

z/OS



DFSMSdss Storage Administration Reference

z/OS



DFSMSdss Storage Administration Reference

Note

Before using this information and the product it supports, be sure to read the general information under "Notices" on page 367.

Seventh Edition, April 2006

This edition applies to Version 1 Release 7 (Refresh Version) of z/OS® (5694-A01), Version 1 Release 7 (Refresh Version) of z/OS.e (5655-G52), and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC35-0423-05.

IBM® welcomes your comments. A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address:

International Business Machines Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+845+432-9405

FAX (Other Countries):

Your International Access Code +1+845+432-9405

IBMLink™ (United States customers only): IBMUSM10(MHVRCFS)

Internet e-mail: mhvrcfs@us.ibm.com

World Wide Web: <http://www.ibm.com/servers/eserver/zseries/zos/webqs.html>

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this document
- Page number or topic related to your comment

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.

© Copyright International Business Machines Corporation 1984, 2006. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About This Document	vii
Required product knowledge	vii
Referenced documents	vii
Accessing z/OS DFSMS documents on the Internet	vii
Using LookAt to look up message explanations	vii
Summary of Changes	ix
Summary of Changes for SC35-0424-06 z/OS Version	
1 Release 7	ix
New Information	ix
Changed Information	ix
Summary of Changes for SC35-0424-05 z/OS Version	
1 Release 7	ix
New Information	x
Changed Information	x
Deleted Information	x
Summary of Changes for SC35-0424-04 z/OS Version	
1 Release 6	x
New Information	x
Changed Information	xi
Chapter 1. Specifying DFSMSdss Commands	1
Command Syntax	1
How Many Subkeywords Are Allowed?	2
Specifying Subkeywords in a Command Data Set	2
How to Read Syntax Diagrams	3
JCL That You Need	5
How to Control DFSMSdss through PARM	
Information in the EXEC Statement	6
Examples of Invoking DFSMSdss with JCL	9
Chapter 2. Filtering—Choosing the Data Sets You Want Processed	11
How DFSMSdss Filters Data Sets	11
Virtual Storage Access Method (VSAM) Data Set Considerations	11
Filtering by Data Set Names	12
Using a * in Partially Qualified Data Set Names	12
Using a % in Partially Qualified Data Set Names	12
Examples of Fully and Partially Qualified Data Set Names	13
Relative Generation Filtering	14
Filtering by Data Set Characteristics	14
Some Examples of the BY Keywords	19
Standard Catalog Search Order	19
Broken Data Set Considerations	19
Chapter 3. Syntax—Function Commands	21
What DFSMSdss Commands Do	21
Building the Stand-Alone IPL-able Core Image	21
Using DUMP and RESTORE for Backup and Recovery	21
Moving Data with COPY	21
Converting to and from Storage Management Subsystem (SMS) with CONVERTV	22
Managing Space with COMPRESS, DEFrag, and RELEASE	22
Using COPY for Partitioned Data Set (PDS) and Partitioned Data Set Extended (PDSE) Conversions	22
Copying DFSMSdss-Produced Dump Data with COPYDUMP	22
Printing for Diagnostic Purposes with PRINT	22
BUILDsa Command	24
BUILDsa Syntax	25
Explanation of BUILDsa Command Keywords	25
BUILDsa Command Examples	28
& CGCREATE Command	30
& CGCREATE Syntax	30
& ACCESSVOLUME	30
& FCCGVERIFY	31
COMPRESS Command	31
COMPRESS Syntax	31
Explanation of COMPRESS Command Keywords	32
Example of Compress Operations	37
CONVERTV Command	37
CONVERTV Command Syntax	38
Explanation of CONVERTV Command Keywords	38
Examples of CONVERTV Operations	41
COPY Command	42
Special Considerations for COPY	43
COPY DATASET Syntax	43
COPY FULL and COPY TRACKS Syntax	46
Explanation of COPY Command Keywords	48
Data Integrity Considerations for Full or Tracks Copy Operation	95
Examples of Full and Tracks Copy Operations	96
Examples of Data Set Copy Operations	97
ALLDATA and ALLEXCP Interactions	102
COPYDUMP Command	105
COPYDUMP Syntax	105
Explanation of COPYDUMP Command Keywords	105
Examples of COPYDUMP Operations	106
DEFrag Command	107
DEFrag Syntax	107
Explanation of DEFrag Command Keywords	108
Examples of DEFrag Operations	118
Results of a Successful DEFrag Operation	119
DUMP Command	121
Special Considerations for Dump	121
DUMP FULL and DUMP TRACKS Syntax	122
DUMP DATASET Syntax for Logical Data Set	123
DUMP DATASET Syntax for Physical Data Set	125
Explanation of DUMP Command Keywords	128
Data Integrity Considerations for Full or Tracks Dump Operation	151
Format of the Output Data Set	151

Examples of Full and Tracks Dump Operations	151
Examples of Physical Data Set Dump Operations	152
Examples of Logical Data Set Dump Operations	154
PRINT Command	157
PRINT Syntax	158
Explanation of PRINT Command Keywords	159
Examples of Print Operations	164
RELEASE Command	166
RELEASE Syntax for Physical Processing	167
RELEASE Syntax for Logical Processing	168
Explanation of RELEASE Command Keywords	169
Example of a Release Operation	176
RESTORE Command	177
Special Considerations for RESTORE	178
Data Integrity Considerations for Full or Tracks	
Restore Operations	179
RESTORE FULL and RESTORE TRACKS	
Command Syntax	179
RESTORE DATASET Command Syntax for Logical Data Set	180
RESTORE DATASET Command Syntax for Physical Data Set	184
Explanation of RESTORE Command Keywords	185
DFSMSdss RESTORE Process	216
Examples of Full and Tracks Restore Operations	220
Examples of Physical Data Set Restore Operations	221
Examples of Logical Data Set Restore Operations	223
Chapter 4. Syntax—Auxiliary Commands	229
Writing to the Operator	229
WTO Command	229
WTOR Command	229
Scheduling Tasks	230
SERIAL Command	230
PARALLEL Command	230
Controlling Task Processing	231
SET Command—Setting Condition Codes and Patch Bytes	231
IF-THEN-ELSE Command Sequence—Using Condition Codes	233
EOJ Command—Ending Your DFSMSdss Step	236
Chapter 5. DFSMSdss Stand-Alone Services	237
Preparing to Run the Stand-Alone Services Program	237
Running Stand-Alone Services in 370 Mode	237
Running Stand-Alone Services in XA or ESA Mode	238
Running Stand-Alone Services with a Predefined Console	239
Using a Tape Library	239
Using an Automatic Cartridge Loader	242
Controlling Command Sequence Processing	242
IPLing and Running the Stand-Alone Services Program	242
IPLing Stand-Alone Services	243
RESTORE—Restoring a Formatted Dump Tape	247
TAPECNTL—Rewinding and Unloading a Tape	252
Building the IPL-able Core Image	253
The BUILDFA Function	253
Understanding BUILDFA Command	
Authorization Levels	253

Chapter 6. Data Security and Authorization Checking **255**

Effects of SPECIAL, OPERATIONS, and DASDVOL	255
SPECIAL	256
OPERATIONS	256
DASDVOL	256
General Data Security Information	257
Protecting Resources and Data Sets	257
Protecting Usage of DFSMSdss	257
Password Protection	258
Protected User and Group Data Sets	259
Generic and Discrete Profile Considerations	260
Security-Level, Category, and Label Checking	262
Protect-All and Always-Call	262
Standard Naming Conventions	262
DFSMSdss Temporary Data Set Names	262
Discretely Protected Multivolume Data Set	264
Erase-on-Scratch	264
SMS-Managed Data Set Protection	264
Logging	265
DFSMSdss Storage Administrator	265
ADMINISTRATOR Keyword	265
FACILITY-Class Profiles for the ADMINISTRATOR Keyword	266
DFSMSdss Volume, Data Set and Catalog Access Authority	267
Volume Access and DASDVOL	268
Data Set Access Authorization Levels	270
Protected Catalogs	270
Non-SMS Versus SMS Authorization	271
System Operator Authorization, Special Data Set Types	271
Access Authorization for DFSMSdss Commands	271

&	CGCREATE	272
	COMPRESS	272
	CONVERTV	272
	COPY	273
	COPYDUMP	277
	DEFRAG	278
	DUMP	278
	PRINT	279
	RELEASE	279
	RESTORE	279

Appendix A. Coexistence Considerations **285**

ALLMULTI Keyword	285
Restoring Existing DFDSS Dumps using DFSMSdss	285

Appendix B. Data Integrity—Serialization **287**

Volume Serialization	287
----------------------	-----

Avoiding Lockout	288
The WAIT Option	289
Data Set Serialization	289
Enqueuing—ENQ	289
Dumping HFS Data Sets	290
Dumping zFS Data Sets	291
Dynamic Allocation (DYNALLOC)	292
Enqueuing versus Dynamic Allocation of Data Sets	292
Read/Write Serialization Scheme	293
WAIT Option	295
An Example of RESERVE-ENQUEUE Processing	296
Backup-While-Open Data Sets (CICS and DFSMStvs)	297
Backup-While-Open Status Definition	298
Backup-While-Open Processing	299
Backup-While-Open and Concurrent Copy TOLERATE (ENQFAILURE) and SHARE Considerations	300
CICS Recovery Data	301
Backup-While-Open Data Sets (IMS)	301
Appendix C. Application Programming Interface	303
Calling Block Structure	303
User Interactions	306
Cross-Memory Application Interface Overview	307
Using the Cross Memory Application Interface to Control DFSMSdss	308
System Programming Information	310
Application Interface Blocks	311
Exit Identification Block	311
Cross-Memory Application Interface Restrictions	314
User Interaction Module Exit Option Descriptions	315
Function Startup (Eioption 00)	315
Reading SYSIN Record (Eioption 01)	317
Printing SYSPRINT Record (Eioption 02)	317
Reading Physical Tape Record (Eioption 03)	317
Reading Logical Tape Record (Eioption 04)	318
Writing Logical Tape Record (Eioption 05)	318
Writing Physical Tape Record (Eioption 06)	318
Reading Disk Track (Eioption 07)	319
Writing Disk Track (Eioption 08)	319
Reading Utility SYSPRINT (Eioption 09)	319
Writing SYSPRINT Record (Eioption 10)	319
Writing WTO Message (Eioption 11)	320
Writing WTOR Message (Eioption 12)	320
Presenting ADRUFO Record (Eioption 13)	320
Function Ending (Eioption 14)	321
Presenting WTOR Response (Eioption 15)	321
OPEN/EOV Tape Volume Security and Verification Exit (Eioption 16)	321
OPEN/EOV Nonspecific Tape Volume Mount (Eioption 17)	321
Insert Logical VSAM Record During Restore (Eioption 18)	322
Output Tape I/O Error (Eioption 19)	322
Volume Notification (Eioption 20)	322
Data Set Verification (Eioption 21)	323
Bypass Verification Exit (Eioption 22)	323
Data Set Processed Notification Exit (Eioption 23)	326
Concurrent Copy Initialization Complete (Eioption 24)	328
Backspace Physical Tape Record (Eioption 25)	329
Dump Volume Output Notification (Eioption 26)	329
Avoiding Lockout	330
Application Interface Summary	330
ADREID0 Data Area	331
Constants	338
Cross Reference	339
Example: Invoking DFSMSdss by Using an Application Program	342
How to Determine DFSMSdss Version, Release, and Modification Level	342
Appendix D. Examples of the Application Program with the User Interaction Module (UIM)	345
Appendix E. Data Set Attributes	361
Appendix F. Customizing ISMF	363
ISMF Libraries That Can Be Customized	363
Restrictions to Customizing ISMF	363
Appendix G. Accessibility	365
Using assistive technologies	365
Keyboard navigation of the user interface	365
z/OS information	365
Notices	367
Programming interface information	368
Trademarks	369
Glossary	371
Index	383

About This Document

This document describes the DFSMSdss™ commands and is intended for the storage administrator and the system programmer. Its purpose is to explain the command syntax in order to perform various storage management tasks.

For information about the accessibility features of z/OS, for users who have a physical disability, see Appendix G, "Accessibility," on page 365.

Required product knowledge

To use this manual you need some knowledge of DFSMSdfp™, DFSMShsm™, Resource Access Control Facility (RACF®, a component of the Security Server for z/OS®), and job control language (JCL).

Referenced documents

The following publications are referenced in this book:

Publication Title	Order Number
<i>z/OS DFSMS Introduction</i>	SC26-7397
<i>z/OS DFSMSdss Storage Administration Guide</i>	SC35-0423
<i>z/OS DFSMS Installation Exits</i>	SC26-7396
<i>z/OS DFSMS Using the Interactive Storage Management Facility</i>	SC26-7411
<i>z/OS MVS System Messages, Vol 1 (ABA-AOM)</i>	SA22-7631
<i>z/OS DFSMSdfp Utilities</i>	SC26-7414
<i>z/OS DFSMS Using Data Sets</i>	SC26-7410
<i>z/OS MVS JCL Reference</i>	SA22-7597
<i>z/OS MVS Programming: Extended Addressability Guide</i>	SA22-7614
<i>z/OS Security Server RACF Security Administrator's Guide</i>	SA22-7683
<i>z/OS DFSMS Macro Instructions for Data Sets</i>	SC26-7408
<i>z/OS DFSMS Advanced Copy Services</i>	SC35-0428

Accessing z/OS DFSMS documents on the Internet

In addition to making softcopy documents available on CD-ROM, IBM provides access to unlicensed z/OS softcopy documents on the Internet. To view, search, and print z/OS documents, go to the z/OS Internet Library:

www.ibm.com/servers/eserver/zseries/zos/bkserv/

Using LookAt to look up message explanations

LookAt is an online facility that lets you look up explanations for most of the IBM® messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can use LookAt from the following locations to find IBM message explanations for z/OS® elements and features, z/VM®, and VSE:

- The Internet. You can access IBM message explanations directly from the LookAt Web site at <http://www.ibm.com/eserver/zseries/zos/bkserv/lookat/>.
- Your z/OS TSO/E host system. You can install code on your z/OS or z/OS.e systems to access IBM message explanations, using LookAt from a TSO/E command line (for example, TSO/E prompt, ISPF, or z/OS UNIX® System Services running OMVS).
- Your Windows® workstation. You can install code to access IBM message explanations on the *z/OS Collection* (SK3T-4269), using LookAt from a Windows DOS command line.
- Your wireless handheld device. You can use the LookAt Mobile Edition with a handheld device that has wireless access and an Internet browser (for example, Internet Explorer for Pocket PCs, Blazer, or Eudora for Palm OS, or Opera for Linux handheld devices). Link to the LookAt Mobile Edition from the LookAt Web site.

You can obtain code to install LookAt on your host system or Windows workstation from a disk on your *z/OS Collection* (SK3T-4269), or from the LookAt Web site (click **Download**, and select the platform, release, collection, and location that suit your needs). More information is available in the LOOKAT.ME files available during the download process.

Summary of Changes

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

You may notice changes in the style and structure of some content in this document—for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and irretrievability of information in our documents.

Summary of Changes for SC35-0424-06 z/OS Version 1 Release 7

This document contains information previously presented in *z/OS Version 1 Release 7 DFSMSdss Storage Administration Reference* (SC35-0424-05).

The following sections summarize the changes to that information.

New Information

This edition includes the following new information:

- New CGCREATE command with the required ACCESSVOL keyword and optional FCCGVERIFYFRZSTATE keyword is added in Chapter 3, “Syntax—Function Commands,” on page 21.
- The command syntax for the FCCGFREEZE keyword is added to the syntax diagrams for the COPY command in Chapter 3, “Syntax—Function Commands,” on page 21.
- The command syntax for the FCINCRMENTAL, FCINCREMNTALLAST, FCINCRVERIFY(REVERSE | NOREVERSE), and FCWAIT keywords is added to the syntax diagrams for the COPY command in Chapter 3, “Syntax—Function Commands,” on page 21.
- The command syntax for the FCNOCOPYTOCOPY keyword is added to the syntax diagrams for the COPY command in Chapter 3, “Syntax—Function Commands,” on page 21.
- RACF FACILITY class profile names are added to “Protecting Usage of DFSMSdss” on page 257 in Chapter 6.
- New section titled “Access Authorization for DFSMSdss Commands” on page 271 is added in Chapter 6.

Changed Information

This edition includes the following changed information:

- Changed TYPRUN=NORUN description in “How to Control DFSMSdss through PARM Information in the EXEC Statement” on page 6.

Summary of Changes for SC35-0424-05 z/OS Version 1 Release 7

This document contains information previously presented in *z/OS Version 1 Release 6 DFSMSdss Storage Administration Reference* (SC35-0424-04).

The following sections summarize the changes to that information.

New Information

This edition includes the following new information:

Changed Information

This edition includes the following changed information:

- References to ISAM data sets have been deleted from the following commands:
 - COPY
 - FORCE
 - RESTORE
 - FORCE
 - RENAME
 - RENAMEUNCONDITIONAL
- References to ISAM data sets have been removed from the table in Appendix E
- A note regarding the use of large format sequential data sets has been added to the following commands:
 - COPY
 - REPLACE
 - REPLACEUNCONDITIONAL
 - RESTORE
 - REPLACE
 - REPLACEUNCONDITIONAL

Deleted Information

This edition includes the following deleted information:

- References to STEPCAT and JOBCAT have been removed. STEPCAT and JOBCAT DD statements are no longer allowed in z/OS v1r7. Users of DFSMSdss will no longer be able to request DFSMSdss to process data sets that are cataloged in a catalog where the catalog is outside of the standard order of search.

Summary of Changes for SC35-0424-04 z/OS Version 1 Release 6

This document contains information previously presented in *z/OS Version 1 Release 5 DFSMSdss Storage Administration Reference* (SC35-0424-03).

The following sections summarize the changes to that information.

New Information

This edition includes the following new information:

- SNAPX parameter:

You can use the SNAPX parameter on:

 - JCL invocations of the Cross Memory application programming interface (API).
 - System invocations of the Cross Memory API.
- This document has been enabled for the following z/OS Library Center advanced searches:
 - Commands, at the highest level
 - Select examples

Changed Information

This edition includes the following changed information:

- FCTOPPRCPrimary keyword has been added to the following commands:
 - COPY
 - COPY DATASET
 - COPY FULL
 - COPY TRACKS
 - DEFrag

Chapter 1. Specifying DFSMSdss Commands

This chapter is divided into the following sections:

- “Command Syntax” describes syntax requirements.
- “How Many Subkeywords Are Allowed?” on page 2 explains the allowable number of subkeywords that you can specify in the input (command) stream.
- “Specifying Subkeywords in a Command Data Set” on page 2 explains how to use keywords in a command data set.
- “How to Read Syntax Diagrams” on page 3 explains the syntax conventions used in this book.
- “JCL That You Need” on page 5 summarizes the job control language (JCL) that you might need to use DFSMSdss.

Command Syntax

You can write DFSMSdss commands in free form in columns 2 through 72 (inclusive). Any character in column 1 or beyond column 72 is ignored, causing unpredictable results when the task is processed. Syntax requirements are:

Commands: A command must appear first, followed by its keywords. Each command must take up only one line, unless a continuation character is used to indicate continuation of the command on the next line. A command is separated from its keywords by one or more blanks, a comment, or both. For example:

```
DUMP FULL INDD(DASD1) OUTDD(TAPE1)
or
DUMP FULL INDD(DASD1) -
    OUTDD(TAPE1)
```

Comments: A comment is a string of characters that begins with a /* and ends with an */. For example:

```
/*THIS IS A COMMENT */
```

A comment that does not begin and end on the same line must contain a continuation character, or syntax errors will result. For example:

```
/* THIS IS A MULTI -
LINE COMMENT */
```

Separators: A separator can be a comma (,), one or more blanks, or a comment. Separators shown in the syntax diagrams in this manual are always commas, but any of the three types can be used.

Keywords: Keywords are parameters separated by one or more separators.

Subkeywords: Subkeywords follow their associated keyword and are separated from them by a pair of enclosing parentheses. One or more blanks can precede and follow each parenthesis in the pair. For example:

```
REBLOCK( DATASET1 )
or
REBLOCK(DATASET1)
```

Command Syntax

If two or more subkeywords are permissible for a single keyword, they are separated from one another by one or more blanks or by commas. Each comma can be preceded and followed by one or more blanks. For example:

```
REBLOCK(DATASET1 , DATASET2)
      or
REBLOCK(DATASET1,DATASET2)
      or
REBLOCK(DATASET1 DATASET2)
```

Continuation: Continuation of a command is specified by a hyphen (-) as the right-most nonblank character, preceded by one or more blanks. If a continuation character is used, the following line is read as if it were part of the previous line. Since only one command is allowed per line, no additional commands may be included on the continued line. If no continuation character is used, the first word on the following line must be a command. For example:

```
COPY DATASET (INCLUDE(DATASET1)) ALLDATA(*) -
          CATALOG REBLOCK(DATASET1)
```

For examples of the continuation usage, see “Continuation Rules for IF-THEN-ELSE Command Sequencing” on page 234, using the IF-THEN-ELSE command sequence.

The absence of such a hyphen indicates the end of the command. If a keyword or subkeyword cannot fit on the remainder of a line, it can be started on that line, followed immediately by a plus sign (+) in column 72, and continued on the next line.

End of a command:

The end of a command can be specified by a semicolon (;). Everything to the right of the semicolon is ignored.

How Many Subkeywords Are Allowed?

DFSMSdss allows most command keywords to have a maximum of 255 subkeywords specified within the inline command stream. Exceptions to this rule are noted within individual command descriptions.

Specifying Subkeywords in a Command Data Set

Rather than specifying the data set names in the inline (command) stream, you can instead specify the ddname of a sequential data set or a member of a partitioned data set that contains the list of data set names. This allows you to specify more than 255 data set names.

The following keywords allow this ddname specification:

DATASET	(FILTERDD(ddn))
EXCLUDE	(DDNAME(ddn))
FILTERDD	(ddn)
PASSWORD	(ddn)

How to Read Syntax Diagrams

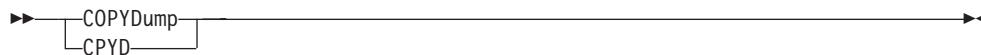
To read syntax diagrams, follow one line at a time from the beginning to the end, and code everything you encounter on that line.

The following conventions apply to all syntax diagrams for DFSMSdss commands:

- Read the syntax diagrams from left to right and top to bottom.
- Each syntax diagram begins with a double arrowhead (►►) and ends with opposing arrows (◄◄).
- An arrow (→) at the end of a line indicates that the syntax continues on the next line. A continuation line begins with an arrow (→).
- Commands and keywords are shown in uppercase and lowercase letters. The uppercase portion is the minimum needed to code the command properly; the lowercase portion is optional. For example, COPYDump can be coded in any of the following ways: COPYD, COPYDU, COPYDUM, or COPYDUMP.

Note: Commands *must* be entered in uppercase. Lowercase is not recognized.

- Some commands and keywords have alternative abbreviations; these appear as part of the stack for that command or keyword. For example, the alternative abbreviation for COPYDump is CPYD.



