYPE=FILE,NBGEN=2,SHARE=UNSPEC;
BBN:
PALC $NMBU.$ROOT$FILEBB/G+1,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=1),INCRSZ=1,
     UFAS=(SEQ=(CISZ=2048,RECSZ=350,RECFORM=V)),
     &CATF$CTBU;
PALC $NMBU.$ROOT$FILEBB/G+1,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=1),INCRSZ=1,
     UFAS=(SEQ=(CISZ=2048,RECSZ=350,RECFORM=V)),
     &CATF$CTBU;
```

```
COMM 'DSMS 2.5';
COMM '*****************************************';
COMM '*            DSINALBJ                   *';
COMM '*            ========                   *';
COMM '*   BJ FILE ALLOCATION                  *';
COMM '*****************************************';
MVL  CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     CTBUY='FILESTAT=CAT,UNCATNOW',RFBU=&CTBU$CTBU,
     CBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     CBUY='FILESTAT=CAT',RBU=&CBU$CTBU,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     DV1='DVC=$DVBU',MD1='MD=$MDBU',
     CATFN='FILESTAT=UNCAT',
     CATFY='FILESTAT=CAT';
DALC $NMBU.$ROOT$FILEBJ,&RFBU;
JUMP CONTINUE;
DALC $NMBU.$ROOT$FILEBJ/G-1,&RFBU;
JUMP CONTINUE;
UNCAT $NMBU.$ROOT$FILEBJ,TYPE=FILE;
JUMP CONTINUE;
JUMP BJ$CTBU;
BJY:
CAT  $NMBU.$ROOT$FILEBJ,TYPE=FILE,NBGEN=2,SHARE=UNSPEC;
BJN:
PALC $NMBU.$ROOT$FILEBJ/G+1,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=1),INCRSZ=1,
     UFAS=(SEQ=(CISZ=2048,RECSZ=180,RECFORM=F)),
     &CATF$CTBU;
PALC $NMBU.$ROOT$FILEBJ/G+1,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=1),INCRSZ=1,
     UFAS=(SEQ=(CISZ=2048,RECSZ=180,RECFORM=F)),
     &CATF$CTBU;
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINALBQ                    *';
COMM '*             ========                    *';
COMM '*   BQ FILE ALLOCATION                    *';
COMM '*******************************************';
MVL  CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     CTBUY='FILESTAT=CAT,UNCATNOW',RFBU=&CTBU$CTBU,
     CBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     CBUY='FILESTAT=CAT',RBU=&CBU$CTBU,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     DV1='DVC=$DVBU',MD1='MD=$MDBU',
     CATFN='FILESTAT=UNCAT',
     CATFY='FILESTAT=CAT';
DALC $NMBU.$ROOT$FILEBQ,&RFBU;
JUMP CONTINUE;
JUMP BQ$CTBU;
BQY:
CAT  $NMBU.$ROOT$FILEBQ,TYPE=FILE,SHARE=UNSPEC;
BQN:
PALC $NMBU.$ROOT$FILEBQ,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=1),INCRSZ=1,
     UFAS=(SEQ=(CISZ=2048,RECSZ=180,RECFORM=F)),
     &CATF$CTBU;
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINALDA                    *';
COMM '*             ========                    *';
COMM '*   DA FILE ALLOCATION                    *';
COMM '*******************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     CTTUY='FILESTAT=CAT,UNCATNOW',RFTU=&CTTU$CTTU,
     CTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     CTUY='FILESTAT=CAT',RTU=&CTU$CTTU,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     DV1='DVC=$DVTU',MD1='MD=$MDTU',
     CATFN='FILESTAT=UNCAT',
     CATFY='FILESTAT=CAT';
DALC $NMTU.$ROOT$FILEDA,&RFTU;
JUMP CONTINUE;
JUMP DA$CTTU;
DAY:
CAT  $NMTU.$ROOT$FILEDA,TYPE=FILE,SHARE=UNSPEC;
DAN:
PALC $NMTU.$ROOT$FILEDA,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=1),INCRSZ=1,
     UFAS=(INDEXED=(CISZ=2048,RECSZ=340,RECFORM=V,
           KEYLOC=3,KEYSZ=40)),
     &CATF$CTTU;
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINALDC                    *';
COMM '*             ========                    *';
COMM '*   DC FILE ALLOCATION                    *';
COMM '*******************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     CTTUY='FILESTAT=CAT,UNCATNOW',RFTU=&CTTU$CTTU,
     CTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     CTUY='FILESTAT=CAT',RTU=&CTU$CTTU,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     DV1='DVC=$DVTU',MD1='MD=$MDTU',
     CATFN='FILESTAT=UNCAT',
     CATFY='FILESTAT=CAT';
DALC $NMTU.$ROOT$FILEDC,&RFTU;
JUMP CONTINUE;
JUMP DC$CTTU;
DCY:
CAT  $NMTU.$ROOT$FILEDC,TYPE=FILE,SHARE=UNSPEC;
DCN:
PALC $NMTU.$ROOT$FILEDC,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=1),INCRSZ=1,
     UFAS=(INDEXED=(CISZ=4096,RECSZ=168,RECFORM=V,
           KEYLOC=3,KEYSZ=31)),
     &CATF$CTTU;
CR   IF=($NMLI.$LIBJCL,&RFLI,SUBFILE=DSINMAXI),
     OF=($NMTU.$ROOT$FILEDC,&RFTU),
     COMFILE=($NMLI.$LIBJCL,&RFLI,SUBFILE=DSEXPDSL),START=2;
END:
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINALDX                    *';
COMM '*             ========                    *';
COMM '*   DX FILE ALLOCATION                    *';
COMM '*******************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     CTTUY='FILESTAT=CAT,UNCATNOW',RFTU=&CTTU$CTTU,
     CTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     CTUY='FILESTAT=CAT',RTU=&CTU$CTTU,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     DV1='DVC=$DVTU',MD1='MD=$MDTU',
     CATFN='FILESTAT=UNCAT',
     CATFY='FILESTAT=CAT';
DALC $NMTU.$ROOT$FILEDX,&RFTU;
JUMP CONTINUE;
JUMP DX$CTTU;
DXY:
CAT  $NMTU.$ROOT$FILEDX,TYPE=FILE,SHARE=UNSPEC;
DXN:
PALC $NMTU.$ROOT$FILEDX,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=1),INCRSZ=1,
     UFAS=(INDEXED=(CISZ=2048,RECSZ=80,RECFORM=F,
           KEYLOC=1,KEYSZ=50)),
     &CATF$CTTU;
```