- Words in all lowercase letters represent information you supply. For example, *volser* or *ddn*.
- You must provide all items enclosed in parentheses, (), and you must include the parentheses.
- Where you can choose from two or more keywords, the choices are stacked one above the other. If one choice within the stack lies on the main path, you must choose a keyword. In the following example you must choose either BLK, TRK, or SOURCE.

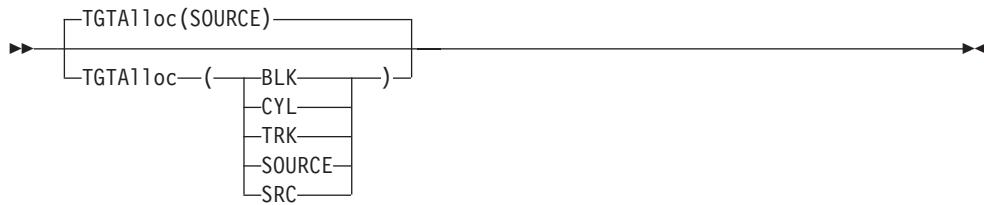


- If one or more keywords are below the main path, they are optional. In the following example CATalog and RECATalog are optional keywords. You can choose one, or the other, or none.



- If a stack of keywords are below the main path and one keyword is above the main path, the use of the keyword is optional, and the above item is the default. In the following example, if no keywords are specified, the default TGTA1loc(SOURCE) is taken.

Syntax Conventions



- The repeat symbol is shown below:



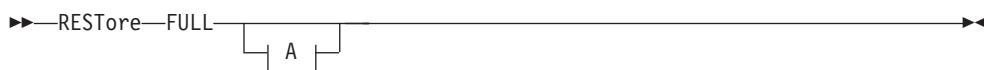
The repeat symbol appearing above keywords and variables indicates that you can specify those keywords and variables more than once. If a comma appears in the repeat symbol, you must separate repeated keywords or variables with a comma or any valid separator.

For example, after the keyword BYPASSACS, you can enter multiple data set names separated by commas.



- Substitution blocks are used to simplify the diagrams. They indicate that blocks of the syntax diagram are located outside of the main diagram. You insert the keywords for that block where the symbol appears, and return to the main diagram to continue with the command.

In the following example the substitution block, A, points to a block of the syntax diagram that immediately follows the FULL keyword.



A: Optional Keywords Used With FULL:



The above example is equivalent to the following:



JCL That You Need

The examples in this manual are shown with the necessary JCL. You may need the following JCL to use DFSMSdss:

Statement	Usage
-----------	-------

JOB (required)

Initiates your job.

EXEC (required)

Specifies the program name (PGM=ADRDSU) or, if the job control statements reside in a procedure library, the procedure name. See “How to Control DFSMSdss through PARM Information in the EXEC Statement” on page 6 for additional information that can be entered in the PARM parameter of the EXEC statement.

SYSPRINT DD (required)

Defines a sequential message data set. The data set can be written to a system output device, a tape volume, or a direct-access device. If the DCB keyword LRECL is specified on the DD statement, it must be from 84 to 137 inclusive. If the BLKSIZE keyword is specified, it must be at least four greater than LRECL. If LRECL is less than 84, the return code is 8, and an error message is issued. If the specified LRECL is greater than 137, the LRECL and BLKSIZE are set to 137 and 141, respectively.

Note: If the SYSPRINT DD or a temporary message data set resides on a volume that is being processed by DFSMSdss and a secondary allocation is needed for it, the job may result in an S138 abend. This is because the DADSM EXTEND function attempts to enqueue on the volume's VTOC while DFSMSdss holds the enqueue on that VTOC. To avoid this situation, define the SYSPRINT DD to another volume or use the WORKUNIT or WORKVOL parameters, or both.

SYSIN DD (required)

Defines a command data set containing your DFSMSdss commands. It usually resides in the input stream. However, it can be defined as a blocked or unblocked sequential data set or as a member of a partitioned data set. Records must be fixed format, LRECL=80.

input DD (optional)

Defines the input (also called the source). The ddname, *input*, is supplied by you and is referred to by your DFSMSdss commands. This DD statement is not required for some operations. Do not specify the BUFNO keyword.

The following example shows an input DD statement that specifies a DASD volume:

```
//DASD DD UNIT=3380,VOLUME=(PRIVATE,SER=111111),DISP=OLD
```

See the reference under the INDDNAME, INDYNAM, DDNAME and DYNAM keywords of the various commands (for example, COPY or DUMP) for additional information.

output DD (optional)

Defines the output (also called the target). The ddname, *output*, is

supplied by you and is referred to by your DFSMSdss commands. This DD statement is not required for some DFSMSdss operations. Do not specify the BUFNO keyword. Code a volume count when a new data set will reside on six or more volumes.

Notes:

1. DISP=MOD is not supported for an output DD statement.
2. For dump operations, the output DD statement describes a sequential data set that DFSMSdss dumps the data to. If this dump data set resides on a DASD volume that is being processed by DFSMSdss and a secondary allocation is needed for it, the job may result in a S138 abend. This is because the DADSM EXTEND function attempts to enqueue on the volume's VTOC while DFSMSdss holds the enqueue on that VTOC.
3. For dump operations, the output DD statement can be directed to DUMMY.
4. If multiple dumps are done in the same step, each dump command should have an output DD statement that refers to a unique JCL DD statement.
5. The output data set must be a standard format sequential data set and cannot use any extended-format features, such as compression.

The following example shows an output DD statement that specifies a tape volume:

```
//TAPE DD UNIT=3480,VOLUME=SER=TAPE01,LABEL=(1,SL),
// DISP=(NEW,CATLG),DSNAME=USER2.DUMP
```

See the reference under the OUTDDNAME and OUTDYNAM keywords of the various commands (for example, COPY or DUMP) for additional information.

filter DD (optional)

Defines a data set consisting of cards or card-image records that contains the filtering criteria (INCLUDE, EXCLUDE, and BY) to be used in a data set command. The ddname, *filter*, is supplied by you and is referred to by your DFSMSdss commands.

password DD (optional)

Defines a data set consisting of card-image records that contains data set names and their passwords. The ddname, *password*, is supplied by you and is referred to by your DFSMSdss commands.

For more information on coding JCL, refer to *z/OS MVS JCL Reference*.

How to Control DFSMSdss through PARM Information in the EXEC Statement

The EXEC statement for DFSMSdss can contain PARM information that is used by the program. You can use the following keyword parameters:

ABEND=nnn

nnn is a 3-digit decimal message number (ADRnnnx). If this is specified and this message is to be issued, DFSMSdss performs a user 0001 ABEND dump

after issuing the message, and the task stops. To get a dump, include a DD statement for SYSABEND, SYSMDUMP, or SYSUDUMP. This keyword is provided for diagnostic purposes only.

AMSGCNT=nnnn

The abend message occurrence count that tells DFSMSdss to end abnormally on the nth occurrence of the message specified in ABEND=nnn. **nnnn** for AMSGCNT can be a number between 1 and 9999. The default is 1 (first occurrence). This keyword is provided for diagnostic purposes only.

DEBUG=FRMSG

This parameter causes DFSMSdss to issue an informational message during copy or defrag operations for which fast replication methods, such as SnapShot and FlashCopy, cannot be used. The informational message indicates why DFSMSdss could not use fast replication. This keyword is provided for diagnostic purposes only.

Guidelines:

- When you specify this parameter, DFSMSdss interprets it as though you have specified the DEBUG(FRMSG(SUMMARIZED)) keyword.
- You can use the DEBUG(FRMSG(MIN | SUM | DTL)) keyword with the COPY and DEFrag commands. It is used when designating the use of fast replication methods. It is recommended that you convert your DEBUG=FRMSG parameter to this new keyword specification in your JCL jobs.

LINECNT=nnnn

nnnn is a 1-digit to 4-digit number indicating the number of lines to be printed per page for the SYSPRINT data set. The default is 60. Specify LINECNT=9999 to prevent a page ejection.

PAGENO=nnnn

nnnn is a 1-digit to 4-digit number indicating the starting page number to be used for the SYSPRINT data set. The default is 1.

SDUMP=nnn

nnn is a 3-digit decimal message number (ADRnnnx). If this is specified, DFSMSdss requests an SVC dump after issuing the message, and the task continues processing. The SVC dump is directed to a system defined data set (SYS1.DUMPnn) for later analysis. This keyword is provided for diagnostic purposes only.

SIZE=nnnnK

nnnn is a 1-digit to 4-digit number indicating the number of KB (1KB equals 1024 bytes) of main storage to be used by DFSMSdss. This amount must be less than or equal to that specified by the REGION keyword. The default value is the region size.

SMSGCNT=nnnn

The sdump message occurrence count that tells DFSMSdss to request an SVC DUMP on the nth occurrence of the message specified in SDUMP=nnnn. **nnnn** for SMSGCNT can be a number between 1 and 9999. The default is 1 (first occurrence). This keyword is provided for diagnostic purposes only.

TMPMSGDS=YES | NO

This parameter indicates whether or not a temporary SYSPRINT message data set is to be allocated. When NO, SYSPRINT messages are buffered in an ESA HiperSpace™ and performance is improved. The default is NO.

&
&
&
&
&
&
&
&

TRACE=YES

When used during a defragmentation operation, this prints messages that indicate the relocated extents. This is a diagnostic tool.

TYPRUN=NORUN

For copy, dump, restore, compress, and release operations, only input data set selection is done without actually processing data sets. Printed output for the run indicates the data sets selected. For a defragmentation operation, the initial volume statistics are printed without actually relocating any extents. For a CONVERTV operation, a full report is produced, but no volumes or data sets are actually converted. For CGCREATE operation, only control card syntax checking is done. The task is not processed.

TYPRUN=SCAN

Only control card syntax checking is done. No tasks are processed.

UTILMSG=YES | NO | ERROR

This parameter controls the output of messages from auxiliary programs invoked by DFSMSdss (including ICKDSF, IDCAMS, IEBCOPY, IEBISAM, and IEHMOVE) to the DFSMSdss SYSPRINT output. When YES, informational, warning, and error messages from these auxiliary programs are copied to the DFSMSdss SYSPRINT output. When NO, messages are *not* copied to the output. When ERROR, messages are copied only if the auxiliary return program returns an error code to DFSMSdss. The default is ERROR.

WORKUNIT=workunit

You can supply an esoteric DASD unit name (for example, SYSDA), a generic DASD unit name (for example, 3380), or a specific DASD address. DFSMSdss passes this unit to dynamic allocation when temporary data sets are allocated. When WORKUNIT is specified by itself, the volumes to be processed by DFSMSdss should not fall within the esoteric group passed as the WORKUNIT name. If the esoteric name applies to the volumes to be processed by DFSMSdss, specify WORKVOL.

WORKVOL=volser

You can supply a volume serial number on which DFSMSdss should allocate temporary data sets. DFSMSdss passes the volume serial number to dynamic allocation. The volume serial number passed as a WORKVOL should not be the same as any volume being processed by DFSMSdss under the current task.

Notes:

1. WORKUNIT, WORKVOL, or both parameters can be specified when invoking DFSMSdss.
2. When DFSMSdss invokes the IDCAMS utility to copy an ICF user catalog, the export data set is allocated as a permanent data set. The permanent data set cannot be placed on the volume specified by WORKUNIT or WORKVOL.
3. An esoteric unit name that requests virtual I/O can be used in the WORKUNIT parameter, but when DFSMSdss invokes the IEHMOVE utility during data set copy, the default dynamic allocation unit name is used (SYSALLDA).

XABUFF=ABOVE16 | BELOW16

The I/O buffers used by DFSMSdss for COPY, COPYDUMP, DEFrag, DUMP, PRINT, and RESTORE operations are to be above or below 16 megabytes virtual storage. The default is ABOVE16.

ZBUFF64R=YES | NO

The I/O buffers that DFSMSdss uses for COPY, COPYDUMP, DEFrag,

DUMP, PRINT, and RESTORE operations can be backed by real storage that is anywhere in 64-bit addressing, or below the 2-gigabyte bar. The ZBUFF64R EXEC parameter allows a user to indicate to DFSMSdss where the I/O buffers should be located.

DFSMSdss obtains I/O buffers that are backed by real storage anywhere in 64-bit addressing when you specify EXEC PARM='ZBUFF64R=YES'. The default is ZBUFF64R=YES, which results in all I/O buffers being obtained such that they can be backed above the 2-gigabyte bar.

When running DFSMSdss operations against storage devices that do not support 64-bit real addressing, you must tell DFSMSdss not to use I/O buffers that can be backed above the 2-gigabyte bar. This can be done by either specifying ZBUFF64R=OFF on the EXEC statement in your JCL or by turning off the UFPZB64R bit in the ADRUFO block with the Options Installation Exit Routine, ADRUIXIT.

Examples of PARM information are:

```
// EXEC PGM=ADRDSU,
//       PARM='PAGENO=8,LINECNT=57,SIZE=500K,UTILMSG=YES'

// EXEC PGM=ADRDSU,
//       PARM='TYPRUN=SCAN,DEBUG=FRMSG,XABUFF=ABOVE16,UTILMSG=YES'

// EXEC PGM=ADRDSU,
//       PARM='ZBUFF64R=NO,UTILMSG=YES'
```

Related reading: For additional information about the PARM parameter of an EXEC statement, see the *z/OS MVS JCL Reference*.

Related reading: For additional information about the ADRUFO parameter list and the ADRUIXIT exit, see the *z/OS DFSMS Installation Exits*.

Examples of Invoking DFSMSdss with JCL

The following are some examples of JCL job streams for invoking DFSMSdss.

Note: Throughout the examples in this manual, a DASD is presented as UNIT=3380 or UNIT=3390, and a tape device as UNIT=3480. This is only for illustration; you can specify any supported DASD and any supported tape device. Refer to the appropriate system generation manual for the device-type notation to be used in the UNIT parameter of a DD statement.

Moving a Data Set

The following example shows how to move a data set from one DASD volume to another using JCL and a DFSMSdss command. The source data set is deleted and the target data set is cataloged to reflect its new location.

JCL

```
//MYJOB    JOB  accounting information,REGION=nnnnK
//STEP1    EXEC  PGM=ADRDSU
//SYSPRINT DD   SYSOUT=A
//DASD1    DD   UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//DASD2    DD   UNIT=3390,VOL=(PRIVATE,SER=222222),DISP=OLD
//SYSIN    DD   *
COPY DATASET(INCLUDE(MYDATSET)) -
        LOGINDNAME(DASD1) OUTDDNAME(DASD2) DELETE CATALOG
/*
/*
```

Dumping a Data Set

Do not use a data set name that DFSMSdss will reference during execution. Otherwise, enqueue contention with the operating system initiator will occur. This example shows how to back up a DASD data set to tape:

```
//MYJOB    JOB  accounting information,REGION=nnnnK
//STEP1    EXEC  PGM=ADRDSU
//SYSPRINT DD   SYSOUT=A
//DASD     DD   UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//TAPE     DD   UNIT=3480,VOL=SER=TAPE01,
//          LABEL=(1,SL),DISP=(NEW,CATLG),DSNAME=MYDATSET.BACKUP
//SYSIN    DD   *
DUMP DATASET(INCLUDE(MYDATSET)) -
        INDDNAME(DASD) OUTDDNAME(TAPE)
/*
/*
```

Restoring a Data Set

Do not use a data set name that DFSMSdss will reference during execution. Otherwise, enqueue contention with the operating system initiator will occur. In this example, the data set (MYDATSET) that has been backed up on tape in the “Dumping a Data Set” example is restored to the original DASD volume on which it resided.

```
//MYJOB    JOB  accounting information,REGION=nnnnK
//STEP1    EXEC  PGM=ADRDSU
//SYSPRINT DD   SYSOUT=A
//TAPE     DD   UNIT=3480,VOL=SER=TAPE01,
//          LABEL=(1,SL),DISP=(OLD,KEEP),DSNAME=MYDATSET.BACKUP
//DASD     DD   UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//SYSIN    DD   *
RESTORE DATASET(INCLUDE(MYDATSET)) -
        INDDNAME(TAPE) OUTDDNAME(DASD) REPLACE
/*
/*
```

Chapter 2. Filtering—Choosing the Data Sets You Want Processed

This chapter is divided into the following sections:

- How DFSMSdss Filters Data Sets describes how DFSMSdss filters data sets by data set name and by data set characteristic.
- “Filtering by Data Set Names” on page 12 explains how to specify fully and partially qualified data set names for filtering. This section includes rules for spelling out qualifiers and examples of qualified data set names.
- “Filtering by Data Set Characteristics” on page 14 lists the data set characteristics on which filtering can be done.
- “Standard Catalog Search Order” on page 19 identifies the standard order in which DFSMSdss searches for a data set name.

See Appendix E, “Data Set Attributes,” on page 361 for information about DFSMSdss and the way that it determines individual data set attributes.

How DFSMSdss Filters Data Sets

DFSMSdss filters data sets in the following ways:

- Applying inclusion and exclusion criteria to fully or partially qualified data set names.
 - If you specify inclusion criteria (INCLUDE), DFSMSdss *tentatively* selects all data sets that fit any of the criteria.
 - If you do not specify inclusion criteria, you must specify EXCLUDE or BY. In this case, DFSMSdss tentatively selects all data sets (equivalent to INCLUDE(**)). For the DEFrag command, INCLUDE(**) is always implied.
 - If you specify exclusion criteria (EXCLUDE), DFSMSdss surveys the data sets that are selected with the inclusion criteria. It then rejects the data sets that fit any of the new criteria.
- Applying data set characteristics (BY) criteria to the data sets that are tentatively selected with the inclusion and exclusion criteria. DFSMSdss selects only those data sets that satisfy all BY criteria.

DFSMSdss lets you find out which data sets are selected by the filtering process without actually performing the requested operations. You can do this by specifying TYPRUN=NORUN on the JCL EXEC PARM field.

Virtual Storage Access Method (VSAM) Data Set Considerations

Considerations for VSAM data sets include the following:

- INCLUDE and EXCLUDE filtering is performed on the cluster names of the data sets.
- If you specify the BY criterion DSORG, MULTI, CATLG, or FSIZE, filtering is done at the cluster level. One exception is the DEFrag command, that filters at the VSAM component level. If you specify other BY criteria, the data components of the selected clusters are further filtered on those BY criteria by using information from the volume table of contents (VTOC). If a data

Filtering

component is selected, the index component for the cluster is also selected. Again, the one exception is the DEFrag command, that requires index components to pass all BY criteria.

- DFSMSdss supports the BY criterion EXPDT as it exists in the VTOC only.
- For a physical data set RESTORE, to prevent index components from being restored inadvertently, you must specify the fully qualified name of the cluster.
- For data set print operation, the fully qualified name of the data set to be printed is required. Support is at the component, not the cluster, level.

Filtering by Data Set Names

A *fully qualified* data set name is one in which all qualifiers are spelled out completely. A *partially qualified* data set name is one in which asterisks (*) or percent signs (%) are used to represent qualifiers or parts of qualifiers.

Using a * in Partially Qualified Data Set Names

The **single asterisk**, *, is used in place of exactly *one* qualifier. In addition, it can be used to indicate to DFSMSdss that only *part* of a qualifier has been specified. For example, just the first, last, middle, or first and last parts.

When used with other qualifiers, the **double asterisk**, **, indicates either the nonexistence of leading, trailing, or middle qualifiers, or the fact that they play no role in the selection process.

The rules for using asterisks in a qualifier are:

- Two asterisks are the maximum permissible in a qualifier.
- If there are two asterisks in a qualifier, they must be the first and last characters.

Consequently, the following are permissible qualifiers:

**
A

The following are **not** permissible qualifiers:

**A*
*A*B*
*A*B
A*B*C

Using a % in Partially Qualified Data Set Names

The % acts as a place holder for a single character during data set name filtering.

The rules for using % in a qualifier are:

- Each % corresponds to exactly one character.
- % can be specified more than once, consecutively or in any level of the qualifier.
- A % cannot match a null ('') or a period ('.') .
- Use of a % in filtering does not change any of the other filtering specifications for data set names.

Consequently, specifying I%M.A.%AT%%ET matches the data set names IAM.A.DATASET and IBM.A.BATTYET, but not IAM.A.DATASE, IAM.AA.DATASET, IAM.A.ATASET, or IM.A.DATASET.

Examples of Fully and Partially Qualified Data Set Names

The following are examples of fully and partially qualified data set names.

Fully Qualified Data Set Names:

USER2LD The first and only qualifier is *USER2LD*.

SYS1.UTIL3.LOAD The first qualifier is *SYS1*, the second, *UTIL3*, and the third, *LOAD*.

USER2.PROGRAM1.LIST

The first qualifier is *USER2*, the second, *PROGRAM1*, and the third, *LIST*.

Partially Qualified Data Set Names Using **:

****.LOAD** All data sets whose last, or only, qualifier is *LOAD*.

SYS1.** All data sets whose first, or only, qualifier is *SYS1*.

USER2..LIST** All data sets whose first and last qualifiers are *USER2* and *LIST*, respectively, including *USER2.LIST*.

More Partially Qualified Data Set Names:

***.LOAD** All data sets with two qualifiers whose last qualifier is *LOAD*.

SYS1.* All data sets with two qualifiers whose first qualifier is *SYS1*.

SYS1.*.LOAD All data sets with three qualifiers whose first and last qualifiers are *SYS1* and *LOAD*, respectively.

SYS1.UT*.LIST All data sets with three qualifiers whose first and last qualifiers are *SYS1* and *LIST*, respectively, and whose second qualifier begins with *UT*.

****.*LIB** All data sets whose last, or only, qualifier ends with *LIB*.

****.*LIB*** All data sets whose last, or only, qualifier has *LIB* in it.

.*PLI*. All data sets with three qualifiers whose second qualifier has *PLI* in it.

***.*.P*M** All data sets with three qualifiers whose last qualifier begins with *P* and ends with *M*.

****** All data sets.

%.LIST All data sets with one character in the first qualifier and *LIST* in the last qualifier.

USER%.* All data sets with two qualifiers whose first qualifier is *USER* followed by some other character.

*****%** All data sets whose single qualifier consists of two or more characters.

Note: Single quotation marks around a data set name indicates the name is fully qualified. For example, specifying ‘USER.%’ selects a data set whose first qualifier is *USER* and whose second qualifier is the character ‘%’. This data set is selected using the filter *USER.** because the wildcard (*) matches the character %.

Relative Generation Filtering

DFSMSdss allows filtering on relative generations of a generation data group (GDG) data set in the INCLUDE, EXCLUDE, and REBLOCK data set name lists. A GDG is specified as *dsn(n)* where *dsn* is a fully or partially qualified base name without the last qualifier (*GggggVvv*), and *n* is the relative generation number or * for all generations.

Guideline: The last qualifier consists of the generation number (*Ggggg*) and the version number (*Vvv*).

The following are examples of relative generation filtering:

- dsn(0)** For the current generation
- dsn(-x)** For the *x*th prior generation
- dsn(+x)** For the *x*th future generation
- dsn(*)** For all generations.