```
    COMM 'DSMS 2.5';
    MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
         CTTUY='FILESTAT=CAT,UNCATNOW',RFTU=&CTTU$CTTU,
         CTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
         CTUY='FILESTAT=CAT',RTU=&CTU$CTTU,
         CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
         RFLI=&CTLI$CTLI,
         DV1='DVC=$DVTU',MD1='MD=$MDTU',
         CATFN='FILESTAT=UNCAT',
         CATFY='FILESTAT=CAT';
    DALC $NMTU.$ROOT$ROOTDE,&RFTU;
    JUMP CONTINUE;
    JUMP DE$CTTU;
    DEY:
    CAT  $NMTU.$ROOT$ROOTDE,TYPE=FILE,SHARE=UNSPEC;
    DEN:
    PALC $NMTU.$ROOT$ROOTDE,
         UNIT=CYL,&DV1,GBL=(&MD1,SZ=8),INCRSZ=1,
         UFAS=(INDEXED=(CISZ=2048,RECSZ=90,KEYLOC=1,KEYSZ=17)),
         &CATF$CTTU;
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINALDJ                    *';
COMM '*             ========                    *';
COMM '*******************************************';
MVL  CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     CTDJY='FILESTAT=CAT,UNCATNOW',RFDJ=&CTDJ$CTDJ,
     CDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     CDJY='FILESTAT=CAT',RDJ=&CDJ$CTDJ,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     DV1='DVC=$DVDJ',MD1='MD=$MDDJ',
     CATFN='FILESTAT=UNCAT',
     CATFY='FILESTAT=CAT';
DALC $NMDJ.$ROOT$FILEDJ,&RFDJ;
JUMP CONTINUE;
JUMP DJ$CTDJ;
DJY:
CAT  $NMDJ.$ROOT$FILEDJ,TYPE=FILE,SHARE=UNSPEC;
DJN:
PALC $NMDJ.$ROOT$FILEDJ,
     UNIT=CYL,&DV1,GBL=(&MD1,SZ=1),INCRSZ=1,
     UFAS=(RELATIVE=(CISZ=2048,RECSZ=180)),
     &CATF$CTDJ;
```

2. TDS PREPARATION (DSINMPRE)

Two different preparations are possible:

1- Creation of an independent DSMS TDS

The DSMS TDS preparation is executed by a procedure supplied with the on-site Operating system, 'TP7PREP' (or MTPREP for GCOS7 V5) which is located in the SYS.HSLLIB library. The 'DSINMPRE' member included in the $NMLI.$LIBJCL library starts the execution of this procedure.

2- Integration of DSMS into an existing TDS

The preparation simply consists of the initialization of the TPRs which will receive on-line program compile-units, at the level of the corresponding SMLIB.

NOTE

The allocation of the $NMTD.$LIBSL library is generally executed during the preparation step. This allocation is not included in the DSMS installation procedures. Whichever solution is chosen, the presence of this library must be checked before the execution of the next step.

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*************************************************';
COMM '*                                               *';
COMM '*   PREPARATION PROCEDURE FOR TDS               *';
COMM '*                                               *';
COMM '*   IMPORTANT:                                  *';
COMM '*   THE "SYSFILE" PARAMETER INDICATES THE STATUS  *';
COMM '*   OF THE TREE SYSTEM FILES OF THE $NMTD TDS     *';
COMM '*   ($NMTD.CTLM, $NMTD.CTLN, $NMTD.RECOV)         *';
COMM '*                                               *';
COMM '*   - SYSFILE=CAT   FOR A CATALOGING OPTION       *';
COMM '*   - SYSFILE=RSD   FOR A RESIDENT OPTION         *';
COMM '*   - SYSFILE=RSDN  FOR A NON-RESIDENT OPTION     *';
COMM '*************************************************';
OVL  HOLD;
VL   PRY='SYSFILE=CAT,FILESTAT=CAT,CATNAME=$CTNM,IMPORT=NO',
     PREV6=TP7PREP,
     PRN='SYSFILE=RSD,FILESTAT=UNCAT',
     FF='$NMTD,$DVTD,$MDTD,$DVTD,$MDTD,DEAL=Y',
     GG='DBGSZ=1,MAXDBG=3,CBLSZ=1,SMSZ=7,MAXSM=10',
     VLVL='VL=('&FF','&PR$CTTD'';
IV   &PREV6 SYS.HSLLIB &VLVL,&GG);
SEND '===>  PREPARATION OF '$NMTD' SUCCESSFUL  <===';
```

3. INSTALLATION OF FILES AND PROGRAMS (DSINUNLD)

The installation of files and programs is executed by the 'DSINUNLD' member included in the $NMLI.$LIBJCL library.

This member consists of a sequence of LIBMAINT and CREATE. It may be logically split as follows:

```
. COBOL sources unloading:
      $NMTD.$LIBSL

. Programs (compile-units) unloading:
      $NMLI.$LIBCUB
      $NMLI.$LIBCUT

. Copy of the compile-units of on-line sub-programs in the
  permanent library:
      $NMLI.$LIBCUP

. Copy of the error messages file
      $NMBS.$ROOT$ROOTDE0 and $NMTU.$ROOT$ROOTDE

. Copy of the control sources library
      $NMLI.SOURCE

. Copy of the versions library
      $NMLI.ABOUT
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '************************************';
COMM '*    INSTALLATION TAPE UNLOADING    *';
COMM '*            LIBRARIES              *';
COMM '*           SYSTEM FILES            *';
COMM '*                                   *';
COMM '************************************';
MVL  LANG=E,
     DVT='DVC=$DVTP',MDT='MD=$TAPE',
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTTDN='FILESTAT=UNCAT,DVC=$DVTD,MD=$MDTD',
     RFTD=&CTTD$CTTD,
     RFTM='DVC=$DVTM,MD=$MDTM';
LMN  SL,IF=(SV.SL,&DVT,&MDT,FSN=ANY,END=LEAVE),
     LIB=($NMTD.$LIBSL,&RFTD),
     COM='MV INFILE:*,REPLACE;';
LMN  CU,IF=(SV.CUB,&DVT,&MDT,FSN=ANY,END=LEAVE),
     LIB=($NMLI.$LIBCUB,&RFLI),
     COM='MV INFILE:*,REPLACE;';
LMN  CU,IF=(SV.CUT,&DVT,&MDT,FSN=ANY,END=LEAVE),
     LIB=($NMLI.$LIBCUT,&RFLI),
     COM='MV INFILE:*,REPLACE;';
LIB  CU,IL1=($NMLI.$LIBCUT,&RFLI);
LMN  CU,LIB=($NMLI.$LIBCUP,&RFLI),
     COM='MV IL1:DSCHOI,REPLACE;STATUS RESET;
          MV IL1:ZAR980,REPLACE;STATUS RESET;
          MV IL1:ZAR985,REPLACE;STATUS RESET;
          MV IL1:PDTPDF,REPLACE;STATUS RESET;';
CR   IF=(SV.DE0,&DVT,&MDT,FSN=ANY,END=LEAVE),
     OF=($NMBS.$ROOT$ROOTDE0,&RFBS);
CR   IF=($NMBS.$ROOT$ROOTDE0,&RFBS),
     OF=($NMTU.$ROOT$ROOTDE,&RFTU);
LMN  SL,IF=(SV.SOURCE,&DVT,&MDT,FSN=ANY,END=LEAVE),
     LIB=($NMLI.SOURCE,&RFLI),
     COM='MV INFILE:*,REPLACE;';
LMN  SL,IF=(SV$LANG.ABOUT,&DVT,&MDT,FSN=ANY,END=LEAVE),
     LIB=($NMLI.ABOUT,&RFLI),
     COM='MV INFILE:*,REPLACE;';
```

4. DSMS TDS GENERATION (DSINMGEN)

This paragraph only describes the generation of an independent DSMS TDS.