For logical operations using catalog filtering, you must use one of the following search criteria:

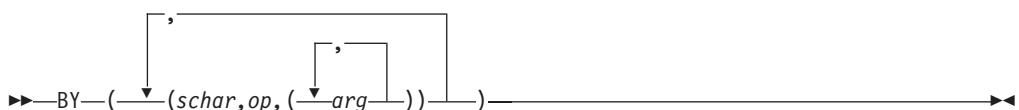
- The fully qualified data set name
- The partially qualified base name with the last qualifier (*GggggVvv*) not specified (for example, *USER1.GDG.**)
- The above relative generation filtering

Partial qualification of the last qualifier (*GggggVvv*) is not supported unless a wildcard (*) or (%) is specified in the first character of the qualifier. For example, *USER1.GDG.*V01* is valid, but *USER1.GDG.G%10** is not.

Note: Relative generation filtering is not applicable when using the CONVERTV, COPYDUMP, or PRINT functions. These functions do not allow filtering of any kind.

Filtering by Data Set Characteristics

After DFSMSdss has tentatively selected data sets by applying INCLUDE and EXCLUDE criteria, you can apply BY criteria to further restrict the data sets finally chosen. You can, for example, use BY to select data sets by creation date, storage class, and a wide variety of other criteria. The BY keyword takes this form:



Where Represents

schar The selection characteristics:

Keyword	Criteria
ALLOC	Allocation type (cylinder, track, block, absolute track, or movable)
CATLG	Whether the data set is currently cataloged or not (using the standard catalog search order)

Filtering by Data Set Characteristics

CREDIT	Creation date (absolute or relative)
DSCHA	Whether the data-set-changed flag is on or off
DSORG	Data set organization (SAM, PAM, PDS, PDSE, BDAM, HFS, EXCP, ISAM, VSAM or zFS)
EXPDT	Expiration date (absolute or relative). Data sets without expiration dates explicitly assigned to them are considered to have an expiration date of zero. If you wish to exclude these data sets from expiration date processing, you must specifically exclude them in your filtering list, that is, BY EXPDT NE 0000000.
EXTNT	Number of allocated or used extents for the entire data set on all the volumes on which it resides
FSIZE	Number of allocated or used tracks for the entire data set on all the volumes on which it resides (data set size)
MULTI	Whether the data set is singlevolume or multivolume (Single volume data sets that have been allocated but have never been opened and are not cataloged may be selected as multivolume)
REFDT	Last-referenced date (absolute or relative)
DATACLAS	Data class for SMS
MGMTCLAS	Management class for SMS
STORCLAS	Storage class for SMS
op	The operator:
	Operator Meaning
	EQ or = Equal to
	LE or <= Less than or equal to
	LT or < Less than
	GT or > Greater than
	GE or >= Greater than or equal to
	NE or != Not equal to
arg	An argument that qualifies the selection characteristic (<i>schar</i>).

Table 1 summarizes the permissible combinations of *schar*, *op*, and *arg*.

Table 1. BY Keywords

schar	op	arg	Notes
ALLOC	EQ NE	CYL (cylinder allocation) TRK (track allocation) BLK (block length allocation) ABSTR (absolute track allocation) MOV (movable data sets)	If MOV is picked, the data sets to be processed cannot be allocated as any of the following: <ul style="list-style-type: none"> • PSU (physical sequential unmovable) • POU (partitioned organization unmovable) • DAU (BDAM unmovable) • ABSTR (absolute tracks) • ISAM (indexed sequential access method). COMPRESS command: If MOV is picked, only data sets allocated as POU cannot be compressed.

Filtering by Data Set Characteristics

Table 1. BY Keywords (continued)

schar	op	arg	Notes
MULTI	EQ	YES (or 1) NO (or 0)	<p>YES: DFSMSdss processes only multivolume data sets. NO: DFSMSdss processes only singlevolume data sets. DEFrag command: If a data set's volume sequence number is greater than one in the VTOC, DFSMSdss assumes it is multivolume. If this sequence number is 1, DFSMSdss assumes it is a single volume data set. Compress command: Because DFSMSdss assumes the data set is single volume, you do not need to specify MULTI.</p> <p>Note: Single volume data sets that have been allocated but have never been opened and are not cataloged may be selected as multivolume.</p>
CATLG	EQ	YES (or 1) NO (or 0)	<p>YES: Only currently cataloged data sets are processed. NO: Only uncataloged data sets are processed. Dump command: The CATLG filter is valid only when used with an input volume list (INDD, INDY, LOGINDD, LOGINDY, STORGRP). COPY command: The CATLG filter is valid only when used with an input volume list (INDD, INDY, LOGINDD, LOGINDY, STORGRP). RESTORE command: The target data set is tested to determine if it is cataloged or not.</p> <p>Notes:</p> <ol style="list-style-type: none"> 1. Data sets cataloged through JCL are not cataloged until the job step ends. Therefore, when you use the (CATLG,EQ,NO) filter, you should also use a CREDIT or REFDT filter. The value used in the CREDIT or REFDT filter should be for data sets at least two days old: (CREDIT,LE,*,-2) or (REFDT,LE,*,-2). 2. A data set is considered uncataloged if it is not cataloged in the standard order of search or if it is cataloged in an unavailable catalog.
CREDIT EXPDT REFDT	LT GT EQ NE GE LE	<p>[yylyyddd,(n): 4-digit (or 2-digit) year and 3-digit day, modified by an optional 1-digit to 4-digit positive or negative number of days, n. The year values range from 1900 through 9999. For example, 1998100 (or 98100) is the 100th day of 1998 and 2000001 (or 00001) is the first day of 2000.</p> <p>*(n): Current date (run date), modified by an optional 1-digit to 4-digit positive or negative number of days, n. For example, *,-5 is five days before this job is run.</p> <p>NEVER: Never-expire date (valid only for EXPDT). A never-expire data set has an expiration date in its VTOC entry of 99365, 99366, or 99999.</p>	<p>The preferred form of a date has a 4-digit year (yyyyddd). When you specify a date with a 4-digit year other than a date of all zeros, the smallest valid date before modification is 1900001. When you specify a date with the 2-digit year format (yyddd), a yy of 00 through 49 indicates years 2000 through 2049; a yy of 50 through 99 indicates years 1950 through 1999. You must use the 4-digit year format if you wish to filter on years 1900 through 1949 or on years higher than 2049.</p> <p>CREDIT and EXPDT: For a multivolume data set, the date from its first volume's VTOC is used for checking.</p> <p>EXPDT: When you specify a date of 99365 or 1999365, you are specifying the last day of 1999. When you specify EXPDT with NEVER, 99366, 1999366, 99999, 1999999, or 9999999, you ask DFSMSdss to look for all valid forms of never-expire dates (including 99365 in a data set's VTOC entry). Therefore, BY(EXPDT,EQ,1999365) selects no data sets. DFSMSdss considers the valid forms of never-expire dates as equal to each other and as greater than all other dates. A never-expire date cannot be modified and cannot be specified with the GT operator.</p> <p>REFDT: For a multivolume data set, the latest date from all its VTOCs is used for checking. RELEASE and DEFrag commands: The date in the VTOC of the volume being processed is used for filtering both single- and multivolume data sets.</p>
DSCHA	EQ NE	YES (or 1) NO (or 0)	For a multivolume data set, the value used for checking is 1 if any of the indicators from all of its VTOCs is 1. Otherwise, the value is 0.

Table 1. BY Keywords (continued)

schar	op	arg	Notes
DSORG	EQ NE	SAM (all sequential data sets, including compressed and striped), PAM (partitioned data sets, including PDS and PDSE), PDS, PDSE, HFS (hierarchical file system data sets), BDAM (all direct access data sets), ISAM (all indexed sequential data sets), VSAM (includes all VSAM data set types), zFS (zSeries file system data sets) EXCP (applies to data sets not allocated as any of the listed organizations and not accessed by using any of the listed access methods)	COMPRESS command: Because DFSMSdss assumes data set organization, you do not need to specify it. Note: The selective characteristic of DSORG can be specified more than once.
DATACLAS MGMTCLAS STORCLAS	EQ NE	An appropriate SMS class name.	

Filtering by Data Set Characteristics

Table 1. BY Keywords (continued)

schar	op	arg	Notes
EXTNT FSIZE	LT GT EQ NE GE LE	nnnnnnnn (1-digit to 8-digit decimal number, from 0 to 99999999)	<p>nnnnnnnn is the number of used or allocated extents (EXTNT) or used or allocated tracks (FSIZE). RESTORE command: The data set that was dumped determines the number of used or allocated extents or tracks.</p> <p>DUMP and COPY commands:</p> <ul style="list-style-type: none"> For non-VSAM data sets: <ul style="list-style-type: none"> If ALLDATA or ALLEXCP is specified, FSIZE is equal to the allocated tracks, and EXTNT is equal to the allocated extents. If ALLDATA or ALLEXCP is not specified, FSIZE is equal to the used tracks, and EXTNT is equal to the used extents. Note: A specification of FSIZE,EQ,0 can be used to select PDSE data sets that have no members (that is, have no used directory blocks), and a specification of FSIZE,NE,0 can be used to exclude PDSE data sets that have no members. For VSAM data sets, FSIZE is always equal to the allocated tracks, and EXTNT is always equal to the allocated extents. <p>Logical data set COPY, DUMP, and RESTORE commands:</p> <ul style="list-style-type: none"> FSIZE criteria are applied once for the entire data set on all volumes that the data set resides on. For logical processing of HFS files with TYPRUN NORUN, FSIZE equals all of the allocated space. For logical processing of HFS files without TYPRUN NORUN, the used space represents the FSIZE unless ALLDATA was specified. EXTNT criteria are applied to non-VSAM data sets once for the data set on all volumes that the data set resides on. EXTNT criteria are applied to each VSAM data component on all volumes that the VSAM data component resides on. If a VSAM data component is selected by EXTNT filtering, the index component for the cluster is automatically selected. <p>Physical data set DUMP and RESTORE commands:</p> <ul style="list-style-type: none"> FSIZE criteria are applied once for each volume being processed of a non-VSAM or VSAM data set. For HFS data sets, FSIZE equates to the allocated space, not the space actually used. EXTNT criteria are applied once for each volume being processed of a non-VSAM data set or VSAM data component. If a VSAM data component residing on a volume is selected by EXTNT filtering, the index component (if any) residing on the same volume for the cluster is automatically selected. FSIZE and EXTNT can be used to select certain volumes of a multivolume data set without selecting all of the volumes that the data set resides on. <p>DEFRAG commands:</p> <ul style="list-style-type: none"> For VSAM data sets, EXTNT and FSIZE criteria are applied at the VSAM component level for the volume being processed only. EXTNT and FSIZE filtering are performed for both VSAM data and index components; VSAM index components are not automatically selected if the data component for the cluster is selected. For non-VSAM data sets, EXTNT and FSIZE criteria are applied for the volume being processed only. For HFS data sets, FSIZE equates to the allocated space, not the space actually used.
Notes:			
<ol style="list-style-type: none"> When multiple arguments are specified for an NE operation, DFSMSdss selects only those data sets not matching any of the arguments. When multiple arguments are specified for an EQ operation, DFSMSdss selects those data sets matching any of the arguments. BY criteria do not apply for the CONVERTV or COPYDUMP commands. 			

Some Examples of the BY Keywords

If you code

```
BY((ALLOC EQ CYL) (CATLG EQ YES))
```

you receive all cataloged data sets with cylinder allocation.

If you code

```
BY(FSIZE GE 100)
```

you receive all data sets whose size is greater than or equal to 100 tracks.

If you code

```
BY(DSORG EQ (PAM,SAM))
```

DFSMSdss selects all partitioned and sequential data sets.

Standard Catalog Search Order

When catalogs are searched for a data set name, the standard order of the search is:

1. DFSMSdss first searches the catalogs specified with INCAT, if any. If INCAT and ONLYINCAT are specified, the following steps are skipped.
2. DFSMSdss searches the user catalog when the data set name meets one of the following criteria:
 - The data set is a qualified name and is the name of a user catalog.
 - The data set name is the same as the alias of a user catalog.
3. DFSMSdss searches the master catalog.

Broken Data Set Considerations

Broken data sets are data sets that do not comply with defined IBM data set standards. This includes data sets for which catalog entries, VTOC entries, or VSAM volume data set (VVDS) entries are either missing or invalid. DFSMSdss may not properly select broken data sets for processing because DFSMSdss relies on the validity of these structures during filtering.

Catalog Search Order

Chapter 3. Syntax—Function Commands

&

This chapter describes *function* commands. Function commands specify operations, or “tasks,” for DFSMSdss to perform. The following function commands specify such tasks:

- BUILDSA
- CGCREATE
- COMPRESS
- CONVERTV
- COPY
- COPYDUMP
- DEFrag
- DUMP
- PRINT
- RELEASE
- RESTORE

The *z/OS DFSMSdss Storage Administration Guide* contains usage notes, including DFSMSdss restrictions. Refer to these notes as needed.

What DFSMSdss Commands Do

DFSMSdss commands can perform a variety of functions.

Building the Stand-Alone IPL-able Core Image

The DFSMSdss BUILDSA command allows you to build the Stand-Alone Services IPL-able core image. The core image is used to IPL Stand-Alone Restore.

Using DUMP and RESTORE for Backup and Recovery

DFSMSdss can be used to back up data so that it can be recovered in the event of hardware failure, application failure, or user error. DFSMSdss can back up and recover data sets, entire volumes, or specific tracks. The DFSMSdss DUMP command is used to back up tracks, volumes, and data sets, whereas the RESTORE command is used to recover them. DFSMSdss may be used for the following backup functions:

- Back up data on direct-access storage devices (DASD).
- Restore data from the backup if the original becomes lost, damaged, or inadvertently changed.
- Back up application data for vital records purposes and disaster recovery.

See Appendix A, “Coexistence Considerations,” on page 285 for coexistence issues concerning the DUMP and RESTORE commands. For information on how to back up and restore Linux for OS/390 or Linux for zSeries partitions and volumes, see the appendix titled “Linux-z/OS DFSMSdss DUMP or RESTORE HOW-TO” in the *z/OS DFSMSdss Storage Administration Guide*.

Moving Data with COPY

You must move data in order to replace storage devices, add storage capacity, and meet storage requirements. DFSMSdss allows you to perform the following data set movements:

Syntax—Function Commands

- Move data sets from old to new DASD.
- Move data sets between Storage Management Subsystem (SMS) and non-SMS-managed volumes.
- Move data sets off a volume when requiring hardware maintenance.
- Move or copy data sets for other purposes.

The DFSMSdss COPY command performs data set, volume, and track movement.

Data sets may move from one DASD volume to another volume of either like or unlike device types. Where *like* devices have the same track capacity (3390 Model 2 and 3390 Model 3), unlike devices have different track capacities (3380 Model K and 3390 Model 3).

If you copy a full volume or range of tracks, however, the DASD must be of like device type. The user must specify the source and target volumes. This process permits only one source volume and one target volume.

Converting to and from Storage Management Subsystem (SMS) with CONVERTV

SMS allows you to match the needs of users' data (like data set organization, size, and format) to the characteristics of storage devices without requiring user knowledge or understanding of the installation's hardware configuration. With SMS, users can both store and retrieve data without awareness of space limitations, device characteristics, and volume serial numbers.

The DFSMSdss CONVERTV command can convert existing volumes to and from SMS management without moving data. See Appendix A, "Coexistence Considerations," on page 285 for coexistence issues concerning the CONVERTV command.

Managing Space with COMPRESS, DEFrag, and RELEASE

DFSMSdss provides features which consolidate free space on volumes, compress partitioned data sets, and release unused data set space.

To help prevent out-of-space abends on new allocations, the DFSMSdss DEFrag command consolidates free space on a volume. The DFSMSdss COMPRESS command compresses partitioned data sets by consolidating unused space at the end of the data set. The DFSMSdss RELEASE command frees unused space for use by other data sets within sequential, partitioned, and extended-format VSAM data sets.

Using COPY for Partitioned Data Set (PDS) and Partitioned Data Set Extended (PDSE) Conversions

The DFSMSdss COPY command can move or copy a PDS, then convert the PDS to a PDSE, and vice versa.

Copying DFSMSdss-Produced Dump Data with COPYDUMP

The DFSMSdss COPYDUMP command can create a maximum of 255 copies of DFSMSdss-produced dump data.

Printing for Diagnostic Purposes with PRINT

DASD data may be printed to SYSPRINT or to a sequential data set, in print format. For data set printing, tracks are printed in a logical sequence reflecting the

data set on the volume. It does not reflect the data set within the physical cylinder or head sequence.

Syntax—Function Commands

The DFSMSdss PRINT command prints the following:

- A single-volume, non-virtual storage access method (non-VSAM) data set.
- A single-volume VSAM data set component.
- All or part of the VTOC.
- Ranges of tracks.

BUILDSA Command

Use the BUILDSA command to build the IPL-able core image for the Stand-Alone Services program. With the BUILDSA command you can specify the device (card reader, tape drive, or DASD volume) from which Stand-Alone Services will be IPLed. You can also specify the operator console to be used for Stand-Alone Services.

The BUILDSA function allocates temporary data sets if needed for BUILDSA processing. These data sets are deleted when the BUILDSA operation is complete. System-generated, temporary data set names are used.

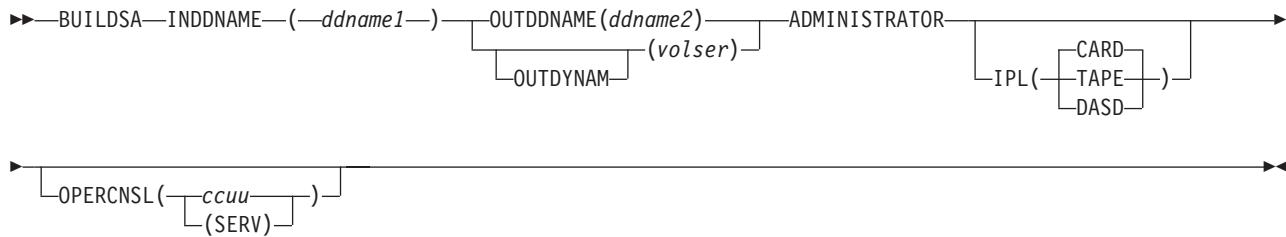
The BUILDSA function invokes the linkage editor (or Binder) utility. Specify UTILMSG=YES when you run the BUILDSA function, and keep the output as a debugging reference when you run the Stand-Alone Services program.

The BUILDSA function builds the IPL-able core image under the current operating system, and determines a record size based on whether the IPL is from card, tape, or DASD.

Notes:

1. The Stand-Alone Services modules reside in target library SYS1.SADRYLIB after they are installed and accepted by System Modification Program (SMP) or System Modification Program Extended (SMP/E). If you name this data set something other than SYS1.SADRYLIB, then substitute that name for SYS1.SADRYLIB in the examples shown later in this section.
2. Stand-Alone Services does not support the creation of the core image on an SMS-managed volume.
3. To ensure that Stand-Alone Services is available when you run from DASD, do not delete the SYS1.ADR.SAIPLD.Vvolser data set or move it to another volume.
4. If you IPL from DASD and later change the volume serial number, you must rerun the BUILDSA function to create a new core image data set with the new volume serial number in the name.
5. Consider creating a password or providing other security protection for the SYS1.ADR.SAIPLD.Vvolser data set and for the Stand-Alone Services modules.
6. If you specify TYPRUN=NORUN with the EXEC statement, the BUILDSA task ends without processing input or output.

BUILDSEA Syntax



Explanation of BUILDSEA Command Keywords

This section describes the keywords for the BUILDSEA command.

ADMINISTRATOR

```
>>—ADMINistrator—
```

ADMINISTRATOR specifies that you are a DFDSS-authorized or DFSMSdss-authorized administrator for the BUILDSEA command. If you are not authorized to use the ADMINISTRATOR parameter, the command is ended with an error message. If you are authorized, and you have access authorization to the output data set, you may create an IPL-able core image for the DASD volume. The ADMINISTRATOR parameter does not give you access to the input data sets, or to the output data sets for IPL(TAPE) or IPL(CARD).

To use the ADMINISTRATOR parameter, all of the following must be true:

- FACILITY class is active.
- Applicable FACILITY-class profile is defined.
- You have READ access to that profile.

For more details, see “Understanding BUILDSEA Command Authorization Levels” on page 253.

INDDNAME

```
>>—INDDName—(ddname1)—
```

INDDNAME specifies the DD card in the JCL that identifies the partitioned data set that contains the information that is needed to build the STAND-ALONE Services core image. This is the target library (SYS1.SADRYLIB) where the Stand-Alone Services modules are placed after the SMP or SMP/E install is complete.

ddname1 specifies the name of the DD statement that identifies a volume whose partitioned data set contains the input information.

BUILDSA Command

IPL



IPL specifies the type of device from which Stand-Alone Services will be IPLed. The appropriate loader is then created for the specified device type. If the **IPL** parameter is not specified, then the core image is created to IPL from a card reader.

CARD specifies that the core image be created for IPLing from a card reader. When **IPL(CARD)** is specified, the **OUTDDNAME** parameter must also be specified. The core image is created with **BLKSIZE=80** and **LRECL=80**, and is placed into the data set specified by the **OUTDD** parameter. This core image can be used to IPL from a card reader or tape drive. Create the output data set with **DSORG=PS**, **RECFM=F**, **BLKSIZE=80** and **LRECL=80**.

TAPE specifies that the core image be created in order to IPL from a tape drive. This core image can only be used to IPL from a tape drive. The tape is created with a blocksize determined to be the best for IPL of the Stand-Alone Services program from tape, and IBM therefore recommends that you specify **TAPE** when IPLing from a tape drive. Create the output data set with **DSORG=PS**, **RECFM=U**, **BLKSIZE=32760**, and **LRECL=32760**. When **IPL(TAPE)** is specified, the **OUTDDNAME** parameter must also be specified.

DASD specifies that the core image be created for IPLing from a DASD volume. When **IPL(DASD)** is specified, the **OUTDYNAM** parameter must also be specified. The volume specified by the **OUTDYNAM** parameter is the volume from which the Stand-Alone Services program will be IPLed. The core image is created with a record size that is determined to be the best for IPL of the Stand-Alone Services program from DASD. It is then placed in data set **SYS1.ADR.SAIPLD.Vvolser** that is allocated by Stand-Alone Services on the volume specified in the **OUTDYNAM** parameter.

Additionally, the **BUILDSA** command invokes the **ICKDSF REFORMAT** command to place the **IPLTEXT** and bootstrap records on this volume. The **BUILDSA** command provides **ICKDSF** with **IPLTEXT** necessary to IPL the Stand-Alone Services core image.

This core image can only be used to IPL from the DASD volume specified in the **OUTDYNAM** parameter.

Note: Stand-Alone Services does not support the creation of the core image from an SMS-managed volume.