The generation is executed by the DSINMGEN procedure included in the $NMLI.$LIBJCL library.

Warning:

Before generating the TDS, the user must choose one of the sources supplied in the $NMTD.$LIBSL library, then copy or rename it with the reserved name, 'STDS', and give the "PROGRAM-ID" clause the $NMTD value (default value: DS82).

```
.STDSD  :  TDS source for DSMS
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*************************************************';
COMM '*                                               *';
COMM '*              TDS SYSTEM GENERATION            *';
COMM '*                                               *';
COMM '*************************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     TDDVN=$DVTD,TDMDN=$MDTD,
     GENV6=TP7GEN,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     LMDVN=$DVLI,LMMDN=$MDLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     CTTDN='FILESTAT=UNCAT,DVC=$DVTD,MD=$MDTD',
     RFTD=&CTTD$CTTD,
     RFTM='DVC=$DVTM,MD=$MDTM';
OVL  HOLD;
LIB  CU,INLIB1=($NMLI.$LIBCUP,&RFLI);
LMN  CU,LIB=TEMP,
     COM='MV IL1:DSCHOI;STATUS RESET;
          MV IL1:PDTPDF;STATUS RESET;
          MV IL1:ZAR980;STATUS RESET;
          MV IL1:ZAR985;STATUS RESET;';
IV   &GENV6 SYS.HSLLIB
     VL=($NMTD,&TDDV$CTTD,&TDMD$CTTD,,,,,,$NMLI.$LIBLM,
        LMDVC=&LMDV$CTLI,LMMD=&LMMD$CTLI);
```

5. PROGRAM LINK-EDIT:

The program link-edit is executed by nine members included in the $NMLI.$LIBJCL library:

. Link-edit of on-line programs: 'DSINTLNK',

. Link-edit of the programs included in the print of query output (DPRT) procedure: 'DSINB1LN',

. Link-edit of the programs included in the entity extraction (DEXT) procedure: 'DSINB2LN',

. Link-edit of the programs included in the batch updating (DUPT) procedure: 'DSINB3LN',

. Link-edit of the programs included in the table extraction for revamped DSMS: 'DSINB4LN',

. Link-edit of the programs included in the DSMS archived journal extraction procedure: 'DSINB5LN'.

. Link-edit of the programs included in the DSMS codes and keywords loading procedure: 'DSINB6LN'.

. Link-edit of the programs included in DAF pre-processor: 'DSINB7LN'.

. Link-edit of the programs included in the other batch programs: 'DSINBLNK'.

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINTLNK                    *';
COMM '*             ========                    *';
COMM '*   LINKING OF ALL ON-LINE PROGRAMS       *';
COMM '*                                         *';
COMM '* $TPR0 : D S M S TPR  NAME IN SM LIBRARY  *';
COMM '* $TPR1 : D S M S TPR  NAME IN SM LIBRARY  *';
COMM '*                                         *';
COMM '*   PLEASE KEEP ALL LINKING REPORTS       *';
COMM '*                                         *';
COMM '*******************************************';
MVL  TPR3='$TPR0',TPR4='$TPR1',
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
OVL  HOLD;
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS0AA0,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00AA,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00AB,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00BA,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00B1,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00B2,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00B3,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00B4,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00B5,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00EA,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00E1,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00E2,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00E3,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00E4,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00E5,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00E6,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00FA,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00FB,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00HE,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00JO,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00KA,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00K1,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00K2,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00K3,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00LE,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00LS,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00MA,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00PA,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00P1,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00QA,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00QB,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00QC,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00Q1,&TPR3);
```

```
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00Q2,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00Q3,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00Q4,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00Q5,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00Q6,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00Q7,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00Q8,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00Q9,&TPR3);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00SA,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00SI,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00S1,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00S3,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00S4,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00S5,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00S6,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00S7,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00S8,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00S9,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00TA,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00TT,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00TU,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00TV,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00TW,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00TX,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00TY,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00TZ,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00T1,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00T2,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00T3,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00T4,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00T5,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00T6,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00T7,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00T8,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00T9,&TPR4);
IV   DSINTLSO,($NMLI.$LIBJCL,&RFLI),VL=(DS00UD,&TPR4);
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*****************************************';
COMM '*             DSINTLSO                  *';
COMM '*             ========                  *';
COMM '*   MODEL LINKER FOR ON-LINE PROGRAMS   *';
COMM '*   CALLED BY INVOKE STATEMENT FROM     *';
COMM '*****************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     LINKV6=TP7LINK&2,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTTDN='FILESTAT=UNCAT,DVC=$DVTD,MD=$MDTD',
     RFTD=&CTTD$CTTD,
     RFTM='DVC=$DVTM,MD=$MDTM';
LIB  CU,INLIB1=($NMLI.$LIBCUT,&RFLI);
LK   &1,SM,OUTLIB=($NMTD.$LIBSM,&RFTD),
     COMFILE=($NMTD.$LIBSL,&RFTD,SUBFILE=&LINKV6);
JUMP CONT,SEV,LE,1;JUMP CONTINUE;
SEND 'DSINTLNK: '&1' NOT LINKED INTO '&2'.';
CONT:
END:
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINB1LN                    *';
COMM '*             ========                    *';
COMM '*   LINKING OF THE PRINT OF QUERY OUTPUT   *';
COMM '*   PROCEDURE (DPRT) PROGRAMS              *';
COMM '*                                         *';
COMM '*******************************************';
MVL  CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     PGE='E',PGLNG=&PG$LANG;
OVL  HOLD;
LIB  CU,INLIB1=($NMLI.$LIBCUB,&RFLI);
LK   PDSB&PGLNG,OUTLIB=($NMLI.$LIBLM,&RFLI),
     COM='INCLUDE=(PDSA10,PDSRQ0,PDSRQ1,PDSRQ2,
     PDSRQ3,PDSE90,PDSFAC,PDSUAA);';
END:
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINB2LN                    *';
COMM '*             ========                    *';
COMM '*   LINKING OF THE ENTITY EXTRACTION      *';
COMM '*   PROCEDURE (DEXT) PROGRAMS             *';
COMM '*                                         *';
COMM '*******************************************';
MVL  CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     PGE='E',PGLNG=&PG$LANG;
OVL  HOLD;
LIB  CU,INLIB1=($NMLI.$LIBCUB,&RFLI);
LK   PDSEX&PGLNG,OUTLIB=($NMLI.$LIBLM,&RFLI),
     COM='INCLUDE=(PDSXCT,PDSRQ0,PDSRQ1,PDSXST,
     PDSXTR,PDSFAC,PDSUAA);';
END:
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*****************************************';
COMM '*             DSINB3LN                  *';
COMM '*             ========                  *';
COMM '*   LINKING OF THE BATCH UPDATE PROCEDURE  *';
COMM '*   (DUPT) PROGRAMS                       *';
COMM '*                                         *';
COMM '*****************************************';
MVL  CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI;
OVL  HOLD;
LIB  CU,INLIB1=($NMLI.$LIBCUB,&RFLI);
LK   PDSUP0,OUTLIB=($NMLI.$LIBLM,&RFLI),
     COM='INCLUDE=(PDCHOI,PDSFAC,PDSUAA,PDSUB1,
     PDSUB2,PDSUB3,PDSUB4,PDSUE1,PDSUE2,PDSUE3,
     PDSUK1,PDSUQ1,PDSUQ2,PDSUQ5,PDSUQ6,PDSUQ7,
     PDSUQ8,PDSUS1,PDSUS3,PDSUS4,PDSUS6,PDSUSI,PDSUT1,
     PDSUT2,PDSUT3,PDSUT4,PDSUT5,PDSUT6,PDSUT7,
     PDSUT8,PDSUT9,PDSUTT,PDSUTV,PDSUTW,PDSUTX,
     PDSUTY,PDSUTZ,PDSUP1);';
END:
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINB4LN                    *';
COMM '*             ========                    *';
COMM '*   LINK    OF THE TABLE EXTRACTION       *';
COMM '*   PROCEDURE (DEXH) PROGRAMS             *';
COMM '*                                         *';
COMM '*******************************************';
MVL  CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI;
OVL  HOLD;
LIB  CU,INLIB1=($NMLI.$LIBCUB,&RFLI);
LK   PDSXTH,OUTLIB=($NMLI.$LIBLM,&RFLI),
     COM='INCLUDE=(PDSFAC,PDSUAA);';
END:
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINB5LN                    *';
COMM '*             ========                    *';
COMM '*   LINK    OF THE JOURNAL EXTRACTION     *';
COMM '*   PROCEDURE (DXBJ) PROGRAMS             *';
COMM '*                                         *';
COMM '*******************************************';
MVL  CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI;
OVL  HOLD;
LIB  CU,INLIB1=($NMLI.$LIBCUB,&RFLI);
LK   PDS700,OUTLIB=($NMLI.$LIBLM,&RFLI),
     COM='INCLUDE=(PDSFAC,PDSUAA);';
END:
```