OPERCNSL



OPERCNSL specifies the device address to be used as the operator console when running Stand-Alone Services.

ccuu specifies that Stand-Alone Services attempt to use the device address as the operator console instead of using the device that generates the first interrupt. Specify a valid device that exists in the configuration where the Stand-Alone Services program will be executed. You can specify a 3-digit or 4-digit address.

SERV specifies that the ES/9000® service console be used as the Stand-Alone Services operator console instead of a unit address.

When OPERCNSL is not specified, Stand-Alone Services loads a wait-state, and then waits for the operator console to generate an interrupt. Limited verification of this parameter is performed during processing of the BUILDSEA command. For more information about the predefined console, see “Running Stand-Alone Services with a Predefined Console” on page 239.

OUTDDNAME



OUTDDNAME specifies the output location for the IPL-able core image.

ddname2 specifies the DD card in the JCL where the IPL-able core image is placed when IPL(TAPE) or IPL(CARD) is specified.

When IPL(TAPE) is specified, it becomes the physical sequential data set where the core image (including bootstrap) is placed to IPL from a tape drive.

When IPL(CARD) is specified, it becomes the physical sequential data set where the core image (including bootstrap) is placed to IPL from a card reader. If this is a DASD data set, you can then punch it to cards or use it in a VM virtual card reader for IPL.

Specify either OUTDDNAME or OUTDYNAM, not both. When OUTDDNAME is specified, IPL(CARD) is the default. You can specify the IPL(CARD) parameter or the IPL(TAPE) parameter.

OUTDYNAM



OUTDYNAM specifies the output volume serial number for the DASD volume where the IPL-able core image is placed when IPL(DASD) is specified.

volser specifies the name of the DASD volume from which the Stand-Alone Services program will be IPLED.

BUILDSA Command

The IPL bootstrap and IPLTEXT (needed to read in the core image) are placed on this volume. Additionally, Stand-Alone Services allocates data set SYS1.ADR.SAIPLD.Vvolser on this volume and places the core image into this data set. If the SYS1.ADR.SAIPLD.Vvolser data set already exists, Stand-Alone Services deletes and reallocates it.

Specify either OUTDDNAME or OUTDYNAM, not both. When OUTDYNAM is specified, IPL(DASD) must also be specified.

BUILDSA Command Examples

Example 1: Core Image Using the Default Parameters

In this example, the IPL parameter is not specified, so the default (CARD) is used. Stand-Alone Services is created with BLKSIZE=80, LRECL=80, a bootstrap, and is placed in a data set on volume 339001. This core image can be used to IPL from a tape or a card reader. You can either punch the Stand-Alone Services program to a card reader or use IEBGENER to copy the Stand-Alone Services program to tape. The OPERCNSL parameter is not specified; after Stand-Alone Services is IPLED, it loads a wait-state and then waits for the first interrupt to define the operator console.

Note: The DCB parameters must be coded as shown.

```
//BUILDSA JOB accounting information,REGION= nnnnK
//STEP1 EXEC PGM=ADRDSU,PARM='UTILMSG=YES'
//SAMODS DD DSN=SYS1.SADRYLIB,DISP=SHR
//CARDDD DD DSN=ADRSA.IPLC,UNIT=3390,
//          DISP=(NEW,KEEP),VOL=SER=339001,
//          SPACE=(TRK,(40,5)),
//          DCB=(DSORG=PS,RECFM=F,BLKSIZE=80,LRECL=80)
//SYSPRINT DD SYSOUT=A
//SYSIN   DD *
      BUILDSA -
          INDD(SAMODS) -
          OUTDD(CARDDD)
/*
*/
```

The following example copies the Stand-Alone Services core image (created in Example 1) to tape.

Note: The DCB parameters must be coded as shown.

```
//COPYSA  JOB accounting information
//STEP1  EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSUT1  DD DSN=ADRSA.IPLC,DISP=SHR,VOL=SER=339001,UNIT=3390,
//          DCB=(RECFM=F,BLKSIZE=80,LRECL=80)
//SYSUT2  DD DSN=DSSSA,DISP=(,KEEP),VOL=SER=T11002,LABEL=(,NL),
//          DCB=(RECFM=F,BLKSIZE=80,LRECL=80),
//          UNIT=3480
//SYSIN   DD DUMMY
/*
*/
```

Example 2: Core Image for IPL from Tape

In this example, Stand-Alone Services is created for IPLing in stand-alone mode from a tape. The core image is then placed on an unlabeled tape. The OPERCNSL option is not specified; after Stand-Alone Services is IPLED, it loads a wait-state and then waits for the first interrupt to define the operator console.

Note: The DCB parameters must be coded as shown.

```
//BUILDSEA JOB accounting information,REGION= nnnnk
//STEP1 EXEC PGM=ADRDSU,PARM='UTILMSG=YES'
//SAMODS DD DSN=SYS1.SADRYLIB,DISP=SHR
//TAPEDD DD DSN=ADRSA.IPLT,UNIT=3480,LABEL=(,NL),
//          DISP=(NEW,KEEP),VOL=SER=TAPE01,
//          DCB=(DSORG=PS,RECFM=U,BLKSIZE=32760,LRECL=32760)
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
BUILDSEA -
    INDD(SAMODS) -
    OUTDD(TAPEDD) -
    IPL(TAPE)
/*

```

Example 3: Core Image for IPL from DASD

In this example, the core image is created for IPLing in stand-alone mode from the DASD with volume label IPLVOL. The core image, the IPL bootstrap, and IPLTEXT are all placed on volume IPLVOL in data set SYS1.ADR.SAIPLD.VIPLVOL. The OPERCNSL option is not specified; after Stand-Alone Services is IPLED, it loads a wait-state and then waits for the first interrupt to define the operator console.

```
//BUILDSEA JOB accounting information,REGION= nnnnk
//STEP1 EXEC PGM=ADRDSU,PARM='UTILMSG=YES'
//SYSPRINT DD SYSOUT=A
//SAMODS DD DSN=SYS1.SADRYLIB,DISP=SHR
//SYSIN DD *
BUILDSEA -
    INDD(SAMODS) -
    OUTDYNAM(IPLVOL) -
    IPL(DASD)
/*

```

Example 4: Core Image for IPL from DASD with OPERCNSL Option

In this example, the core image is created for IPLing in stand-alone mode from the DASD with volume label IPLVOL. The core image, the IPL bootstrap, and IPLTEXT are all placed on volume IPLVOL in data set SYS1.ADR.SAIPLD.VIPLVOL. The OPERCNSL customization option is specified for the operator console definition; after Stand-Alone Services is IPLED, it attempts to use the device at address 0009 as the operator console instead of waiting for the first interrupt.

```
//BUILDSEA JOB accounting information,REGION= nnnnk
//STEP1 EXEC PGM=ADRDSU,PARM='UTILMSG=YES'
//SYSPRINT DD SYSOUT=A
//SAMODS DD DSN=SYS1.SADRYLIB,DISP=SHR
//SYSIN DD *
BUILDSEA -
    INDD(SAMODS) -
    OUTDYNAM(IPLVOL) -
    IPL(DASD) -
    OPERCNSL(0009)
/*

```

CGCREATE Command

& CGCREATE Command

&
&
&
&
&
The CGCREATE command signals that a FlashCopy Consistency Group has been formed or aborted. I/O activity can resume on the "frozen" FlashCopy source volumes previously established by specifying FCCGFREEZE on the COPY command.

&
&
&
&
&
The CGCREATE operation is processed at logical subsystem (LSS) level. It thaws all the volumes currently in "frozen for consistency grouping" state in the LSS that received the command. When a "thaw" command is received by an LSS that does not have any frozen volumes in a FlashCopy Consistency Group, the command is accepted, but no actual processing takes place.

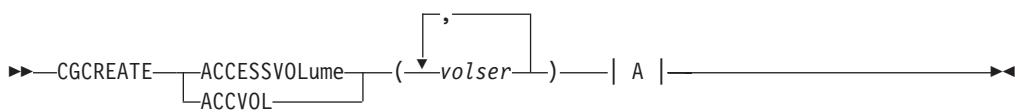
&
&
&
The freeze and thaw operations require the specified devices support the FlashCopy Consistency Group function. Appropriate LIC level is required on the ESS Model 800, the DS8000, or the DS6000.

&
&
The CGCREATE command is restricted by RACF FACILITY-Class profile 'STGADMIN.ADR.CGCREATE'.

& **Related reading:**

- For additional information about creating consistent copies with FlashCopy Consistency Group, see *z/OS DFSMSdss Storage Administration Guide*.
- For additional information about FlashCopy Consistency Group, see *z/OS DFSMS Advanced Copy Services* and the IBM TotalStorage ESS User's Guide.
- For additional information about RACF authorization, see *z/OS DFSMSdss Storage Administration Guide*.
- For additional information about RACF FACILITY class profiles, see *z/OS Security Server RACF Security Administrator's Guide*.

& CGCREATE Syntax



& **A: Optional keywords::**



& ACCESSVOLUME

& *volser* Specifies the volume serial number of a CKD volume.

&
&
&
&
&
ACCESSVOLUME specifies one or more CKD access volumes to be used for a z/OS host to communicate with the storage facility and to direct the "Consistency Group Created" command to the logical subsystems where the access volumes reside. The volumes must be mounted and online. You can specify up to 255 volumes. You cannot specify a nonspecific volume serial number using an asterisk (*).

& ACCESSVOLUME is a required keyword on the CGCREATE command. Because
 & the CGCREATE command is processed at LSS level, only one volume needs to be
 & specified for each LSS containing frozen volumes in the FlashCopy Consistency
 & Group.

& If the specified access volume does not reside in an LSS with Consistency Group
 & timer enabled or if the storage facility does not support FlashCopy Consistency
 & Group, the CGCREATE command will fail for the volume.

& FCCGVERIFY

& *volser* Specifies the volume serial number of the FlashCopy Consistency Group
 & verification volume.

& FCCGVERIFYFRZSTATE specifies the volume serial number of the verification
 & volume that DFSMSdss will use to validate the state of the FlashCopy Consistency
 & Group before thawing all the volumes. This will help you determine if the copies
 & of the group of volumes are consistent. An error message is issued if the frozen
 & state cannot be verified. Regardless of the verification result, DFSMSdss will
 & proceed to thaw all the volumes in the designated logical subsystems.

& For the verification volume, IBM recommends that you select the first source
 & volume that was copied with FCCGFREEZE in the group. When the logical
 & subsystems have different Consistency Group timer values, select the volume
 & residing in the LSS with the smallest Consistency Group timer value.

COMPRESS Command

The COMPRESS command compresses partitioned data sets on a specified volume. Compressing (degassing) removes unused space between members in a partitioned data set. Depending on the filtering criteria that you specify, you can compress either all or some of the partitioned data sets. This command is useful for compressing system partitioned data sets before you apply maintenance (to avoid certain space-related abends).

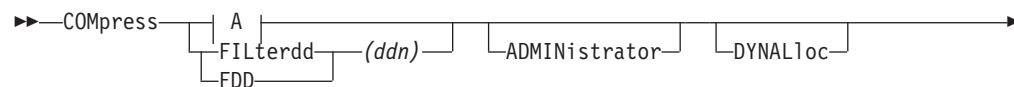
Restriction: You must not compress data sets that contain DFSMSdss or IEBCOPY executable code.

The actual PDS compression is done on the existing volume using the IEBCOPY utility. To prevent loss of data if the system or IEBCOPY abnormally ends during the processing, back up volumes or data sets that meet the filtering criteria before you use the COMPRESS command.

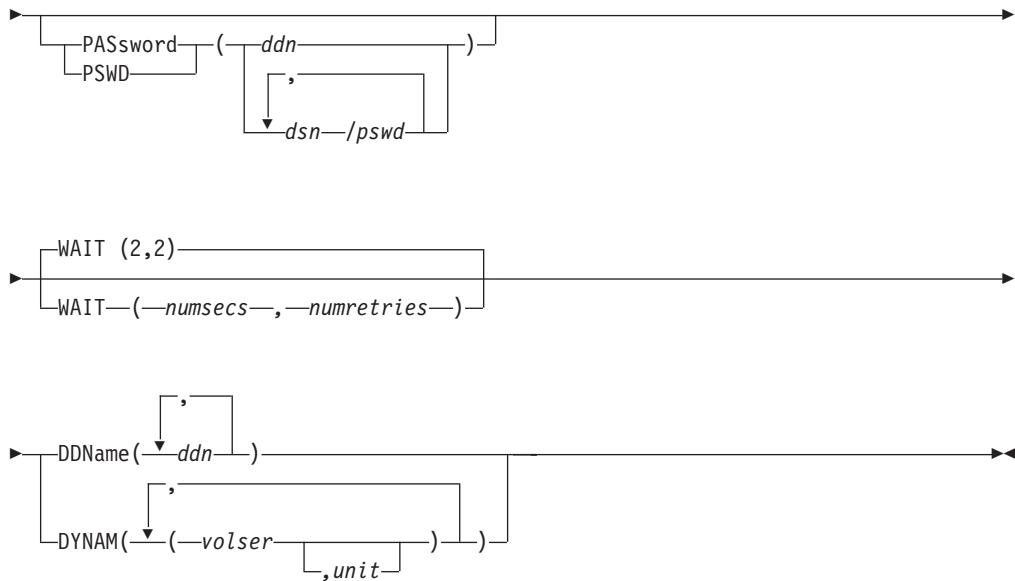
The COMPRESS command cannot process partitioned data sets that:

- Are unmovable
- Have no directory

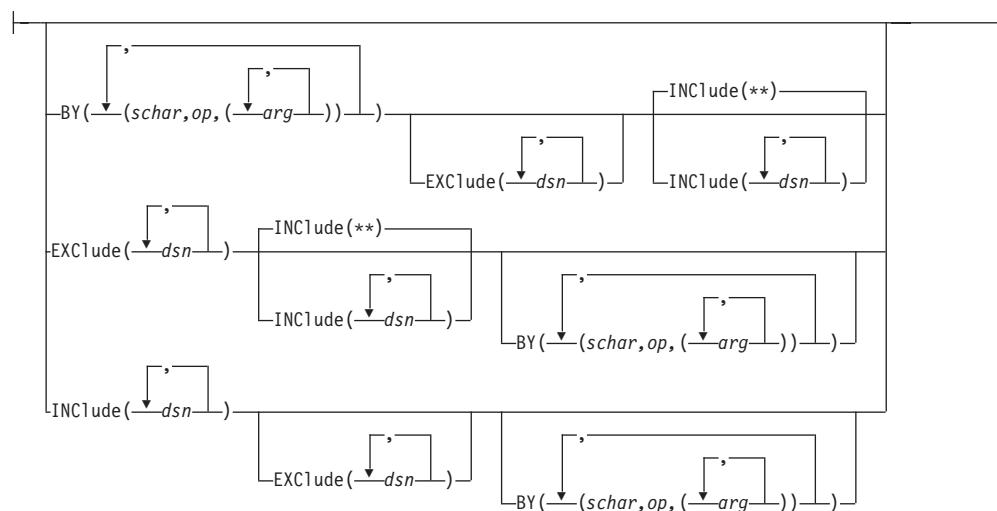
COMPRESS Syntax



COMPRESS Command



A: Additional Keywords with the COMPRESS command:



Explanation of COMPRESS Command Keywords

This section describes the keywords for the COMPRESS command.

ADMINISTRATOR



ADMINISTRATOR lets you act as a DFSMSdss-authorized storage administrator for the COMPRESS command. If you are not authorized to use the **ADMINISTRATOR** keyword, the command is ended with an error message. Otherwise, access checking to data sets and catalogs is bypassed.

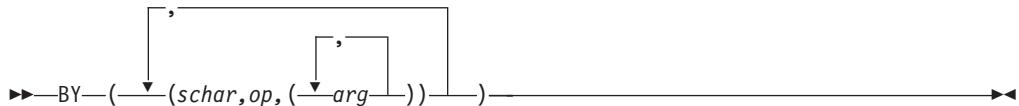
To use the **ADMINISTRATOR** keyword all of the following must be true:

- FACILITY class is active.

- The applicable FACILITY-class profile is defined.
- You have READ access to that profile.

For more details, see “ADMINISTRATOR Keyword” on page 265.

BY



BY specifies that the data sets selected up to this point, by the processing of the INCLUDE and EXCLUDE keywords, are to be further filtered. To select the data set, *all* BY criteria must be met. See “Filtering by Data Set Characteristics” on page 14 for a full discussion of *schar*, *op*, and *arg*. See the separate discussions of the INCLUDE and EXCLUDE keywords for information on how these keywords are specified.

Rule: You must use FILTERDD when you have more than 255 entries in the INCLUDE, EXCLUDE, or BY list keywords.

DDNAME



ddn Specifies the name of the DD statement that identifies a volume whose partitioned data sets, if selected, are to be compressed. To assure correct processing, each of the DD statements corresponding to a DDNAME (*ddn*) must identify only one volume serial number.

Related reading: For additional information about storage requirements when processing multiple volumes, see the *z/OS DFSMSdss Storage Administration Guide*.

DYNALLOC



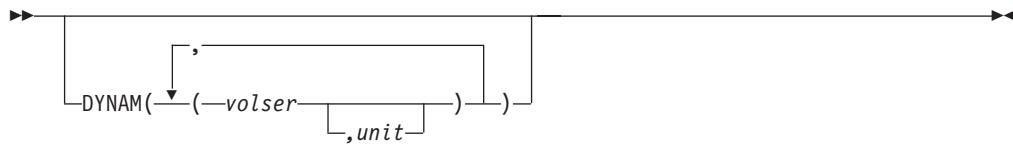
DYNALLOC specifies dynamic allocation, instead of enqueue, to serialize the use of selected partitioned data sets. This allows cross-system serialization in a JES3/MVS environment.

Consider:

- The serialization is of value only when the dynamic allocation/JES3 interface is not disabled.
- Run time increases when DYNALLOC is used to serialize data sets (as opposed to enqueue) because overhead is involved in dynamic allocation and serialization across multiple processors.

COMPRESS Command

DYNAM



DYNAM specifies a dynamically allocated volume whose partitioned data sets, if selected, are to be compressed. The volume must be mounted and online. You cannot specify a nonspecific volume serial number using an asterisk (*). Consider using DYNAM instead of DD statements to allocate DASD volumes. This does not appreciably increase run time and permits easier coding of JCL and command input.

volser Specifies the volume serial number of a DASD volume to be processed.

unit Specifies the device type of a DASD volume to be processed. This parameter is optional.

Related reading: For additional information regarding storage requirements when processing multiple volumes, see the *z/OS DFSMSdss Storage Administration Guide*.

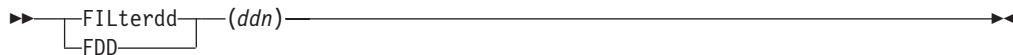
EXCLUDE



dsn Specifies the name of a data set to be excluded from the data sets selected by the INCLUDE keyword. Either a fully or a partially qualified data set name can be used. See the separate discussions of the INCLUDE and BY keywords for information on how they are specified.

Rule: You must use FILTERDD when you have more than 255 entries in the INCLUDE, EXCLUDE, or BY list keywords.

FILTERDD



ddn Specifies the name of the DD statement that identifies the sequential data set or member of a partitioned data set that contains the filtering criteria to use. This is in the form of card-image records, in DFSMSdss command syntax, that contain the INCLUDE, EXCLUDE, and BY keywords that complete the syntax of the COMPRESS command.

Rule: You must use FILTERDD when you have more than 255 entries in the INCLUDE, EXCLUDE, or BY list keywords.

INCLUDE

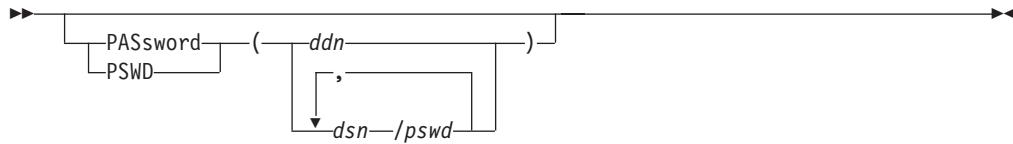
```
►►—INClude—(—dsn—, —dss—)————►►
```

dsn Specifies the name of a data set eligible to be compressed. Either a fully or a partially qualified data set name can be used. See “Filtering by Data Set Names” on page 12. If INCLUDE is omitted (but EXCLUDE or BY is specified) or if INCLUDE(**) is specified, *all* partitioned data sets are eligible to be selected for compressing. See the separate discussions of EXCLUDE or BY for information on how these keywords are specified.

Rule: You must use FILTERDD when you have more than 255 entries in the INCLUDE, EXCLUDE, or BY list keywords.

COMPRESS Command

PASSWORD



PASSWORD specifies the passwords DFSMSdss uses for selected password-protected data sets. (Password checking is bypassed for data sets that are protected by the resource access control facility (RACF).) This must be specified only if:

- You do not have the required RACF DASDVOL or RACF DATASET access.
- The installation authorization exit does not bypass the checks.

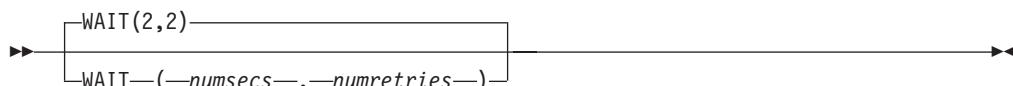
Note: You should specify the passwords for all data sets that do not have RACF protection but do have password protection. During processing, a utility invoked by DFSMSdss may have to prompt the operator for a password. You can control authorization checking by using the installation authorization exit.

ddn Specifies the name of the DD statement that identifies the sequential data set or member of a partitioned data set that contains data set names and their passwords. This data set must contain card-image records in DFSMSdss command syntax format.

dsn/pswd *dsn* is a fully qualified data set name. *pswd* is its password. If no password follows the slash (/), *dsn* is treated as though it were *ddn*.

Printing of actual data set passwords specified in your input command stream is suppressed in the SYSPRINT output.

WAIT



WAIT specifies to DFSMSdss the length of a wait in seconds and the number of retries to obtain control of a data set.

numsecs Is a decimal number (0–255) that specifies the interval, in seconds, between retries.

numretries Is a decimal number (0–99) that specifies the number of times an attempt to gain control of a data set can be retried.

The default for *numsecs,numretries* is WAIT(2,2), which specifies two retries at 2-second intervals. If you do not want to wait for a data set, specify 0 for either *numsecs* or *numretries*.

Note: The WAIT keyword does not control wait/retry attempts for system resources (such as the VTOC and the VVDS). For system resources, the default wait time is 3 seconds and the default retry count is 30. This results in a total wait time of 90 seconds.

Related reading: For information about controlling the wait/retry attempts for system resources, see the *z/OS DFSMSdss Storage Administration Guide*.

Example of Compress Operations

The following example compresses a selected partitioned data set.

```
//JOB1      JOB accounting information,REGION=nnnnK
//STEP1     EXEC PGM=ADRDSU
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
      COMPRESS      -
      DYNAM(338000)    /* DYNAM ALLOC VOL 338000      */ -
      EXCLUDE(SYS1.**)  /* EXCL 'SYS1....' DATA SETS      */ -
                        /* IF THEY MEET THIS CRITERION */ -
      BY((DSCHA EQ 0)) /* DATA SET WAS BACKED UP      */ /
/*
```

Compress partitioned data sets on volume 338000 if:

- They are not system data sets (EXCLUDE(SYS1.**)), and
- They have not been updated (DSCHA EQ 0) since the last time they were backed up (dumped). This ensures that the data set can be recovered if the system fails while the compress operation is running.

CONVERTV Command

The CONVERTV command is used to convert existing volumes to and from SMS management without data movement. The CONVERTV command performs three functions:

- Locks volumes that are ready for conversion to prevent new data set allocations (PREPARE keyword).
- Examines volumes identified by SMS to determine if they can be converted to SMS management (TEST keyword). No conversion is actually performed, but DFSMSdss identifies any data sets that cannot be converted to SMS management and why they cannot be converted.
- Performs conversion of volumes into or out of SMS management. Any conditions that prevent conversion are identified.

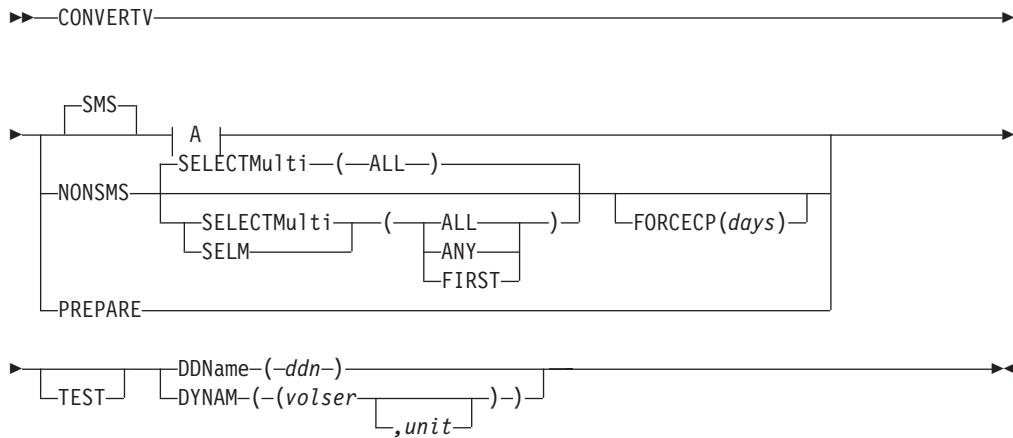
Guideline: Proper RACF security authorization might be required.

Related reading: For additional information about RACF security authorization, see the *z/OS DFSMSdss Storage Administration Guide*.

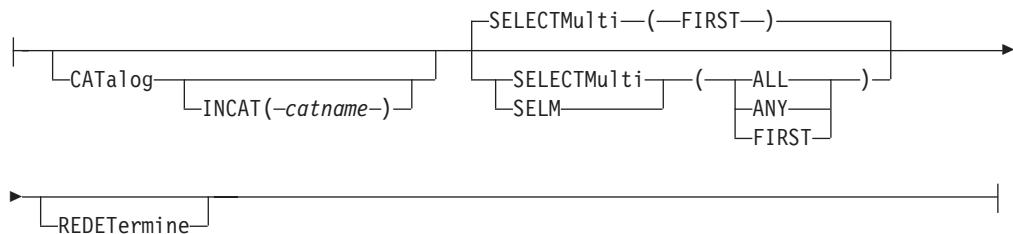
CONVERTV Command

CONVERTV Command Syntax

The syntax of the CONVERTV command is:



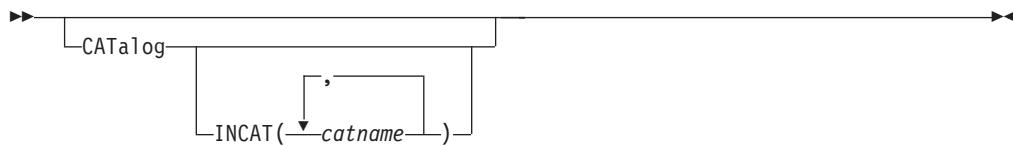
A: The syntax of optional keywords with CONVERTV SMS is::



Explanation of CONVERTV Command Keywords

This section describes the keywords for the CONVERTV command.

CATALOG



CATALOG specifies that if a data set's catalog entry is not found in the standard order of search, the data set is to be cataloged during the conversion.

If the CATALOG keyword has not been specified, and a data set's catalog entry is not found in the standard order of search, the data set is not converted.

INCAT Specifies input catalogs that are not in the standard search order. This allows non-VSAM data sets cataloged outside the standard order of search to be processed.

catname Specifies a fully qualified catalog name.

If CATALOG is specified without INCAT, a single volume, non-VSAM data set cataloged outside the standard order of search might be cataloged in more than one place.

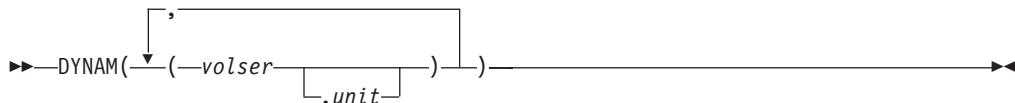
DDNAME



DDNAME specifies a volume that you want converted. Use this keyword to designate the list of volumes that need conversion. Use this keyword when you do not use the DYNAM keyword.

ddn Specifies the name of the DD statement that identifies a volume to be processed. Up to 255 DDNAMES can be specified.

DYNAM



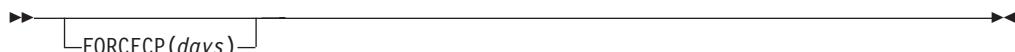
DYNAM specifies the volume that you want to process must be dynamically allocated. The volume must be mounted and online. You cannot specify a nonspecific volume serial number by using an asterisk (*). Use this keyword to designate the list of volumes that you need to convert. Use the DYNAM keyword when you do not specify the DDNAME keyword.

Consider using DYNAM instead of DD statements to allocate DASD volumes. This does not noticeably increase run time and presents easier coding of JCL and command input.

volser Specifies the volume serial number of a DASD volume to be processed. Up to 255 volumes can be specified.

unit Specifies the device type of a DASD volume to be processed. This parameter is optional.

FORCECP



FORCECP specifies that checkpointed data sets resident on the SMS volume can be converted to non-SMS management. Checkpoint indications are removed from the data set during conversion.

days Specifies the number of days that must have elapsed since the last referenced date before the data set can be converted. It is a one-to-three-digit number in the range of zero to 255.

INCAT

See the CATALOG keyword.

CONVERTV Command

NONSMS

►►NONSMS————►►

NONSMS specifies that a volume and all of the data sets on that volume be converted from SMS management to non-SMS management.

PREPARE

►►PREPare————►►

PREPARE specifies that a volume is to be prepared for SMS without conversion of data sets. This prevents the volume from changing prior to performing the full SMS conversion. After the PREPARE is requested, the volume is placed in initial status and you cannot allocate new data sets. However, you can delete existing data sets.

The NONSMS keyword must be specified to return the volume to non-SMS management.

REDETERMINE

►►—REDETermine————►►

REDETERMINE specifies that the SMS class information is to be reset for data sets previously converted to SMS management whose SMS management class or SMS storage class do not match those returned by the current ACS routines.

REDETERMINE allows management class and storage class to be reset, but does not update the data class.

If REDETERMINE is used with the TEST keyword, a report is produced specifying all data sets eligible for conversion, including those already converted.

SELECTMULTI

►►—SELECTMulti—(—ALL—)————►►

►►—SELECTMulti—(—
SELM
—ALL
ANY
FIRST—)————►►

SELECTMULTI specifies how cataloged multivolume data sets are to be selected during conversion to or from SMS management. The volume list is the list of volumes supplied by the DDNAME or DYNAM keyword.

ALL Specifies that DFSMSdss *not process* a multivolume data set unless all of the volumes that contain a part of the non-VSAM data set or VSAM base cluster are in the volume list specified by DDNAME or DYNAM. ALL is the default for non-SMS processing.

ANY Specifies that DFSMSdss process a multivolume data set when any part of

the non-VSAM data set or VSAM base cluster is on a volume in the volume list specified by DDNAME or DYNAM.

FIRST Specifies that DFSMSdss process a multivolume data set only when the DDNAME or DYNAM volume list includes the volume that contains the first part of the non-VSAM data set or the primary data component of the base cluster for a VSAM sphere. FIRST is the default for SMS processing.

SMS



SMS specifies that a volume and all of the data sets on that volume are to be converted to SMS management. SMS is the default when the SMS, NONSMS, or PREPARE keyword is not specified.

TEST



TEST specifies that DFSMSdss is to verify that a volume and its data sets are eligible for conversion or for preparation. The TEST keyword functions just as if TYPRUN=NORUN had been specified on the JCL EXEC PARM field. DFSMS must be active to use this function.

Note: It is also possible to use TEST to verify that the ACS algorithms would process correctly because the resulting report indicates the classes associated with the various data sets on the volume.

Examples of CONVERTV Operations

The following are examples of the CONVERTV command.

Example 1: Using the CONVERTV Command to Simulate Conversion

```

//JOB1     JOB   accounting information,REGION=nnnnK
//STEP1    EXEC  PGM=ADRDSU
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
              

CONVERTV SMS -
           DYNAM((VOL001,3380),(VOL002,3380),(VOL003)) -
           TEST
/*
  
```

The preceding example uses the TEST keyword to simulate conversion. The TEST keyword produces a report that indicates whether the three volumes (VOL001, VOL002, and VOL003) can be converted to SMS management.

CONVERTV Command

Example 2: Using the CONVERTV Command to Convert to SMS

```
//JOB1      JOB  accounting information,REGION=nnnnK
//STEP1     EXEC PGM=ADRDSU
//SYSPRINT DD   SYSOUT=*
//DVOL1     DD   UNIT=SYSDA,VOL=SER=338001,DISP=OLD
//DVOL2     DD   UNIT=SYSDA,VOL=SER=338002,DISP=OLD
//SYSIN     DD   *
               *
               CONVERTV -
               DDNAME(DVOL1,DVOL2) -
               SMS -
               INCAT(SYS1.ICFCAT.V338002) -
               SELECTMULTI(FIRST) -
               CATALOG
/*
               */
```

The non-SMS-managed volume 338002 and the SMS-managed volume (in INITIAL state) 338001 are converted to SMS. The volume 338001 has been placed in the initial state by the storage administrator. Regardless of where the data sets reside, all multivolume data sets whose first extent is on volume 338001 or 338002 are processed. In addition, there are some data sets on volume 338002 cataloged in the user catalog SYS1.ICFCAT.V338002. These data sets are uncataloged from the user catalog and cataloged in the standard order of search. The INCAT keyword provides access to the user catalog.

Example 3: Using the CONVERTV Command to Convert from SMS

```
//JOB1      JOB  accounting information,nnnnK
//STEP1     EXEC PGM=ADRDSU
//SYSPRINT DD   SYSOUT=*
//SYSIN     DD   *
               *
               CONVERTV -
               DYNAM(338003) -
               NONSMS
/*
               */
```

This example converts a volume to non-SMS-managed.

COPY Command

The DFSMSdss COPY command performs data set movement, volume movement, and track movement from one DASD volume to another.

You can copy data sets to another volume of either like or unlike device types. Like devices have the same track capacity (3390 Model 2 and 3390 Model 3), while unlike devices have different track capacities (3380 Model K and 3390 Model 3).

However, the DASD must be of *like* device type if you copy a full volume or range of tracks. The user must specify the source volumes and the target volumes. DFSMSdss only allows one source volume and one target volume.

DFSMSdss offers two ways to process COPY commands as follows:

- *Logical processing* is data set-oriented, which means that it operates against data sets and volumes independently of physical device format.

- *Physical processing* operates against volumes and tracks, but moves data at the track-image level.

Integrated catalog facility catalogs should not have a high-level qualifier of SYSCTLG because this causes DFSMSdss to treat them as control volumes.

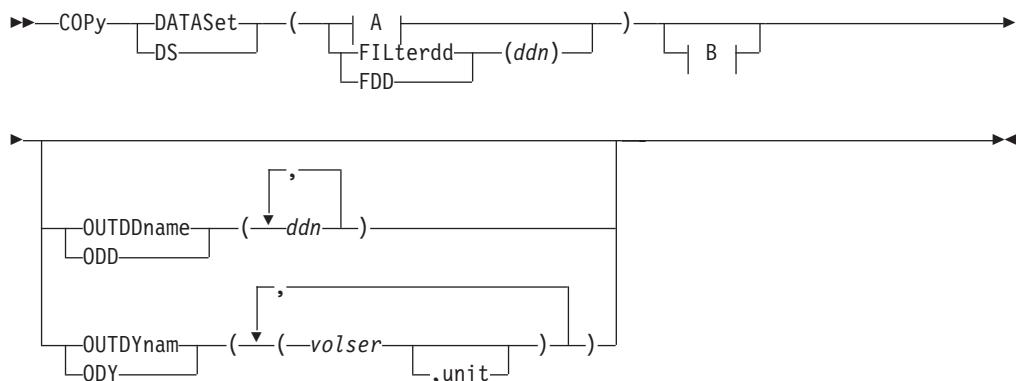
Related reading: For additional information about how to use the COPY command, see the *z/OS DFSMSdss Storage Administration Guide*.

Special Considerations for COPY

The following special considerations may apply when you perform a COPY operation:

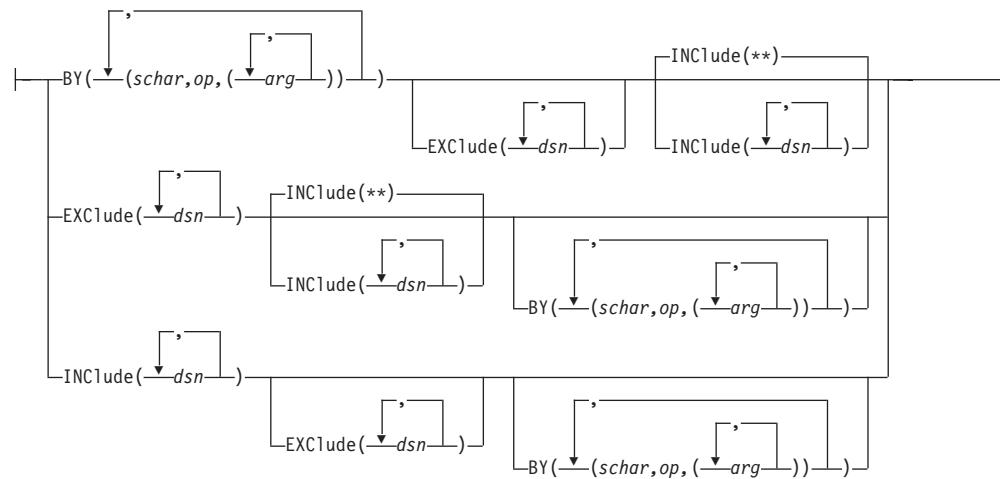
- The logical data set COPY function supports hierarchical file system (HFS) data sets and zSeries file system (zFS) data sets. There is no support for copying individual files within an HFS or zFS.
- The COPY function is not supported for SAM compressed extended-format data sets being copied to a non-SMS-managed target.
- When you perform a logical COPY operation of a VSAM compressed data set, the target data set allocation must be consistent with the source data set allocation as follows:
 - If the source is an extended-format VSAM KSDS, then the target must be an extended-format VSAM KSDS.
 - If the source is a compressed VSAM KSDS, then the target must be a compressed VSAM KSDS.
 - If the source is an alternate index for an extended-format KSDS, then the target must be an alternate index for an extended-format KSDS.
 - The target control interval size must be equal to the source.

COPY DATASET Syntax

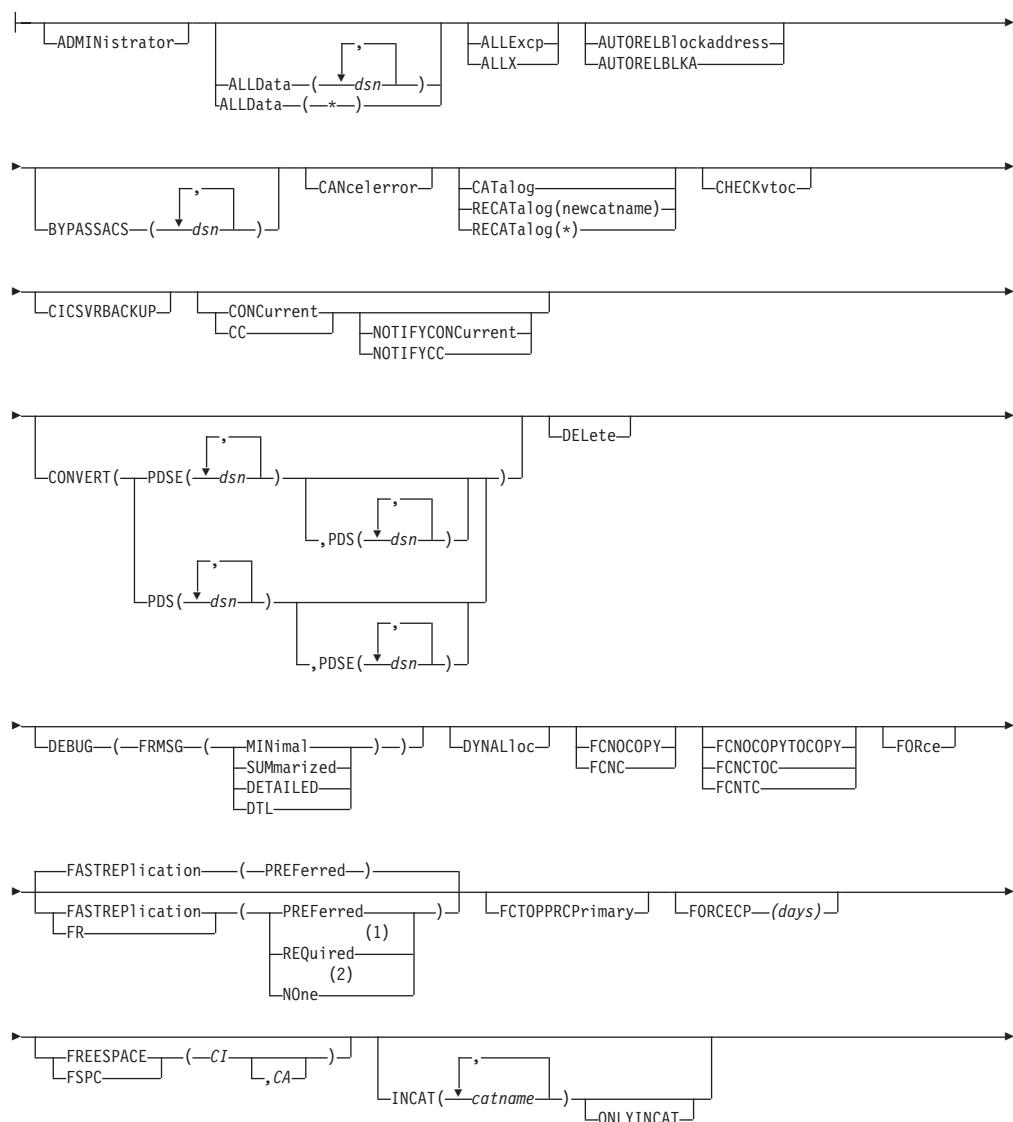


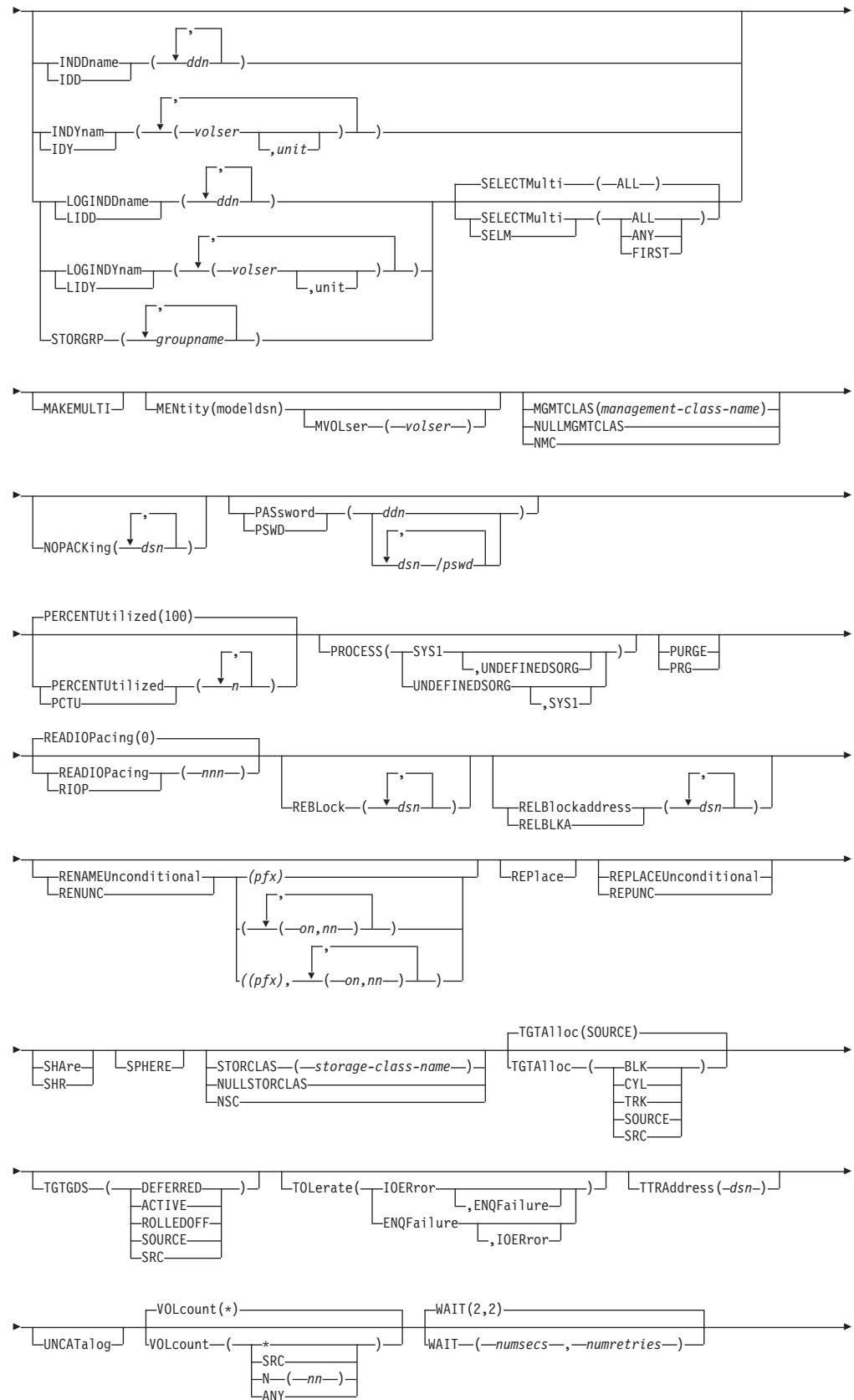
A: Additional Keywords with COPY DATASET:

COPY Command



B: Optional Keywords with COPY DATASET:





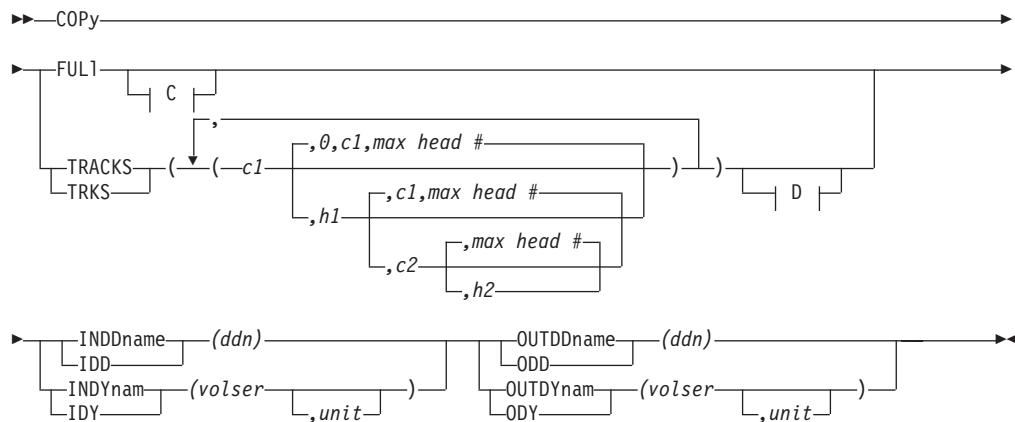
COPY Command



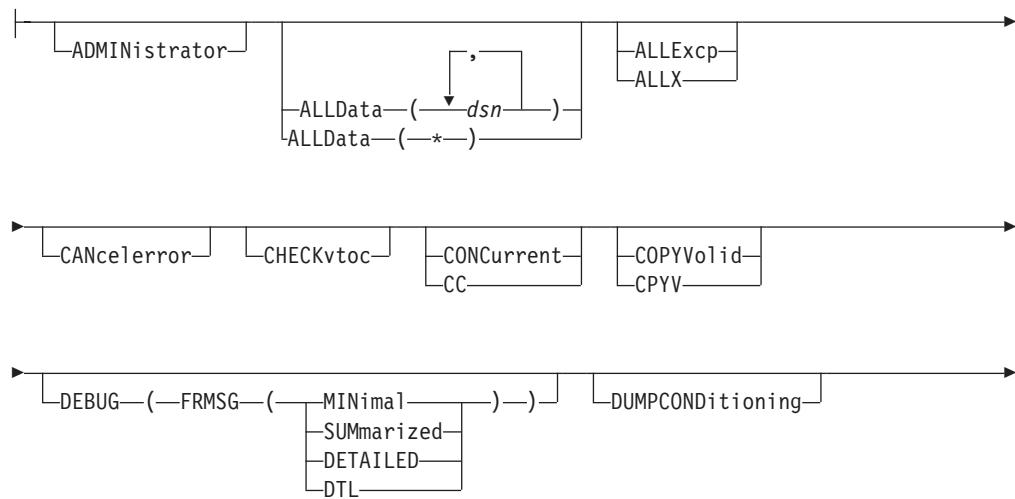
Notes:

- 1 Do not use the FASTREPLICATION (REquired) keyword with the CONCURRENT keyword.
- 2 Do not use the FASTREPLICATION (NONE) keyword with the FCNOCOPY or FCTOPPRCPrimary keywords.

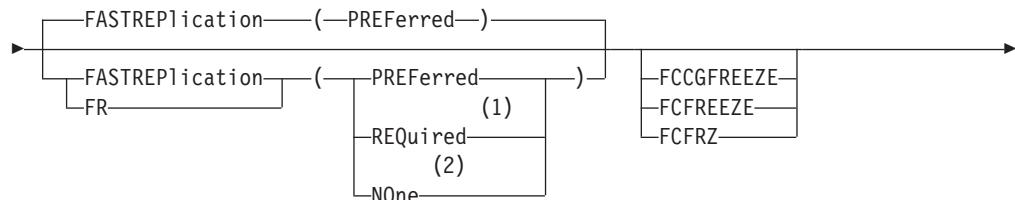
COPY FULL and COPY TRACKS Syntax

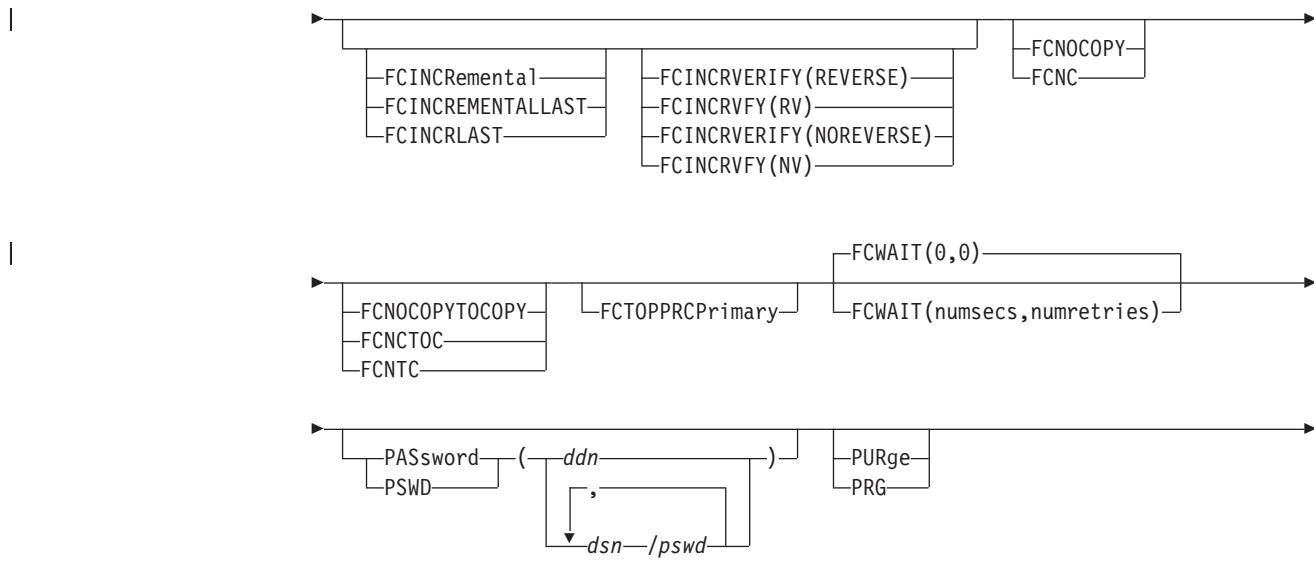
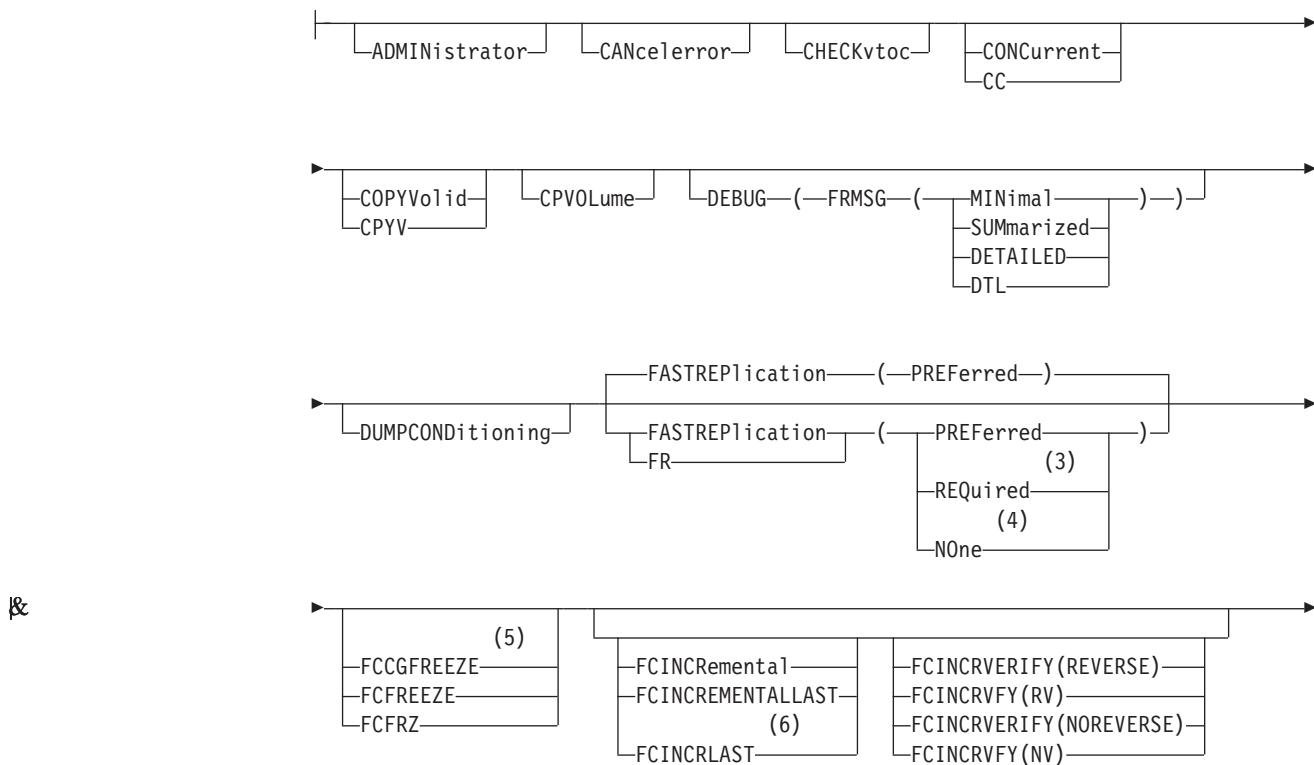


C: Optional Keywords with COPY FULL:

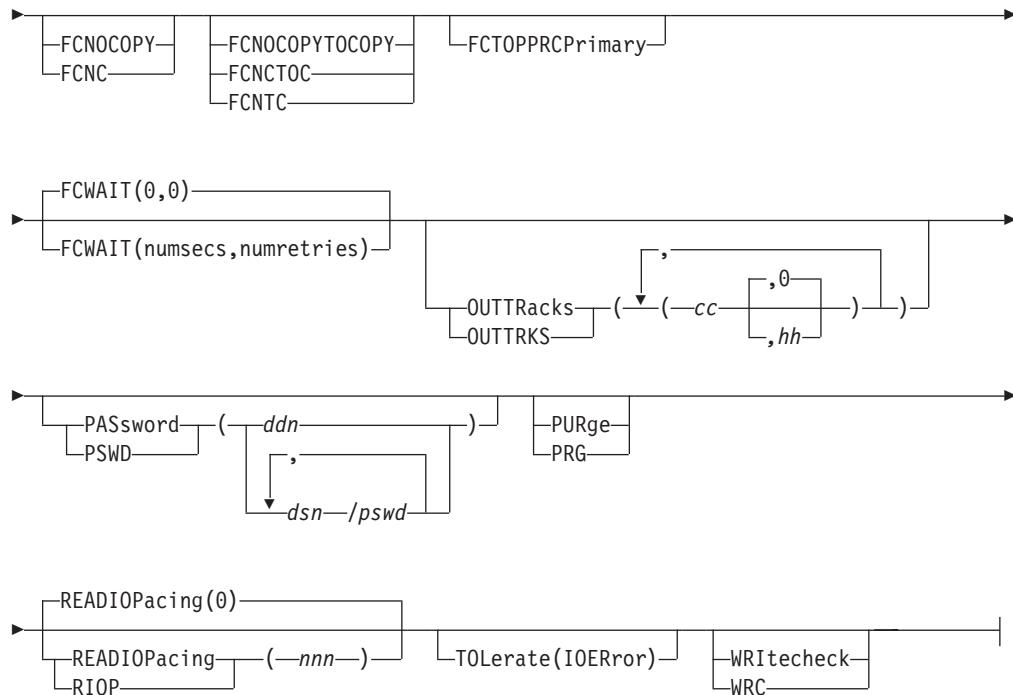


&



**D: Optional Keywords with COPY TRACKS:**

COPY Command



Notes:

- 1 Do not use the FASTREPLICATION (REQuired) keyword with the CONCURRENT keyword.
- 2 Do not use the FASTREPLICATION (NONE) keyword with the FCNOCOPY or FCTOPPRCPrimary keywords.
- 3 Do not use the FASTREPLICATION (REQuired) keyword with the CONCURRENT keyword.
- 4 Do not use the FASTREPLICATION (NONE) keyword with the FCNOCOPY or FCTOPPRCPrimary keywords.
- 5 For COPY TRACKS operations, the FCCGFREEZE, FCINCREMENTAL, and FCINCREMENTALLAST keywords require that the CPVOLUME keyword is also specified. For more information, see the keyword descriptions.
- 6 For COPY TRACKS operations, the FCCGFREEZE, FCINCREMENTAL, and FCINCREMENTALLAST keywords require that the CPVOLUME keyword is also specified. For more information, see the keyword descriptions.

Explanation of COPY Command Keywords

This section describes the keywords for the COPY command.

ADMINISTRATOR



ADMINISTRATOR lets you act as a DFSMSdss-authorized storage administrator for the COPY command. If you are not authorized to use the ADMINISTRATOR keyword, the command is ended with an error message. Otherwise, access checking to data sets and catalogs is bypassed.

To use the ADMINISTRATOR keyword all of the following must be true:

- FACILITY class is active.
- Applicable FACILITY-class profile is defined.
- You have READ access to that profile.

For more details, see “ADMINISTRATOR Keyword” on page 265.

ALLDATA



ALLDATA applies to full and data set copy operations.

dsn Specifies the fully qualified name of a data set whose data set organization is physical sequential (PS), physical sequential undefined (PSU), partitioned organization (PO), partitioned organization undefined (POU), or null.

Specify ALLDATA(dsn) or ALLDATA(*) if the data set is not empty, or ALLEXCP if the data set is empty, for the following conditions (this applies to *like* targets only):

- The data set has data beyond the last-used block pointer in the data set's VTOC entry.
- The data set has a null data set organization.
- The data set is the first or intermediate volume of a multivolume data set and has a null data set organization.

JES2/JES3 data sets can have the characteristics specified above, as can CICS® journal data sets.

The data set is processed as follows:

- For a full-volume copy, all of the allocated space for the source data set is copied to the target volume.
- For a data set copy, the function of ALLDATA is dependent upon certain data set characteristics, device characteristics, and other DFSMSdss keywords specified. See Table 2 on page 103 and Table 3 on page 104 for more information.

* (asterisk)

Specifies all data sets whose data set organization is PS, PSU, PO, POU, or null and are not empty (the last used block pointer in the data set's VTOC entry is not zero). The data sets are processed as follows:

- For a full-volume copy, all of the allocated space for the source data set is copied to the target volume.
- For a data set copy, the function of this parameter is dependent upon certain data set characteristics, device characteristics, and other DFSMSdss keywords specified. See Table 2 on page 103 and Table 3 on page 104 for more information.

COPY Command

Notes:

1. When you specify ALLDATA or ALLEXCP for a sequential extended format data set during a logical copy operation DFSMSdss does not retain data beyond the last used block pointer. Also, DFSMSdss allocates the same amount of space for the target data set as the source data set.
2. When you specify ALLDATA for a PDSE data set during a logical copy operation, DFSMSdss does not retain the data that resides in the allocated but unused space. DFSMSdss allocates the same amount of space for the target data set as the source data set.

DFSMSdss determines the amount of space allocated or used for the data set by counting how many tracks have been allocated or used by the data set. For this reason, the allocated space for the target data set may occupy more tracks than the source when going to a different device type.

Attention: Since the unused portion of the data set may or may not be copied, care should be used when specifying the ALLDATA keyword with DELETE. For example, if a data set contains records past the last used block pointer in the data set's VTOC entry that you wish to preserve and you perform a data set copy with ALLDATA and DELETE to an unlike device, these records are not copied to the target, but the source will be deleted upon successful completion of the copy.

ALLEXCP



ALLEXCP specifies all data sets whose data set organization is PS, PSU, PO, POU, or null and are empty (the last used block pointer in the data set's VTOC entry is zero). The data sets are processed as follows:

- For a full-volume copy, all of the allocated space for the source data set are copied to the target volume.
- For a data set copy, the function of this keyword is dependent upon certain data set characteristics, device characteristics, and other DFSMSdss keywords specified. See Table 2 on page 103 and Table 3 on page 104 for more information.

Attention: Since all of the allocated space may or may not be copied, care should be used when specifying the ALLEXCP keyword with DELETE. For example, if a data set contains records that you wish to preserve, but the last block pointer in the data set's VTOC entry is zero and you perform a data set copy with ALLEXCP and DELETE to an unlike device, these records are not copied to the target, but the source will be deleted upon successful completion of the copy.

AUTORELBLOCKADDRESS

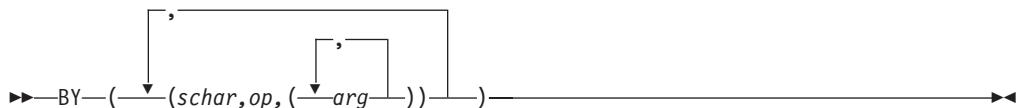


AUTORELBLOCKADDRESS specifies that direct access data sets be automatically processed by relative block address rather than by track-track-record (TTR). The data sets must be accessed with an optional services code (OPTCD) setting. This setting indicates the data sets are organized by relative block address.

Notes:

1. If any such data set is actually organized by TTR, the data set might become unusable.
2. The TTRADDRESS keyword takes precedence over the AUTORELBLOCKADDRESS keyword. Refer to the RELBLOCKADDRESS and TTRADDRESS keywords for more information.
3. AUTORELBLOCKADDRESS is ignored for direct access data sets with variable-spanned record formats or standard user labels.

Related reading: For additional information about the OPTCD, see *z/OS DFSMS Macro Instructions for Data Sets*.

BY

BY specifies that the data sets selected up to this point, by the processing of the INCLUDE and EXCLUDE keywords, are to be filtered further. To select the data set, *all* BY criteria must be met. See “Filtering by Data Set Characteristics” on page 14 for a full discussion of *schar*, *op*, *arg*, and for further information on BY filtering.

Note: You must use FILTERDD when you have more than 255 entries in INCLUDE, EXCLUDE, or BY list keywords.

BYPASSACS

BYPASSACS specifies that the automatic class selection (ACS) routines are not invoked to determine the target data set class names. To specify BYPASSACS, RACF authorization may be required.

dsn Specifies a fully or partially qualified data set name.

If a data set is being renamed, the old name must be specified.

Related reading:

- For additional information about RACF authorization, see the *z/OS DFSMSdss Storage Administration Guide*.
- For information about the assignment of class names using the COPY command, see “Assignment of Class Names by Using the RESTORE and COPY Commands” on page 219.
- For additional information about data set names, see “Filtering by Data Set Names” on page 12.

COPY Command

CANCELERROR



CANCELERROR specifies that the copy task be ended for a permanent read error, or that the copy of a data set is ended for a write error.

- Permanent read error, such as a data check:

If CANCELERROR is specified, the copy task is ended. If this keyword is not specified, the track in error is not copied and the copy continues. Only the data set receiving the error is ended, and the DFSMSdss copy function continues to process any subsequent data sets.

- Write error, such as an invalid track format:

For data set copy, processing of the data set ends and the target data set is deleted. The copy operation continues with the next data set. For full volume and tracks copy, processing for the volume ends. Subsequent tracks are not processed.

DFSMSdss allows you to change this default operation. A patch byte is provided to allow you to change the default handling of invalid tracks created during COPY processing.

During copy operations in which a utility performs the copy, DFSMSdss ignores this keyword. CANCELERROR has no effect on the following types of errors on a DASD volume:

- Equipment check
- Command reject
- Intervention required
- Busout parity

This keyword may be used in conjunction with CHECKVTOC to specify whether or not the operation is to continue in the event of terminating VTOC errors found during VTOC checking. Refer to CHECKVTOC keyword.

Related reading: For additional information about the handling of invalid tracks, see the *z/OS DFSMSdss Storage Administration Guide*.

CATALOG



CATALOG specifies that on a data set copy operation, DFSMSdss is to catalog data sets that it allocates.

CATALOG catalogs the target data set as determined by the standard catalog search order. This is the default for VSAM, multivolume data sets, and SMS-managed data sets.

RECATALOG(*newcatname*)

 catalogs the target data set in the *newcatname* catalog.

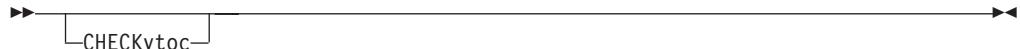
RECATALOG(*)

 catalogs the target data set in the same catalog that points to the

source data set. If the source data set was not cataloged, the new data set is not cataloged either. After DFSMSdss determines the catalog status of the data set and is changed by other means outside of DFSMSdss, the original catalog status is used.

Notes:

1. Be careful when using the RECATALOG(*newcatname*) keyword because the target data set may be cataloged outside of the standard order of search.
2. The CATALOG or RECATALOG operation fails if the target data set is already cataloged in the same catalog and DELETE, RENAMEU, or UNCATALOG is not specified. The RECATALOG keyword is ignored for SMS-managed targets.
3. An alternate index (AIX[®]) is always cataloged in the same catalog as its associated base cluster. If the base cluster is recataloged, the AIX is recataloged.
4. If you omit the CATALOG or RECATALOG keyword for a single volume, non-VSAM, non-SMS-managed data set, the target data set is uncataloged.
5. The CATALOG and RECATALOG keywords are ignored for preallocated data sets.

CHECKVTOC

CHECKVTOC specifies that a VTOC analysis of the source volume be performed during copy processing. In the event of terminating VTOC errors found during analysis, operation continues unless the CANcelerror keyword is specified. CHECKVTOC is ignored if CPVOLUME is also specified.

COPY Command

CICSVRBACKUP



CICSVRBACKUP specifies that DFSMSdss create backups for use by CICSVR for a data set copy operation. DFSMSdss notifies the CICSVR server address space when a CICSVR backup is made for a VSAM base cluster. This enables CICSVR to manage backups that are made by DFSMSdss.

CICSVR provides DFSMSdss with a new name for each VSAM base cluster that is to be copied when CICSVRBACKUP is specified. DFSMSdss uses the CICSVR-generated new name instead of the one that is specified in the RENAMEUNCONDITIONAL keyword.