```
COMM 'DSMS 2.5';
COMM '*********************************************';
COMM '*             DSINB6LN                      *';
COMM '*             ========                      *';
COMM '*                                           *';
COMM '*   LINK OF DREN PROCEDURE PROGRAMS         *';
COMM '*                                           *';
COMM '*********************************************';
MVL  CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     PGE='E',PGLNG=&PG$LANG;
OVL  HOLD;
LIB  CU,INLIB1=($NMLI.$LIBCUB,&RFLI);
LK   PDSMS&PGLNG,OUTLIB=($NMLI.$LIBLM,&RFLI),
     COM='INCLUDE=(PDSRCT,PDSRMS,PDSRFU,PDSJMS,PDSFAC,PDSUAA);';
END:
```

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINB7LN                    *';
COMM '*             ========                    *';
COMM '*******************************************';
MVL  CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI;
OVL  HOLD;
LIB  CU,INLIB1=($NMLI.$LIBCUB,&RFLI);
LK   DAFD10,OUTLIB=($NMLI.$LIBLM,&RFLI),
     COM='INCLUDE=PDSFAC;';
END:
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*******************************************';
COMM '*             DSINBLNK                    *';
COMM '*             ========                    *';
COMM '*   LINKING OF ALL BATCH PROGRAMS         *';
COMM '*                                         *';
COMM '*   PLEASE KEEP ALL LINK REPORTS          *';
COMM '*                                         *';
COMM '*******************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
OVL  HOLD;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSBAS;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSINI;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSLVB;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR10;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR20;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR30;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR40;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR8B;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR8C;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR8X;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR8Q;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR8R;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR15;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDSR5J;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=REP2PJ;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS300;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS320;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS380;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS381;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS400;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS450;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS500;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS600;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS610;
IV   DSINBLSO,($NMLI.$LIBJCL,&RFLI),VL=PDS900;
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*****************************************';
COMM '*             DSINBLSO                  *';
COMM '*             ========                  *';
COMM '*   MODEL LINKER FOR BATCH PROGRAMS     *';
COMM '*   CALLED BY INVOKE STATEMENT FROM     *';
COMM '*   PBINBLNK.                           *';
COMM '*****************************************';
MVL  CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
LIB  CU,INLIB1=($NMLI.$LIBCUB,&RFLI);
LK   &1,OUTLIB=($NMLI.$LIBLM,&RFLI);
JUMP CONT,SEV,LE,1;JUMP CONTINUE;
SEND 'DSINBLNK: '&1' NOT LINKED';
CONT:
END:
```

6. CREATION OF THE DSMS TEST DATABASE (DSINDRS1)

The creation of the DSMS test database is executed by the DSINDRS1 member (DSMS restoration procedure) using as input the backup included in the installation tape (physical name: SV.BB).

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
MVL  PACDMB='MBDRST',
     DVT='DVC=$DVTP',MDT='MD=$TAPE',
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DRS1,
     OF=(TMBREST,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
COMM '*** ALLOCATION : DA,DC,DJ,DX ***';
IV   DSINALDA ($NMLI.$LIBJCL,&RFLI);
IV   DSINALDC ($NMLI.$LIBJCL,&RFLI);
IV   DSINALDJ ($NMLI.$LIBJCL,&RFLI);
IV   DSINALDX ($NMLI.$LIBJCL,&RFLI);
COMM '*** PDS400 ***';
STEP PDS400,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDMB,TMBREST,TEMPRY,&RFTM;
     ASG PACDDJ,$NMDJ.$ROOT$FILEDJ,&RFDJ;
     ASG PACDBB,SV.BB,FILESTAT=UNCAT,&DVT,&MDT,FSN=ANY;
     ASG PACDDC,$NMTU.$ROOT$FILEDC,&RFTU;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU;
     ASG PACDDX,$NMTU.$ROOT$FILEDX,&RFTU;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU;
     ASG PACDMS,TPACDMS,TEMPRY,&RFTM,END=PASS;
     ASG PACDRU,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP END,SW30,EQ,1;
COMM '*** PDS450 ***';
STEP PDS450,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ 60;
     ASG PACDMS,TPACDMS,TEMPRY,&RFTM;
     ASG PACDDA,$NMTU.$ROOT$FILEDA,&RFTU;
     ASG PACDDC,$NMTU.$ROOT$FILEDC,&RFTU;
     ASG PACDDX,$NMTU.$ROOT$FILEDX,&RFTU;
     ASG PACDBJ,$NMBU.$ROOT$FILEBJ,&RFBU;
     ASG PACDDE,$NMTU.$ROOT$ROOTDE,&RFTU;
     ASG PACDRU,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP END;
ERR:
SEND ' DSINDRS1 - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

7. TDS SUBMISSION (DEXTDDS)

The following submission procedure is supplied:

DSEXTDDS: execution of the TDS which corresponds to the 'STDSD' source (DSMS alone).

The parameter '&1' must be set to 'STEP1' in order to load the TPRs in BACKING STORE (VL=STEP1).

The second step of this procedure automatically re-initializes the following files:

```
.DS802HE: Screen backup for documentation help call

.DS802JB: Relative file used by the 'JOB' function.
```

SOURCE OF THE PROCEDURE

```
COMM 'DSMS 2.5';
COMM '*****************************************';
COMM '* EXECUTION OF TDS : $NMTD                *';
COMM '*         D S M S                         *';
COMM '* &1 = BACKING-STORE : STEP1 , ELSE  STEP2 *';
COMM '*****************************************';
MVL  STEP1,START='WARM',
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     CTDJN='FILESTAT=UNCAT,DVC=$DVDJ,MD=$MDDJ',
     RFDJ=&CTDJ$CTDJ,
     CTTDN='FILESTAT=UNCAT,DVC=$DVTD,MD=$MDTD',
     RFTD=&CTTD$CTTD,
     RFTM='DVC=$DVTM,MD=$MDTM';
JUMP &1;
STEP1:
LIB  SM,INLIB1=($NMTD.$LIBSM,&RFTD);
SYSMAINT COMFILE=*DEMER;
$IN DEMER;
SM;
LOAD MODULE=$TPR0 INPUT=INLIB1 REPLACE;
LOAD MODULE=$TPR1 INPUT=INLIB1 REPLACE;
$EIN DEMER;
STEP2:
IV   DSINALJB ($NMLI.$LIBJCL,&RFLI);
IV   DSINALHE ($NMLI.$LIBJCL,&RFLI);
JOBLIB SM,$NMTD.$LIBSM;
STEP $NMTD,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA,OPTIONS=&START;
     SZ  150,POOLSZ=70,NBBUF=70;
     ASG H_BJRNL,DVC=$DVTM,MD=$MDTM,TEMPRY,NEXT,POOL;
     DEF H_CTLM,JOURNAL=BEFORE;
     ASG DBUGFILE,$NMTD.DEBUG,&RFTD,
         SHARE=DIR;
     ASG DS80IE,$NMTU.$ROOT$ROOTDE,
         &RFTU,ACC=WRITE,SHARE=MONITOR;
     DEF DS80IE,JOURNAL=BEFORE;
     ASG DS80IA,$NMTU.$ROOT$FILEDA,
         &RFTU,ACC=WRITE,SHARE=MONITOR;
     DEF DS80IA,JOURNAL=BEFORE;
     ASG PB80DC,$NMTU.$ROOT$FILEDC,
         &RFTU,ACC=WRITE,SHARE=MONITOR;
     DEF PB80DC,JOURNAL=BEFORE;
     ASG DS80IJ,$NMDJ.$ROOT$FILEDJ,
         &RFDJ,ACC=WRITE,SHARE=MONITOR;
     DEF DS80IJ,JOURNAL=BEFORE;
     ASG DS80IX,$NMTU.$ROOT$FILEDX,
         &RFTU,ACC=WRITE,SHARE=MONITOR;
     DEF DS80IX,JOURNAL=BEFORE;
     ASG DS80JB,$NMTU.$ROOT$FILEJB,
         &RFTU,ACC=WRITE,SHARE=MONITOR;
     DEF DS80JB,JOURNAL=BEFORE;
     ASG DS80HE,$NMTU.$ROOT$FILEHE,
         &RFTU,ACC=WRITE,SHARE=MONITOR;
```

```
        DEF DS80HE,JOURNAL=BEFORE;
        ASG DS80PA,$NMTU.$ROOT$FILEPA,
            &RFTU,ACC=WRITE,SHARE=MONITOR;
        DEF DS80PA,JOURNAL=BEFORE;
    ESTP;
    FILLIST  IF=($NMTU.$ROOT$FILEDX,&RFTU) USAGE;
```

18.5. INTEGRATION IN AN EXISTING TDS

INSTALLATION IN AN EXISTING TDS

The compatibility of DSMS parameters with existing names must be carefully checked (for example: $NMTD, $TPR, $LIBSM, IBLM).

The installation process differs somewhat from the process described in Subchapter "INSTALLATION PROCESS":

1. System file allocation:

   This step must be executed as is.

2. Preparation of the DSMS TDS:

   This step is not executed. However, the initialization of the TPRn defined by $TPR0 and $TPR1 must be verified, as well as the size of the $NMTD.$LIBSM library which must be large enough for DSMS (about 30 MS/D500-type cylinders are necessary).

3. Unloading of files and programs:

   This step must be executed as is.

4. Generation of DSMS TDS:

   In addition to the preparation described in Subchapter "INSTALLATION PROCESS", the existing STDS and the source supplied (STDSD) must be merged, integrating in the proper sections the clauses USE, SELECT, FD, FILE-INTEGRITY, PROCESSING-MODE, MESSAGE, and the WORKING STORAGE SECTION zones. The user must also check that the length of the message ("MESSAGE-LENGTH" clause) is greater than 6,000.

A detailed description of the step follows:

- In the existing TDS source, add the DSMS on-line sub- programs: take, from the DSMS-supplied TDS source, the lines USE DSCHOI, USE ZAR980, USE ZAR985, and USE PDTPDF (only for DAF function).

- In FILE-CONTROL, add the DSMS file allocations: take, from the DSMS-

supplied TDS source, the lines between FILE-CONTROL and TDS-FILE-DEFINITION.

- In TDS-FILE-DEFINITION, add the FD clauses from the DSMS files: take, from the DSMS-supplied TDS source, the lines between TDS-FILE-DEFINITION and PROCESSING-CONTROL.

- In PROCESSING-CONTROL, add the file-integrity of DSMS files: take, from the DSMS-supplied TDS source, the lines between PROCESSING-CONTROL and WORKING-STORAGE.

- In WORKING-STORAGE, add the file-status and relative keys of the DSMS files: take, from the DSMS-supplied TDS source, the lines between WORKING-STORAGE and TRANSACTION-SECTION (except the 01 RELKEY level).

- In TRANSACTION-SECTION, add the description of the DSMS transaction: take, from the DSMS-supplied TDS source, the lines between TRANSACTION-SECTION to the end.

IMPORTANT:

- If the DSMS file names are already used in the existing TDS (for example, HE and JB in VA Pac), then it is necessary to assign the files different names in the lines to be copied from the DSMS-supplied TDS.

- If the existing TDS includes VA Pac, the PB80DC file already exists. It is not necessary to take it again from the applicable DSMS-supplied TDS.

ADAPTATION OF GENERATION JCL OF THE EXISTING TDS (MGEN)

- Add the DSMS on-line sub-programs (DSCHOI, ZAR980, ZAR985, and PDTPDF for DAF function):

Under the line LIB CU,INLIB1=(CUP lib from existing TDS) add: INLIB2=(DSMS permanent CUP lib).