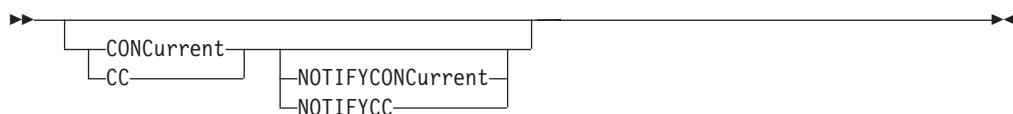
Notes:

1. CICSVRBACKUP is intended to be used with CICSVR. The minimum required CICSVR release is Version 3 Release 1. To use CICSVRBACKUP, the CICSVR server address space must be active.
2. CICSVRBACKUP applies to COPY DATASET processing only.
3. CICSVR manages VSAM base clusters that are backed up using the DFSMSdss COPY command. DFSMSdss COPY fails the processing of alternate indexes when you specify the CICSVRBACKUP keyword. Because CICSVR removes reusable alternate indexes (AIX) from the upgrade set prior to recovery and rebuilds the reusable AIXs after recovery, you need not copy the alternate indexes. DFSMSdss ignores the CICSVRBACKUP keyword when copying non-VSAM data sets.
4. CICSVRBACKUP cannot be specified with the SPHERE or DELETE keyword.
5. You must specify the RENAMEUNCONDITIONAL keyword when you specify the CICSVRBACKUP keyword. The use of RENAMEU must follow the DFSMSdss syntax rules. However, be aware that DFSMSdss uses the CICSVR-generated new name instead of the name that you specify.

Recommendation: To avoid confusion or frustration, you can specify the RENAMEU keyword as RENAMEU(**,CICSVR**).

Related reading: For additional information about CICSVR-generated new name, its naming convention, and required RENAMEU specifications, see the *CICSVR Implementation Guide*.

CONCURRENT



CONCURRENT specifies that the data is to be processed with concurrent copy if it is possible. Otherwise, the data is processed as if CONCURRENT were not specified.

If a concurrent copy operation fails after signaling that the concurrent copy initialization was complete (and update activity on the data has resumed), it is not

possible to recover the data at the point-in-time at which the concurrent copy operation was started. This is because the data may have been updated while the copy operation was progressing.

Attention: Performing concurrent copy operations against many large data sets when there is also heavy update activity (such as reorganizing data sets or initializing the volume the data sets reside on) can result in a shortage of storage. The shortage occurs because data is transferred to data-space storage faster than DFSMSdss can process it.

Note: CONCURRENT cannot be specified with DELETE or UNCATALOG keywords because after the concurrent copy operation has begun, the original data can still be updated.

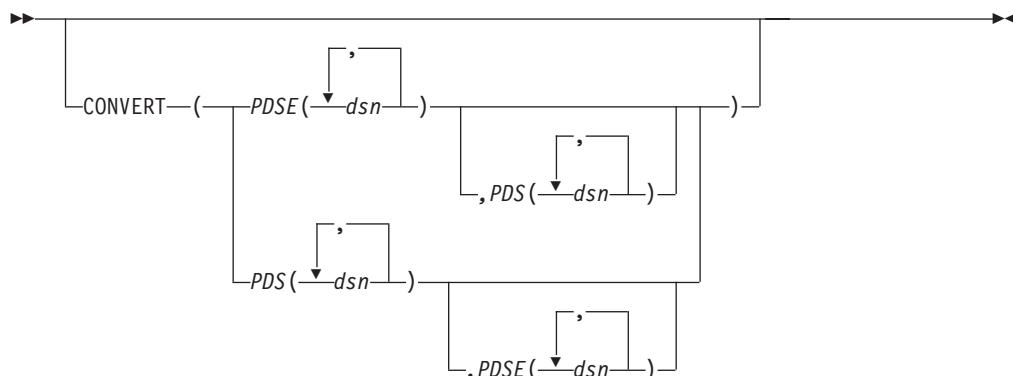
If the source volume is a RAMAC Virtual Array and CONCURRENT keyword is specified, DFSMSdss uses the SnapShot capability of the RVA to provide a function equivalent to concurrent copy. This function is called virtual concurrent copy and is transparent to the user.

NOTIFYCONCURRENT

For a logical data set copy operation, specifies that DFSMSdss issues message ADR767I for every data set that is successfully included in the concurrent copy operation. If not specified, messages are issued only for data sets that are not successfully included in the concurrent copy operation.

Related reading: For additional information about determining concurrent copy storage requirements, see the *z/OS DFSMSdss Storage Administration Guide* in the chapter on managing availability .

CONVERT



CONVERT(PDSE(dsn))

Specifies that the PDSs that are listed in the *dsn* be converted to PDSE

CONVERT(PDS(dsn))

Specifies that the PDSEs that are listed in the *dsn* be converted to PDS

Notes:

1. If the target data set is a PDSE, it must be SMS-managed.

COPY Command

COPYVOLID



COPYVOLID specifies that the volume serial number (VOLID) from the input DASD volume is to be copied to the output DASD volume. This applies to full copy operations and to tracks copy operations if track 0 (zero) is copied.

Notes:

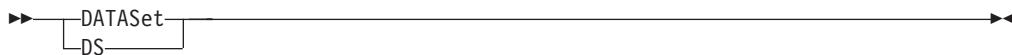
1. DFSMSdss requires the COPYVOLID keyword for a full-volume copy operation of an SMS-managed input volume—unless you specify the DUMPCONDITONING keyword.
2. When the volume serial number is changed by using a COPYVOLID keyword, profiles are not built for the RACF-protected data sets on the target volume or for the RACF DASDVOL for the RACF-protected DASD volume. When the volume serial number on a DASD volume is changed, the operator is notified. The operating system then initiates a demount of the volume.
3. Exercise caution using COPYVOLID in a multiple task job step when two or more of the tasks are using the same output volume. If the output volume is made unavailable by the first task, all succeeding tasks that use the same output volume fail.
4. COPYVOLID cannot be performed if there are permanent I/O errors or if CANCELERROR is specified. If TOLERATE(IOERROR) is honored, however, COPYVOLID is performed.
5. You cannot use the COPYVOLID keyword with the DUMPCONDITONING keyword.

CPVOLUME



CPVOLUME specifies that the input and output volumes are VM-format volumes and that the OS-compatible VTOCs must begin on track zero, record five. You must specify the track range to be copied with the TRACKS keyword, as the OS-compatible VTOCs do not describe the extents of any data on the volume. You must also specify the ADMINISTRATOR keyword with CPVOLUME because DFSMSdss cannot check access authorization for VM data.

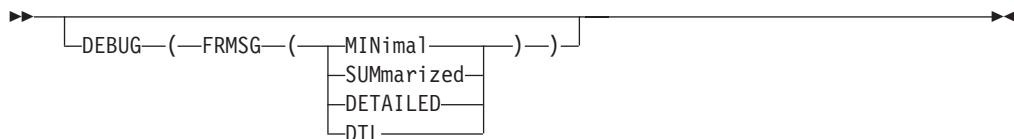
DATASET



DATASET specifies a data set copy operation using filtering. See Chapter 2, “Filtering—Choosing the Data Sets You Want Processed,” on page 11 for an explanation of the filtering process used. Unless ALLDATA or ALLEXCP is specified, only *used tracks* are copied for sequential and partitioned data sets and for data sets with a data set organization that is null (for example, JES2/JES3 data sets). If the free space map in the VTOC is invalid, all tracks for the data set are copied.

Note: Either the FILTERDD, INCLUDE, EXCLUDE, or BY keyword must be specified when data set is selected.

DEBUG



FRMSG is a subkeyword for the DEBUG keyword. DEBUG(FRMSG(MIN | SUM | DTL)) specifies that DFSMSdss issue an informational message about why one of the fast replication methods (FlashCopy or SnapShot) cannot be used during a COPY operation. DEBUG(FRMSG) cannot be specified by itself. You must use one of the subkeywords (MINimal | SUMmarized | DETAILED) when you designate this keyword.

The DEBUG(FRMSG(MINimal | SUMmarized | DETAILED)) keyword overrides the DEBUG=FRMSG parameter that is specified in the JCL EXEC statement.

FRMSG(MINIMAL)

Specifies that DFSMSdss issue an informational message with a minimal level of information about why DFSMSdss could not use a fast replication method. The following are examples of messages that are issued:

Example 1: Data set copy

ADR948I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR DATA SET TEST.SRC.KSDS1 BECAUSE THE TARGET DEVICES DO NOT PROVIDE COMPATIBLE DATA SET FAST REPLICATION FUNCTIONS

Example 2: Data set copy

ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR DATA SET TEST.SRC.KSDS1, RETURN CODE 3

Return code 3 indicates that one or more source devices are not eligible for fast replication at this time.

Example 3: Data set copy

ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR DATA SET TEST.SRC.KSDS1, RETURN CODE 15

Return code 15 indicates that for the SMS allocation, target volumes that would allow fast replication to be used could not be selected.

Example 4: Full volume or tracks copy

ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR VOLUME SRCV01, RETURN CODE 1

Return code 1 indicates that the source device is not capable of fast replication.

COPY Command

Guideline: DFSMSdss suppresses SMS allocation messages regarding fast replication during a data set copy operation when you specify the DEBUG(FRMSG(MINIMAL)) keyword.

FRMSG(SUMMARIZED)

Specifies that DFSMSdss issue an informational message with summarized information that explains why a fast replication method could not be used. When applicable, summarized information regarding ineligible volumes is provided in the message text. The following are examples of the types of messages that are issued:

Example 1: Data set copy

```
ADR948I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED  
FOR DATA SET TEST.SRC.KSDS1 BECAUSE THE TARGET DEVICES DO NOT PROVIDE  
COMPATIBLE DATA SET FAST REPLICATION FUNCTIONS  
2 VOLUMES SUPPORT DATA SET FLASHCOPY  
1 VOLUME SUPPORTS SNAPSHOT  
3 VOLUMES DO NOT SUPPORT ANY TYPE OF DATA SET FAST REPLICATION
```

Example 2: Data set copy

```
ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR DATA SET  
TEST.SRC.KSDS1, RETURN CODE 3  
1 VOLUME WAS REJECTED FOR QFRVOLS REASON CODE 7 - VERSION 1 FC RELATION  
EXISTS  
2 VOLUMES WERE REJECTED FOR QFRVOLS REASON CODE 8 - MAX ESS FC  
RELATIONS  
1 VOLUME WAS REJECTED FOR QFRVOLS VOLUME REASON CODE CA - BOUNDARY  
EXCEPTION
```

Example 3: Data set copy, target data set is non-SMS-managed

```

ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR
DATA SET TEST.SRC.KSDS1, RETURN CODE 14
  1 VOLUME WAS REJECTED FOR QFRVOLS VOLUME REASON CODE 7 - VERSION 1 FC
    RELATION EXISTS
  2 VOLUMES WERE REJECTED FOR QFRVOLS VOLUME REASON CODE 8 - MAX ESS
    FC RELATIONS
  2 VOLUMES WERE REJECTED FOR QFRVOLS VOLUME REASON CODE C9 - FLASHCOPY
    NOT SUPPORTED
  1 VOLUME WAS REJECTED FOR DFSMSDSS REASON CODE 1 - INSUFFICIENT SPACE
  1 VOLUME WAS REJECTED FOR DFSMSDSS REASON CODE 2 - NO FREE DSCB
    IN THE VTOC
  2 VOLUMES WERE REJECTED FOR DFSMSDSS REASON CODE 3 - VOLUME IS SMS
    MANAGED
  1 VOLUME WAS REJECTED FOR DFSMSDSS REASON CODE 4 - LSPACE MACRO FAILED
    WHILE CALCULATING FREE SPACE
  1 VOLUME WAS REJECTED FOR DFSMSDSS REASON CODE 8 - DADSM FAILURE
    OCCURRED WHILE ALLOCATING THE DATA SET ON THE VOLUME

```

Example 4: Data set copy, target data set is SMS-managed

```

ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR
DATA SET TEST.SRC.KSDS1, RETURN CODE 15
IGD17330I DATA SET TEST2.TGT.KSDS1 WAS ALLOCATED ON VOLUME(S) WHICH ARE
NOT ELIGIBLE FOR FAST REPLICATION. PREFERRED FAST REPLICATION
WAS SPECIFIED BY CALLER
IGD17290I THERE WERE 3 CANDIDATE STORAGE GROUPS OF WHICH THE FIRST 3
WERE ELIGIBLE FOR VOLUME SELECTION. THE CANDIDATE STORAGE
GROUPS WERE: SG1, SG2, SG3
IGD17267I THE FOLLOWING 1 CANDIDATE STORAGE GROUPS WERE INELIGIBLE FOR
PREFERRED FAST REPLICATION BECAUSE THEY DID NOT HAVE A SUFFICIENT
NUMBER (2) OF ELIGIBLE FAST REPLICATION VOLUMES: SG3
IGD17268I 2 VOLUMES WERE NOT USED FOR FAST REPLICATION BECAUSE THE SMS VOLUME
STATUS WAS DISABLED
IGD17268I 2 VOLUMES WERE NOT USED FOR FAST REPLICATION BECAUSE THEY WERE
NOT ONLINE
IGD17268I 6 VOLUMES WERE NOT USED FOR FAST REPLICATION BECAUSE OF FLASHCOPY
NOT SUPPORTED - ANTRQST QFRVOLS VOLUME RSN(201)
IGD17268I 2 VOLUMES WERE NOT USED FOR FAST REPLICATION BECAUSE OF BOUNDARY
EXCEPTION - ANTRQST QFRVOLS VOLUME RSN(202)
IGD17268I 2 VOLUMES WERE NOT USED FOR FAST REPLICATION BECAUSE OF XRC SRC
CURRENTLY ACTIVE - ANTRQST QFRVOLS VOLUME RSN(5)
IGD17268I 1 FR-ELIGIBLE VOLUMES WERE NOT USED FOR FAST REPLICATION BECAUSE
STORAGE GROUP HAS INSUFFICIENT FAST REPLICATION VOLUMES
IGD17268I 4 FR-ELIGIBLE VOLUMES WERE NOT USED FOR FAST REPLICATION BECAUSE
THEY DID NOT HAVE SUFFICIENT SPACE
IGD17269I 2 NON-FR VOLUMES WERE REJECTED BECAUSE THE SMS VOLUME STATUS WAS
DISABLED
IGD17269I 2 VOLUMES WERE REJECTED BECAUSE THEY WERE NOT ONLINE
IGD17269I 1 VOLUMES WERE REJECTED BECAUSE OF A DADSM FAILURE
IGD17269I 5 VOLUMES WERE REJECTED BECAUSE THEY DID NOT HAVE SUFFICIENT
SPACE

```

Example 5: Full volume or tracks copy

```

ADR947I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR VOLUME SRCV01 BECAUSE
THE SOURCE AND TARGET DEVICES DO NOT PROVIDE COMPATIBLE FAST REPLICATION FUNCTIONS
VOLUME SRCV01 SUPPORTS DATA SET FLASHCOPY
VOLUME TGT01 SUPPORTS SNAPSHOT

```

In examples 4 and 5, DEBUG(FRMSG(SUM)) and DEBUG(FRMAG(DTL)) result in the same level of informational message being issued.

COPY Command

Guidelines:

- DFSMSdss supplies fast replication ineligible reasons at the summarized level when summary information is applicable.
- Specifying SUMMARIZED causes DFSMSdss to issue SMS allocation messages regarding fast replication in a data set copy operation when the target data set is SMS-managed.
- When the FASTREPLICATION(REQUIRED) keyword is specified and the DEBUG(FRMSG(MIN | SUM | DTL)) keyword is not specified, DFSMSdss still issues an informational message when a fast replication method cannot be used. It is as though the DEBUG(FRMSG(SUMMARIZED)) keyword had been specified.

FRMSG(DETAILED)

Specifies that DFSMSdss issue an informational message with details for why a fast replication method could not be used. When applicable, detailed information regarding ineligible volumes is provided in the message text. The following are examples of the types of messages that are issued:

Example 1: Data set copy

```
ADR948I (ttt)-mmmm(y), FAST REPLICATION COULD NOT BE USED FOR  
DATA SET TEST.SRC.KSDS1 BECAUSE THE SOURCE DEVICES DO NOT PROVIDE COMPATIBLE  
DATA SET FAST REPLICATION FUNCTIONS  
VOLUME SRCV01 SUPPORTS DATA SET FLASHCOPY  
VOLUME SRCV02 SUPPORTS DATA SET FLASHCOPY  
VOLUME SRCV03 DOES NOT SUPPORT ANY TYPE OF DATA SET FAST REPLICATION  
VOLUME SRCV14 SUPPORTS SNAPSHOT  
VOLUME SRCV25 DOES NOT SUPPORT ANY TYPE OF DATA SET FAST REPLICATION  
VOLUME SRCV26 DOES NOT SUPPORT ANY TYPE OF DATA SET FAST REPLICATION
```

Example 2: Data set copy

```
ADR918I (ttt)-mmmm(y), FAST REPLICATION COULD NOT BE USED FOR  
DATA SET TEST.SRC.KSDS1, RETURN CODE 3  
VOLUME SRCV01 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE 7 - VERSION  
1 FC RELATION EXISTS  
VOLUME SRCV02 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE 8 - MAX ESS  
FC RELATIONS  
VOLUME SRCV03 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE 8 - MAX ESS  
FC RELATIONS  
VOLUME SRCV04 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE CA - BOUNDARY  
EXCEPTION
```

Example 3: Data set copy, target data set is non-SMS-managed

```

ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR
DATA SET TEST.SRC.KSDK1, RETURN CODE 14
    VOLUME TGTVO1 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE 7 - VERSION 1
        FC RELATION EXISTS
    VOLUME TGTVO2 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE 8 - MAX ESS
        FC RELATIONS
    VOLUME TGTVO3 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE 8 - MAX ESS
        FC RELATIONS
    VOLUME TGTVO4 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE C9 - FLASHCOPY
        NOT SUPPORTED
    VOLUME TGTVO5 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE C9 - FLASHCOPY
        NOT SUPPORTED
    VOLUME TGTVO11 WAS REJECTED FOR DFMSDSS REASON CODE 1 - INSUFFICIENT
        SPACE
    VOLUME TGTVO22 WAS REJECTED FOR DFMSDSS REASON CODE 2 - NO FREE DSCB IN THE VTOC
    VOLUME TGTVO1 WAS REJECTED FOR DFMSDSS REASON CODE 3 - VOLUME IS
        SMS MANAGED
    VOLUME TGTVO2 WAS REJECTED FOR DFMSDSS REASON CODE 3 - VOLUME IS
        SMS MANAGED
    VOLUME TGTVO23 WAS REJECTED FOR DFMSDSS REASON CODE 4 - LSPACE MACRO
        FAILED WHILE CALCULATING FREE SPACE
    VOLUME TGTVO24 WAS REJECTED FOR DFMSDSS REASON CODE 8 - DADSM FAILURE
        OCCURRED WHILE ALLOCATING THE DATA SET ON THE VOLUME

```

Example 4: Full volume or tracks copy

```

ADR947I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR
VOLUME SRCV01 BECAUSE THE SOURCE AND TARGET DEVICES DO NOT PROVIDE
COMPATIBLE FAST REPLICATION FUNCTIONS
    VOLUME SRCV01 SUPPORTS DATA SET FLASHCOPY
    VOLUME TGTVO1 SUPPORTS SNAPSHOT

```

Guidelines:

- DFSMSdss supplies fast replication ineligible reasons at the individual volume level when detailed information is applicable.
- Specifying DETAILED causes DFSMSdss to issue SMS allocation messages regarding fast replication in a data set copy operation when the target data set is SMS-managed. For an SMS allocation, DFSMSdss supplies the same level of information as if you had specified DEBUG(FRMSG(SUMmarized)).
- For a non-SMS allocation during a data set copy operation, DFSMSdss supplies fast replication ineligible reasons at the individual volume level.

DELETE

DELETE specifies that for a data set copy DFSMSdss deletes VSAM and non-VSAM data sets from the source volume after a successful copy. This moves, in effect, a data set from one volume to another. The data sets are scratched and uncataloged.

Notes:

1. You must specify DELETE when you are copying cataloged data sets. If you do not specify DELETE when you are copying cataloged data sets, the target data

COPY Command

- set must either be cataloged (using the RECATALOG keyword) in a different catalog or renamed (using the RENAMEU keyword).
2. If you copy a data set with DFM attributes to a non-SMS-managed target, the new data set will not have the DFM attributes.
 3. *Unexpired* source data sets are deleted only if you also specify PURGE.
 4. Even if PROCESS (SYS1) is specified, SYS1.VVDS and SYS1.VTOCIX data sets cannot be copied and deleted.
 5. Do not specify SHARE if you specify DELETE.
 6. If DFSMSdss encounters a damaged PDS during logical data set copy, it displays messages indicating the nature and relative location of the problem. In order to maintain complete data integrity, DFSMSdss does not delete the source data set. The copy of the data set fails, and the target is deleted. In order to copy and delete a damaged PDS, use the NOPACKING keyword.
 7. Do not specify DELETE with CONCURRENT, because after the concurrent copy operation has begun, the original data can still be updated.
 8. Do not specify DELETE with CICSVRBACKUP.
 9. Specify DELETE to preserve aliases that are associated with non-VSAM data sets. The following criteria must be met for this to work:
 - RENAMEU cannot be specified at the same time.
 - The data set must be SMS-managed and remain SMS-managed during the copy operation.

Related reading: For additional information about copying non-VSAM data sets that have aliases, see the *z/OS DFSMSdss Storage Administration Guide*.

DUMPCONDITONING



DUMPCONDITONING specifies that you want to create a copy of the source volume for backup purposes rather than for the applications to use the target volume.

When you specify DUMPCONDITONING, the volume serial number of the target volume does not change, and the target volume remains online after the copy. The VVDS and VTOC index names on the target volume will not change to match the target volume serial number. They will continue to match the source volume serial number. This volume is a “conditioned volume.”

Notes:

1. Do not use the DUMPCONDITONING keyword with the COPYVOLID keyword.
2. DUMPCONDITONING applies to TRACKS COPY operations only if the tracks selected for copying include the VTOC. Otherwise, DFSMSdss ignores DUMPCONDITONING.
3. You may not be able to access data on the target volume after a DUMPCONDITONING COPY operation. This is because the VVDS and VTOC index names do not match the target *volser*. Use the resulting target volume for either of the following operations: as the source volume for a FULL volume DUMP operation or the source of another FULL volume DUMPCONDITONING COPY operation.

4. You must specify the DUMPCONDITIONING keyword to perform a FULL volume COPY operation of a conditioned volume.

DYNALLOC

```
►— [ DYNALLOC ] —►
```

DYNALLOC specifies dynamic allocation, instead of enqueue, to serialize the use of data sets. The data sets whose extents are to be relocated are serialized throughout the copy operation. This allows cross-system serialization in a JES3/MVS environment.

Notes:

1. Serialization is of value only when you use the dynamic allocation or the JES3 interface is not disabled.
2. Run time increases when you use the DYNALLOC keyword to serialize data sets (as opposed to enqueue) because overhead is involved in dynamic allocation and serialization across multiple processors.
3. If a data set passes INCLUDE/EXCLUDE filtering and is migrated before BY filtering and the DYNALLOC keyword is used, the dynamic allocation causes the data set to be recalled. DFSMSdss waits for the recall processing to complete. If the data set is recalled to a different volume, a message indicates that the VTOC entry was not found.
4. For an HFS source data set, DFSMSdss ignores DYNALLOC and attempts to get a SYSZDSN enqueue. If the enqueue attempt fails, DFSMSdss attempts to quiesce the HFS data set.