Under the line LMN CU, add: MV IL2:DSCHOI; STATUS RESET; MV IL2:ZAR980; STATUS RESET; MV IL2:ZAR985; STATUS RESET; MV IL2:PDTPDF; STATUS RESET;

5. Link-edit of batch program monitors, and link-edit of the TPRS in the SMLIB prepared for DSMS:

   This step must be executed as is.

6. Creation of a DSMS test database:

   This step must be executed as is.

7. TDS submission:

   The DSEXTDDS procedure may be used, but, in this case, the only files assigned and opened are the DSMS files. To assign and open all the files used in TDS applications, the JCL supplied (DSEXTDDS) and the existing submission JCL must be merged, transferring the "ASSIGN" and "DEFINE" commands.

A detailed description of this step follows:

- In the SYSMAINT step, add the load of the two DSMS-defined TPRs: from the DSEXTDDS JCL, take the applicable load lines.

- Before the TDS submission step, add the DSMS HE and JP file allocations: from the DSEXTDDS JCL, take the two corresponding INVOKE lines.

- In the TDS submission step, add the DSMS file assignments: from the DSEXTDDS JCL, take the "ASSIGN" and "DEFINE" lines of the following files: DS80IA, DS80IE, DS80IJ, DS80IX, DS80JB, DS80HE, DS80PA and PB80DC. (If insertion in a TDS which contains VA Pac, the PB80DC file is already assigned; it is not necessary to retrieve it.)

18.6. INSTALLATION OF DAF FUNCTION ENVIRONMENT

COMPLEMENT: INTALLATION OF DAF ENVIRONMENT

The use of the DAF function involves transforming the DSMS Database SQL access queries described in user programs, by generating data and calling database access sub-program in the COBOL source generated from these programs.

DAF preprocessor processes the generated programs in order to perform this transformation.

It includes the DAFD10 program installed in the batch load-module library.

Several methods are available to process the generated programs using DAF:

- use of the DPDF procedure:

  . by calling this procedure in the Optional Control Commands in front of/in back of program, which are combined with the compilation-link-edit JCL;

  . by calling this procedure after the execution of the standard GPRT procedure, from which the generated flow will be retrieved;

  . by using any other methods best suited for the characteristics of the site.

  (See the DPDF section in Chapter STANDARD PROCEDURES)

Two DAF sub-programs are provided in the installation deck:

- PDTPDF for DAF standard queries
- PDSFAC for physical access to DSMS database

The work file necessary for the operation of DAF in on-line mode has an imposed IFN under TDS: DS80PA. It must be unique for all programs which access the same DSMS database.

Data Element, Data Structure and Segment entities used to write programs involving DAF, are provided as batch transactions in the DAFDICF member (or DAFDICA for the English version) of the SRC Complements' library.

IMPORTANT:

Loading the 'DAF dictionary' in the VA Pac database via the UPDT batch update procedure is the responsibility of the Database Administrator, who must make sure that the codes of the entities provided do not conflict with entities that are already defined in the Database.

In order to avoid compatibility conflicts between the site's Dictionary and entities provided for the DAF function, it is recommended to create an independent library network for the writing of the site's DAF utilities. However it can be installed in the same library as the PAF Dictionary.

```
COMM 'DSMS 2.5';
COMM '*******************************************************';
COMM '*                    JCLDAF                           *';
COMM '*                   ========                          *';
COMM '*          EXAMPLE OF AN EXECUTION JCL                *';
COMM '*          OF A BATCH USER DAF PROGRAM                *';
COMM '*******************************************************';
MVL  SYSDAF='$NMTU.$ROOT$FILEPA',USER='$USER',
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     RFTM='DVC=$DVTM,MD=$MDTM';
IV   DSINALPA,($NMLI.$LIBJCL,&RFLI),
     VL=(USER=&USER,SYSDAF=&SYSDAF);
COMM '*** WITHDAF ***';
STEP WITHDAF,FILE=($NMLI.$LIBLM,&RFLI),REPEAT;
     SZ  200;
     ASG DS80IE,$NMTU.$ROOT$ROOTDE,&RFTU,
         ACC=READ,SHARE=MONITOR;
     DEF DS80IE,READLOCK=STAT;
     ASG DS80IA,$NMTU.$ROOT$FILEDA,&RFTU,
         ACC=READ,SHARE=MONITOR;
     DEF DS80IA,NBBUF=4;
     ASG DS80IX,$NMTU.$ROOT$FILEDX,&RFTU;
     ASG PB80DC,$NMTU.$ROOT$FILEDC,&RFTU;
     ASG SYSDAF,&SYSDAF&USER,&RFTU;
COMM 'ASG --------';
COMM 'ASG --------';
ESTP;
```

18.7. REINSTALLATION

REINSTALLATION

The DSMS system must be reinstalled when the user receives a sub-release which corrects anomalies. This sub-release is identified by a number and is usually supplied as:

. a complete installation tape of the product,
. A list of the corrected anomalies,
. Possibly a document which completes the explanation about re-installation provided in this Subchapter.

Only program libraries and system files are generally concerned by this new release.

To reinstall the DSMS system, it is necessary to execute most of the procedures used at the time of the first installation.

The installation tape is described in a specific Subchapter.

It includes the following steps:

. Installation tape backup,

. Re-allocation of the system files,

. Tape unloading,

. TDS re-generation,

. Program link-edit.

UNLOADING OF INSTALLATION TAPE

Execution of the 'DSINUNLD' procedure.
The unloading of the installation tape is explained in detail in Subchapter 'Installation Process'.

TDS GENERATION

TDS must be generated in order to recognize the new version of the DSMS routines used on-line (DSCHOI, ZAR980, and ZAR985).

PROGRAM LINK-EDIT

Execution of the link-edit procedures:

. DSINTLNK link of TPRs,

. DSINBLNK link of batch programs,

. DSINB1LN link of the DPRT procedure programs,

. DSINB2LN link of the DEXT procedure programs,

. DSINB3LN link of the DUPT procedure programs,

. DSINB4LN link of the DEXH procedure programs.

. DSINB5LN link of the DXBJ procedure programs.

. DSINB6LN link of the DREN procedure programs.

. DSINB7LN link of the DAF pre-processor.

19. RETRIEVAL OF DSMS 8.0.1 --> DSMS 2.5 (DR80)

19.1. 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).

19.2. 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.

19.3. BACKUP RETRIEVAL

DR80: DESCRIPTION OF STEPS

This procedure includes the following steps:

.Recognition of input : CREATE

.Preparation of conversion : PDSR8B

.Conversion of 8.0.2 backup : PDSR8C

.Conversion of 1.5 backup : PDSR15

.Shift of archived file (BB) generations

RECOGNITION OF INPUT: CREATE

This utility creates a temporary file from a file defined under IOF. It is found at the beginning of all procedures which include user input.

. Input file : $INPUT

. Output file (TEMPRY) : TMBRP80
(recsize 80, Cisize 2048)

CONVERSION PREPARATION: PDSR8B

.Permanent input file
-Backup of DSMS 8.0.1 DSMS database
External name &OLDBB

.Output work files:
TPACDIQ temporary file (recsize 284)
TPACDIT temporary file (recsize 104)
TPACDIW temporary file (recsize 92)
TPACDLA temporary file (recsize 1)

.Sort files:

.Input transaction file:
TMBDR80 temporary file

.Output report:
-Retrieval report

8.0.2 BACKUP CONVERSION: PDSR8C

.Input work files:
TPACDIQ temporary file (recsize 284)
TPACDIT temporary file (recsize 104)
TPACDIW temporary file (recsize 92)

TPACDLA temporary file (recsize 1)

.Permanent input file:
-Backup of DSMS 8.0.1 database
External name &OLDBB

.Permanent output file:
-Backup of converted DSMS database
External name &TPACDB1

.Input transaction file:

TMBRP80 temporary file

.Sort files

.Output report:
-Retrieval report
PAC7IO and PAC7EI

CONVERSION OF BACKUP 1.2 TO 2.5 FORMAT: PDSR15

.Permanent input file:
-Backup file of DSMS 1.2 or 1.5 Database
External name &TPACDB1

.Permanent output file:
-Backup of converted DSMS Database
External name $NMBU.$ROOT$FILEBB/G+1

.Output report:
-Printing report
PACDIK

SHIFT OF ARCHIVED FILE GENERATIONS: SHIFT

This step performs a shift of the archive file generations
(BB).

19.4. EXECUTION JCL

```
COMM '*************************************************';
COMM '*  D S M S RETRIEVAL 8.0.1 TO 2.5                *';
COMM '*************************************************';
MVL  PACDBB='$NMBU.$ROOT$FILEBB',SIZEWK=10,OLDBB=,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DR80,
     OF=(TMBRP80,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
COMM '*** PDSR8B ***';
STEP PDSR8B,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDBB,&OLDBB;
     ASG PACDIQ,TPACDIQ,TEMPRY,&RFTM,END=PASS;
     ASG PACDIT,TPACDIT,TEMPRY,&RFTM,END=PASS;
     ASG PACDIW,TPACDIW,TEMPRY,&RFTM,END=PASS;
     ASG PACDMB,TMBRP80,TEMPRY,&RFTM,END=PASS;
     ASG PACDLA,TPACDLA,TEMPRY,&RFTM,END=PASS;
     ASG PAC7EI,SYS.OUT;
     ASG PACDIK,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
COMM '*** PDSR8C ***';
STEP PDSR8C,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDBB,&OLDBB;
     ASG PACDB1,TPACDB1,TEMPRY,&RFTM,END=PASS;
     ALC TPACDB1,UNIT=CYL,SZ=&SIZEWK,INCRSZ=2;
     ASG PACDIQ,TPACDIQ,TEMPRY,&RFTM;
     ASG PACDIT,TPACDIT,TEMPRY,&RFTM;
     ASG PACDIW,TPACDIW,TEMPRY,&RFTM;
     ASG PACDLA,TPACDLA,TEMPRY,&RFTM;
     ASG PACDMB,TMBRP80,TEMPRY,&RFTM,END=PASS;
     ASG PACDIO,SYS.OUT;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT $NMBU.$ROOT$FILEBB;
JUMP END;
COMM '*** PDSR15 ***';
STEP PDSR15,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  100;
     ASG PACDBB,&TPACDB1,TEMPRY,&RFTM;
     ASG PACDB1,&PACDBB/G+1,&RFBU;
     ASG PACDIK,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT &PACDBB;
JUMP END;
ERR:
SEND ' DSINDR80 - ABNORMAL END OF RUN (I/O ERROR) ';
```

```
    LET  SEV 3;
    END:
```

20. RETRIEVAL OF DSMS 8.0.2 01/02 --> DSMS 2.5 (DR8Q)

20.1. 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.

20.2. BACKUP RETRIEVAL

DR8Q: DESCRIPTION OF STEPS

QUERY RETRIEVAL: PDSR8Q

.Permanent input file:
-Backup of DSMS 8.0.2 01/02 Database
External name &OLDBB

.Output work file:
TPACDIQ temporary file (recsize 284)

.Sort files:

MERGE: PDSR8R

.Input work file
TPACDIQ temporary file (recsize 284)

.Permanent input file:
-Backup of DSMS Database 8.0.2 01/02
External name &OLDBB

.Output Permanent file :
-Backup of retrieved DSMS Database
External name &TPACDB1

CONVERSION OF BACKUP 1.2 TO 2.5 FORMAT: PDSR15

.Permanent input file:
-Backup file of DSMS 1.2 or 1.5 Database
External name &TPACDB1

.Permanent output file:
-Backup of converted DSMS Database
External name $NMBU.$ROOT$FILEBB/G+1

.Output report:
-Printing report
PACDIK

SHIFT OF ARCHIVED FILE GENERATIONS: SHIFT

This step performs a shift of the archive file generations (BB).

20.3. EXECUTION JCL

```
COMM '*************************************************';
COMM '*          DSMS   : RETRIEVAL 8.0.2 01 VERSION    *';
COMM '*************************************************';
MVL  PACDBB='$NMBU.$ROOT$FILEBB',OLDBB='BB802',SIZEWK=10,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     RFTM='DVC=$DVTM,MD=$MDTM';
COMM '*** PDSR8Q ***';
STEP PDSR8Q,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDBB,&OLDBB;
     ASG PACDIQ,TPACDIQ,TEMPRY,&RFTM,END=PASS;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
COMM '*** PDSR8R ***';
STEP PDSR8R,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDBB,&OLDBB;
     ASG PACDB1,TPACDB1,TEMPRY,&RFTM,END=PASS;
     ASG PACDIQ,TPACDIQ,TEMPRY,&RFTM;
     ASG PAC7EI,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
COMM '*** PDSR15 ***';
STEP PDSR15,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  100;
     ASG PACDBB,TPACDB1,TEMPRY,&RFTM;
     ASG PACDB1,&PACDBB/G+1;
     ASG PACDIK,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT &PACDBB;
JUMP END;
ERR:
SEND ' DSINDR8Q - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

21. RETRIEVAL OF DSMS 8.0.2, COMPATIBLE VA PAC 8.0.1

21.1. 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.

21.2. '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.

21.3. 'DR8X' PROCEDURE

DR8X: DESCRIPTION OF STEPS

```
CONVERSION FOR VA PAC 8.0.2:  PDSR8X

.Permanent input file:
 -Backup of DSMS 8.0.1 Database
  External name &OLDBB

.Sort files:

.Input transaction file:
  TMBRPDC temporary file

.Permanent output file:
 -Backup of converted DSMS Database
  External name &TPACDB1

.Output report
 -Retrieval report
  PACDIK and PACDEI for anomalies

CONVERSION OF BACKUP 1.2 TO 2.5 FORMAT:  PDSR15

.Permanent input file:
 -Backup file of DSMS 1.2 or 1.5 Database
  External name &TPACDB1

.Permanent output file:
 -Backup of converted DSMS Database
  External name $NMBU.$ROOT$FILEBB/G+1

.Output report:
 -Printing report
  PACDIK



SHIFT OF ARCHIVED FILE GENERATIONS: SHIFT

This step performs a shift of the archive file generations
(BB).
```

21.4. 'DR8X' PROCEDURE - EXECUTION JCL

```
MVL  PACDBB='$NMBU.$ROOT$FILEBB',SIZEWK=10,OLDBB=,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     RFTM='DVC=$DVTM,MD=$MDTM';
CR   IF=*DR8X,
     OF=(TMBRPDC,TEMPRY,&RFTM,END=PASS),
     OUTDEF=(CISZ=2048,RECSZ=80,RECFORM=FB);
COMM '*** PDSR8X ***';
STEP PDSR8X,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  60;
     ASG PACDBB,&OLDBB;
     ASG PACDMB,TMBRPDC,TEMPRY,&RFTM,END=PASS;
     ASG PACDB1,TPACDB1,TEMPRY,&RFTM,END=PASS;
     ALC TPACDB1,UNIT=CYL,SZ=&SIZEWK,INCRSZ=2;
     ASG PAC7EI,SYS.OUT;
     ASG PACDIK,SYS.OUT;
     SWK WKDISK=(SZ=&SIZEWK,&RFTM);
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT $NMBU.$ROOT$FILEBB;
JUMP END;
COMM '*** PDSR15 ***';
STEP PDSR15,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  100;
     ASG PACDBB,TPACDB1,TEMPRY,&RFTM;
     ASG PACDB1,&PACDBB/G+1,&RFBU;
     ASG PACDIK,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT &PACDBB;
JUMP END;
ERR:
SEND ' DSINDR8X - ABNORMAL END OF RUN (I/O ERROR) ';
END:
```

22. RETRIEVAL OF DSMS 1.2 OR 1.5 --> DSMS 2.5

22.1. 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.

22.2. '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
  PACDBB  :    External name &OLDBB

.Permanent output file:
 -Retrieved DSMS database backup
  PACDB1  :    External name $NMBU.$ROOT$FILEBB/G+1

.Output report:
 -Printing report
  PACDIK



SHIFT OF ARCHIVED FILE GENERATIONS: SHIFT

This step performs a shift of the archive file generations
(BB).
```

22.3. 'DR15' PROCEDURE - EXECUTION JCL

```
COMM '************************************************';
COMM '*  DSMS 1.2 OU 1.5 DATABASE RETRIEVAL           *';
COMM '************************************************';
MVL  OLDBB='BB1215',
     PACDBB='$NMBU.$ROOT$FILEBB',SIZEWK=10,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     RFTM='DVC=$DVTM,MD=$MDTM';
COMM '*** PDSR15 ***';
STEP PDSR15,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  100;
     ASG PACDBB,&OLDBB;
     ASG PACDB1,&PACDBB/G+1;
     ASG PACDIK,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT &PACDBB;
JUMP END;
ERR:
SEND ' DSINDR15 - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

22.4. 'DR5J' PROCEDURE - DESCRIPTION OF STEPS

```
RETRIEVAL OF JOURNAL FILE: PDSR5J

.Permanent input file:
 -Sequential image of journal file, Rel. 1.2 or 1.5
  PACDBJ  :    External name &OLDBJ

.Permanent output file:
 -Journal retrieved in the 2.5 format
  PACDJB  :    External name $NMBU.$ROOT$FILEBJ/G+1

.Output report:
 -Printing report
  PACDIK
```

SHIFT OF ARCHIVED FILE GENERATIONS: SHIFT

This step performs a shift of the archive file generations (BJ).

22.5. 'DR5J' PROCEDURE - EXECUTION JCL

```
COMM '************************************************';
COMM '*  DSMS JOURNAL 1.2 OR 1.5 RETRIEVAL            *';
COMM '************************************************';
MVL  OLDBJ='BJ1215',
     PACDBJ='$NMBU.$ROOT$FILEBJ',SIZEWK=2,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     RFTM='DVC=$DVTM,MD=$MDTM';
COMM '*** PDSR5J ***';
STEP PDSR5J,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA;
     SZ  100;
     ASG PACDBJ,&OLDBJ;
     ASG PACDJB,&PACDBJ/G+1;
     ASG PACDIK,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
SHIFT &PACDBJ;
JUMP END;
ERR:
SEND ' DSINDR5J - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```

23. REPLACEMENT OF LOW-VALUES BY BLANKS (DLVB)

23.1. 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

23.2. DLVB: PARAMETERS-DESCRIPTION OF STEPS

DLVB: DESCRIPTION OF STEPS

```
REPLACEMENT OF LOW-VALUES BY BLANKS: PDSLVB

. Enter &OPTS='DATA' in order to keep only the DATA records
  in the output file.
  Do not enter anything to keep the INDEX and the DATA
  records.

.Input file:
 -Database backup
  PACDBB  :   External name $NMBU.$ROOT$FILEBB

.Output file:
 -New Database backup
  PACDB1  :   External name $NMBU.$ROOT$FILEBB/G+1



SHIFT OF ARCHIVED FILE GENERATIONS: SHIFT

This step performs a shift of the archive file generations
(BB).
```

23.3. DLVB: EXECUTION JCL

```
COMM '*******************************************************';
COMM '*                   DSEXDLVB                          *';
COMM '*                   ========                          *';
COMM '*  D S M S           :  SUPPRESS LOW-VALUES IN BB     *';
COMM '*                                                     *';
COMM '*******************************************************';
MVL  PACDBB='$NMBU.$ROOT$FILEBB',OPTS=,
     CTTUN='FILESTAT=UNCAT,DVC=$DVTU,MD=$MDTU',
     RFTU=&CTTU$CTTU,
     CTBSN='FILESTAT=UNCAT,DVC=$DVBS,MD=$MDBS',
     RFBS=&CTBS$CTBS,
     CTLIN='FILESTAT=UNCAT,DVC=$DVLI,MD=$MDLI',
     RFLI=&CTLI$CTLI,
     CTBUN='FILESTAT=UNCAT,DVC=$DVBU,MD=$MDBU',
     RFBU=&CTBU$CTBU,
     RFTM='DVC=$DVTM,MD=$MDTM';
COMM '*** PDSLVB ***';
STEP PDSLVB,FILE=($NMLI.$LIBLM,&RFLI),DUMP=DATA,
     OPTIONS=''&OPTS'';
     SZ  100;
     ASG PACDBB,&PACDBB,&RFBU;
     ASG PACDB1,&PACDBB/G+1,&RFBU;
     ASG PAC7EI,SYS.OUT;
ESTP;
JUMP ERR,SW20,EQ,1;
JUMP ERR,SW30,EQ,1;
SHIFT $NMBU.$ROOT$FILEBB;
JUMP END;
ERR:
SEND ' DSINDLVB - ABNORMAL END OF RUN (I/O ERROR) ';
LET  SEV 3;
END:
```