EXCLUDE

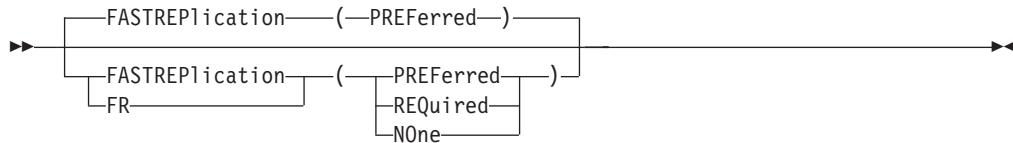
```
►— EXCLUDE ( [ dsn ] ) —►
```

dsn Specifies the name of a data set to be excluded from the data sets selected by the INCLUDE keyword. Either a fully or a partially qualified data set name can be used. See the separate discussions of INCLUDE and BY for information on how these keywords are specified.

Note: You must use FILTERDD when you have more than 255 entries in INCLUDE, EXCLUDE, or BY list keywords.

COPY Command

FASTREPLICATION



FASTREPLICATION specifies whether the use of fast replication is required, preferred, or not desired. This keyword applies to fast replication methods such as FlashCopy and SnapShot. It does not affect concurrent copy or virtual concurrent copy processing.

REQUIRED specifies that fast replication must be used. For full volume or tracks COPY operations, DFSMSdss fails the operation if fast replication cannot be used. For a data set COPY operation, DFSMSdss stops processing the current data set if fast replication cannot be used. However, DFSMSdss continues processing the rest of the data sets using fast replication. When the DEBUG(FRMSG(MIN | SUM | DTL)) keyword is not specified, DFSMSdss still issues summarized information regarding why a fast replication method cannot be used as though DEBUG(FRMSG(SUMMARIZED)) had been specified. The DEBUG(FRMSG(MIN | SUM | DTL)) keyword determines the amount of information provided for why you cannot use a fast replication method.

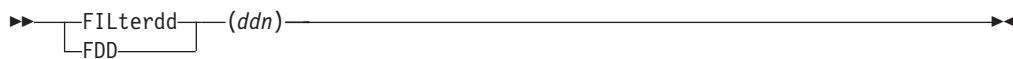
PREFERRED is the default. PREFERRED specifies that the use of fast replication is preferred. If fast replication cannot be used, DFSMSdss completes the operation using traditional data movement methods.

NONE specifies that fast replication should not be used. DFSMSdss does not attempt to use fast replication and completes the operation using traditional data movement methods.

Notes:

1. Do not use the FASTREPLICATION (REQUIRED) keyword with the CONCURRENT keyword.
2. Do not use the FASTREPLICATION (NONE) keyword with the FCNOCOPY keyword.

FILTERDD



ddn Specifies the name of the DD statement that identifies the sequential data set or member of a partitioned data set that contains the filtering criteria to be used. This is in the form of card-image records, in DFSMSdss command syntax, that contain the INCLUDE, EXCLUDE, and BY keywords.

Note: You must use FILTERDD when you have more than 255 entries in INCLUDE, EXCLUDE, or BY list keywords.

&

FCCGFREEZE

&



&

|
|
|
|
|
|
FCCGFREEZE specifies that the source volume is to be part of a FlashCopy Consistency Group. Subsequent I/O activity to the FlashCopy source volume will be held until the CGCREATE (thaw) command is processed on the logical subsystem (LSS) where the volume resides or when the FlashCopy Consistency Group timer expires.

&

Notes:

&

1. Do not specify FCCGFREEZE with any of the following keywords:

&

- CONCURRENT
- DATASET
- FASTREPLICATION(PREFERRED | NONE)
- FCNOCOPYTOCOPY

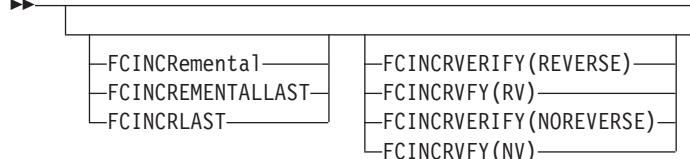
&

2. DFSMSdss supports FlashCopy Consistency Group at a volume level. When you specify FCCGFREEZE with TRACKS, you must also specify the CPVOLUME keyword. FCCGFREEZE is supported with the TRACKS keyword only for full volume copy of VM-format volumes with OS-compatible VTOCs.
3. When FCCGFREEZE is specified, it indicates FlashCopy Version 2 must be used to copy the data. If FlashCopy V2 cannot be used, the copy operation will fail.
4. The FCCGCREEZE option requires the specified devices support the FlashCopy Consistency Group function.

&

FCINCREMENTAL

&



&

&
&
&
&
&
FCINCREMENTAL specifies that DFSMSdss establishes a full volume Incremental FlashCopy relationship from the specified source volume (in the INDD/INDYNAM keyword) to the specified target volume (in the OUTDD/OUTDYNAM keyword). The full volume FlashCopy relationship remains in effect after the initial copy has completed and subsequent changes to the source and target volumes will be tracked so that a future FlashCopy operation will be performed to only copy incremental changes.

&

&
&
When FCINCREMENTAL is specified and no Incremental FlashCopy relationship currently exists between the volume pair, the storage subsystem will initiate background copy of the entire source volume to the target volume.

&

&
&
When FCINCREMENTAL is specified and an Incremental FlashCopy relationship already exists between the volume pair, the storage subsystem will only copy the changed data in the specified direction which can be the same as the existing

COPY Command

& (original) or the reverse of the existing FlashCopy direction. When no updates have
& been made to the existing target since the last Incremental FlashCopy, the reverse
& of FlashCopy direction can be used to restore the original source volume back to
& the previous point-in-time copy state. The new source volume is designated by the
& INDD/INDYDAM keyword and the new target is designated by the
& OUTDD/OUTDYDAM keyword on the copy command.

& **Attention:** DFSMSdss does not inhibit writes to the FlashCopy source or target
& volumes. If you intend to use the Incremental FlashCopy target volume as a
& backup, you must ensure the target volume is not updated inadvertently.

& **Notes:**

1. Do not specify the FCINCREMENTAL keyword with any of the following keywords:
 - DATASET
 - FCINCRMENTALLAST
 - CONCURRENT
 - FASTREPLICATION(PREFERRED | NONE)
 - FCNOCOPY
 - FCNOCOPYTOCOPY
2. Incremental FlashCopy is supported at a volume level. It is not supported for data set FlashCopy.
3. Incremental FlashCopy relationship is limited to one full volume relationship per volume. However, an Incremental FlashCopy relationship can coexist with other non-incremental relationships.
4. When you specify FCINCREMENTAL with TRACKS, you must also specify the CPVOLUME keyword. FCINCREMENTAL is supported with the TRACKS keyword only for full volume copy of VM-format volumes with OS-compatible VTOCs.
5. The benefit of Incremental FlashCopy is to minimize the amount of data transfer needed when a FlashCopy pair is refreshed. There is no saving in data transfer when the no-background copy option is chosen. Therefore, DFSMSdss does not allow FCINCREMENTAL and FCNOCOPY to be specified together.
6. When FCINCREMENTAL is specified, it requires that the storage facility has the Change Recording feature enabled. The Incremental FlashCopy request will fail if the storage facility does not support Change Recording.
7. The Incremental FlashCopy direction can only be reversed when the previous physical background copy has completed. If the background copy is still in progress, the new Incremental FlashCopy attempt will fail. You can instruct DFSMSdss to wait for background copy to complete by specifying the FCWAIT keyword. See the FCWAIT keyword description for more information.
8. FCINCREMENTAL specifies that the full volume FlashCopy relationship remains in effect (persists) after the background copy has completed and subsequent changes to the source and target volumes will be tracked. You can specify the INCREMENTALLAST keyword instead of FCINCREMENTAL If you want DFSMSdss to initiate FlashCopy of the final increment, stop tracking the changes, and have the relationship ended when background copy has completed.

FCINCREMENTALLAST

See "FCINCREMENTAL" in "FCINCREMENTAL" on page 65 for syntax diagram.

FCINCREMENTALLAST specifies that DFSMSdss establishes FlashCopy of the final increment from the specified source volume (in the INDD/INDYNAM keyword) to the specified target volume (in the OUTDD/OUTDYNAM keyword), stops change recording, and have the FlashCopy relationship ended when the background copy has completed.

When FCINCREMENTALLAST is specified and no Incremental FlashCopy relationship currently exists between the volume pair, DFSMSdss will establish a non-incremental FlashCopy of the entire source volume to the target volume. The FlashCopy relationship will end when the background copy has completed.

When FCINCREMENTALLAST is specified and an Incremental FlashCopy relationship already exists between the volume pair, the storage subsystem will only copy the changed data in the specified direction which can be the same as the existing (original) or the reverse of the existing FlashCopy direction. When no updates were made to the existing target since the previous Incremental FlashCopy, the reverse of FlashCopy direction can be used to restore the original source back to the previous point-in-time copy state. The new source volume is designated by the INDD/INDYNAM keyword and the new target is designated by the OUTDD/OUTDYNAM keyword on the copy command.

See the INCREMENTAL keyword description for additional information on Incremental FlashCopy

Notes:

1. Do not specify the FCINCREMENTALLAST keyword with any of the following keywords:
 - DATASET
 - FCINCRMENTAL
 - CONCURRENT
 - FASTREPLICATION(PREFERRED | NONE)
 - FCNOCOPY
 - FCNOCOPYTOCOPY
2. When you specify FCINCREMENTALLAST with TRACKS, you must also specify the CPVOLUME keyword. FCINCREMENTALLAST is supported with the TRACKS keyword only for full volume copy of VM-format volumes with OS-compatible VTOCs.
3. When FCINCREMENTALLAST is specified, it requires that the storage facility has the Change Recording feature enabled. The Incremental FlashCopy request will fail if the storage facility does not support Change Recording.
4. The Incremental FlashCopy direction can only be reversed when the previous physical background copy has completed. If the background copy is still in progress, the new Incremental FlashCopy attempt will fail. You can instruct DFSMSdss to wait for background copy to complete by specifying the FCWAIT keyword. See the FCWAIT keyword description for more information.
5. If you want the full volume Incremental FlashCopy relationship to remain in effect (persists) after the copy has completed and subsequent changes to the source and target volumes to be tracked, specify the FCINCREMENTAL keyword.

FCINCRVERIFY

NOREVERSE Specifies that the new FlashCopy direction is the same as the existing (original) FlashCopy direction.

COPY Command

REVERSE Specifies that the new FlashCopy direction is the reverse of the existing (original) FlashCopy direction.

FCINCRVERIFY specifies that DFSMSdss should verify the new and the existing Incremental FlashCopy direction before copying incremental changes. The new source volume is designated by the INDD/INDYDAM keyword and the new target is designated by the OUTDD/OUTDYDAM keyword on the copy command. If the new and the existing FlashCopy directions match what the user expected -- the same or reversed -- DFSMSdss will proceed to copy the incremental changes in the specified (new) direction. If the new and the existing FlashCopy directions do not match what the user expected, DFSMSdss will fail the copy task without copying the new increment.

When FCINCRVERIFY is specified and no Incremental FlashCopy relationship currently exists, DFSMSdss will fail the copy task without establishing a new FlashCopy relationship.

When FCINCRVERIFY is specified and an Incremental FlashCopy relationship exists between the specified volume pair, DFSMSdss will verify the FlashCopy directions before proceeding.

Note: When FCINCRVERIFY is specified, either the FCINCREMENTAL or the FCINCREMENTALLAST keyword must also be specified.

FCNOCOPY



FCNOCOPY specifies that if FlashCopy is used to perform the copy operation, then the ESS subsystem does not perform a physical copy of the data. If FCNOCOPY is not specified and FlashCopy is used to perform the operation, then the ESS subsystem performs a physical copy of the data in order to release the subsystem resources that are used to maintain the FlashCopy relationship (a virtual copy of the data).

When FlashCopy is not used to perform the copy operation, the FCNOCOPY keyword is ignored.

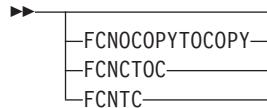
Notes:

1. Do not specify the FCNOCOPY keyword with the FASTREPLICATION(NONE) keyword.
2. If FCNOCOPY is not specified and the ESS subsystem performs the physical copy, the DFSMSdss copy operation will not be delayed. However, performing the physical copy uses subsystem resources, which can impact the performance of other I/O operations that are issued to the ESS.
3. Be aware that if you use the FCNOCOPY keyword, you must withdraw the FlashCopy relationship in which the copy is no longer needed in order to free up the subsystem resources that maintain the FlashCopy relationship. You can withdraw the FlashCopy relationship by performing one of the following options:
 - Initiate a dump of the target of the copy and specify the FCWITHDRAW keyword on the DUMP command.
 - Initiate the TSO FCWITHDR command.

&

FCNOCOPYTOCOPY

&



&

&
&
&
&
&
&
FCNOCOPYTOCOPY specifies that DFSMSdss initiate background copy between the specified source and any target with existing FlashCopy no-background copy (NOCOPY) relationships associated with that source. As a result, the remaining unchanged source tracks will be written to the target. When the physical background copy completes, the FlashCopy relationship will end unless the relationship is persistent.

&
&
&
&
If there are not any existing FlashCopy no-background copy relationships associated with the specified source, no operations will be performed as a result of this COPY command. No messages will be issued if there are not any existing relationships to be converted.

&

Notes:

1. Do not specify the FCNOCOPYTOCOPY keyword with any of the following keywords:
 - DELETE
 - FASTREPLICATION(REQUIRED | PREFERRED | NONE)
 - FCCGFREEZE
 - FCINCREMENTAL | FCINCREMENTALLAST
 - FCNOCOPY
2. FCNOCOPYTOCOPY operation does not create a new copy of the source data.
3. FCNOCOPYTOCOPY operation initiates background copy of any NOCOPY FlashCopy relationships in which the specified source is participating. If output data sets, tracks (OUTTRACKS), or volumes (OUTDDNAME/OUTDYNAM) are specified, they are ignored.
4. The source extent ranges specified or determined by DFSMSdss in the FCNOCOPYTOCOPY conversion request might not match the existing FlashCopy relationships. Any existing no-background copy FlashCopy relationships with extent ranges intersecting the source tracks specified in the conversion request will have the entire relationship converted. For example, if FlashCopy no-background copy relationships were established for 2 extents: tracks 1 through 50 and tracks 70 through 100, a subsequent COPY FCNOCOPYTOCOPY issued for an extent range of tracks 30 through 90 would result in background copy being initiated for tracks 1 through 50 and 70 through 100. Tracks 51 through 69 would be ignored.
5. The existing FlashCopy relationships may have been established with the no-background copy option by a previous DFSMSdss copy job with the FCNOCOPY option, TSO FCESTABL command with MODE(NOCOPY), or other programs. The FCNOCOPYTOCOPY option will convert any existing FlashCopy no-background copy relationships regardless of which program established the NOCOPY relationships.

COPY Command

FCTOPPRCPrimary



FCTOPPRCPrimary specifies that if FlashCopy is used to perform the copy operation, a Peer-toPeer Remote Copy (PPRC) primary volume is allowed to become a FlashCopy target volume. To specify FCTOPPRCPrimary, RACF authorization may be required.

When FlashCopy is not used to perform the copy operation, the FCTOPPRCPrimary keyword is ignored.

When FCTOPPRCPrimary is not specified or if the capability is not supported by the ESS, a PPRC primary volume is not eligible to become a FlashCopy target volume.

Attention: When FCTOPPRCPrimary is specified the FlashCopy operation will cause a PPRC primary volume to become a FlashCopy target volume. The PPRC-SYNC volume pair currently in full duplex state will go into a duplex pending state when the FlashCopy relationship is established. When PPRC completes the copy operation, the PPRC_SYNC volume pair will go to full duplex state. For more information on PPRC options and volume states see: *z/OS DFSMS Advanced Copy Services*.

Notes:

1. Do not specify the FCTOPPRCPrimary keyword with the FASTREPLICATION(NONE) keyword.

Related reading:

- For additional information about RACF authorization see: *z/OS DFSMSdss Storage Administration Guide*.
- For additional information about RACF FACILITY class profiles see: *z/OS Security Server RACF Security Administrator's Guide*.
- For additional information about PPRC (PPRC-SYNC), PPRC-XD, and PPRC V2 see: *z/OS DFSMS Advanced Copy Services* and the IBM TotalStorage Enterprise Storage Server Implementing ESS Copy Services redbook.

FCWAIT



numsecs

Specifies a decimal number (0-255) that designates the time, in seconds, to wait before checking for physical background copy completion.

numretries

Specifies a decimal number (0-99) that designates the maximum number of additional queries to make on physical background copy completion.

FCWAIT specifies to DFSMSdss the length of the wait in seconds, and the number of additional queries on FlashCopy background copy completion. The combination of retry interval and maximum number of retries (numsecs times numretries)

designated in the FCWAIT keyword specifies the maximum length of time for DFSMSdss to wait for an existing physical background copy to complete before either initiating FlashCopy Establish or failing the COPY FULL or COPY TRACKS operation.

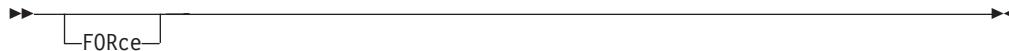
When FCWAIT(numsecs,numretries) is specified, DFSMSdss will check for existing background copy completion before initiating a FlashCopy attempt. If no background copy is currently in progress from the COPY source to the target, DFSMSdss will establish FlashCopy immediately. If background copy is in progress, DFSMSdss will recheck at the specified interval until the designated number of retries has been reached. If background copy remains in progress when the maximum wait time has been reached, DFSMSdss will fail the COPY FULL or COPY TRACKS operation.

The default for numsecs,numretries is (0,0). In other words, when FCWAIT is not specified, or when either numsecs or numretries is 0, DFSMSdss will attempt to establish FlashCopy without waiting for active background copy to end.

Notes:

1. The FCWAIT keyword is ignored when the FCINCREMENTAL or FCINCREMENTALLAST keyword is not also specified.
2. The FCWAIT keyword is ignored when Incremental FlashCopy direction is not being reversed.
3. The FCWAIT keyword is ignored if FlashCopy cannot be attempted.

FORCE



FORCE specifies that DFSMSdss copy one or more unmovable data sets to a like or unlike device type. Unmovable data sets are those allocated as absolute track (ABSTR) or as unmovable (PSU, POU, DAU, or ISU). The allocation attribute, unmovable or ABSTR, is carried over to the output volume.

When copying to like devices, DFSMSdss copies the data sets to the same track locations on the target volume. In this case, FORCE is not required if the target volume uses an indexed VTOC, and the space where the unmovable data set is to reside is available. If any of these conditions is not true, DFSMSdss does not copy any unmovable data sets unless FORCE is specified. In this case, DFSMSdss places the unmovable data sets in any available location.

You must specify FORCE when copying unmovable data sets to unlike devices. DFSMSdss places the unmovable data sets in any available location.

Restriction: Use the EXCLUDE keyword (with the FORCE keyword) to designate data sets that have CCHHR (cylinder, cylinder, head, head, record) location-dependent data. This prevents DFSMSdss from moving the location-dependent data sets.

FORCECP

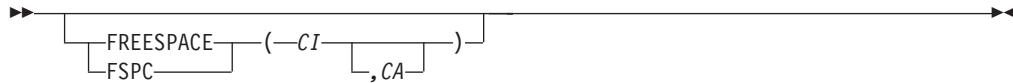


COPY Command

FORCECP specifies that checkpoint data sets resident on the SMS volume or volumes can be copied. Checkpoint indicators are removed from the target data set.

days Specifies a decimal number in the range of zero to 255, and specifies the number of days that must have elapsed since the last referenced date before the data set can be copied.

FREESPACE



FREESPACE specifies free space values for DFSMSdss-allocated target VSAM data sets. If this keyword is omitted, the control interval and control area free space are the same as the source data set.

CI Specifies the percentage of free space to be kept in each control interval during allocation of the data set.

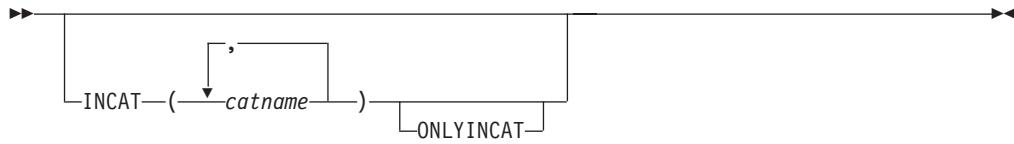
CA Specifies the percentage of free space to be kept in each control area during allocation of the data set. When omitted, the control area free space is the same as the source data set.

FULL



FULL specifies that an entire DASD volume is to be copied. This is the default. Unallocated tracks are not copied. Unless specified by ALLDATA or ALLEXCP, only the used (rather than allocated) tracks are copied for sequential data sets, partitioned data sets, and for data sets with unknown data set organization (for example, JES2/JES3 data sets with a data set organization that is null). If the VTOC has errors, all tracks are copied. Used tracks consist of the tracks from the beginning of the data set to the last-used track (as indicated by the last used block pointer in the data set's VTOC entry).

Note: You cannot specify the SHARE or TOL(ENQF) keywords for FULL operations.

INCAT

INCAT(*catname*) specifies that DFSMSdss search the user catalogs specified by the INCAT(*catname*) keyword, then follow the standard search order to locate data sets. INCAT(*catname*) allows you to identify specific source catalogs. You might need RACF authorization to use the INCAT keyword.

catname Specifies a fully qualified catalog name.

ONLYINCAT Specifies that DFSMSdss only searches catalogs that are specified in the INCAT catalog name list.

DFSMSdss does not process an SMS-managed data set that is cataloged outside the standard order of search, even if it is cataloged in one of the catalogs that is specified with the INCAT keyword. Ensure that the SMS-managed data sets are cataloged under standard catalog search order.

INCLUDE

dsn Specifies the name of a data set eligible to be copied. Either a fully or a partially qualified data set name can be used. See “Filtering by Data Set Names” on page 12. If INCLUDE is omitted (but EXCLUDE or BY is specified) or INCLUDE(**) is specified, *all* data sets are eligible to be selected for copying.

Restrictions:

- You must use FILTERDD when you have more than 255 entries in INCLUDE, EXCLUDE, or BY list keywords.
- DFSMSdss does not support INCLUDE filtering of non-VSAM data sets using an alias.

INDDNAME

ddn Specifies the name of the DD statement that identifies a volume to be copied. For a data set copy operation, you can specify multiple names (that is, multiple volumes), separated by commas. For single-volume data sets, each of the DD statements corresponding to a DDNAME (*ddn*) must identify only one volume serial number.

COPY Command

Note: If no input volumes are specified for a data set copy operation, DFSMSdss selects from all data sets cataloged in the catalogs accessible through the standard search order. If either INDDNAME or INDYNAM is specified, DFSMSdss still uses the standard catalog search order, but it selects data sets only from the specified volumes. For multivolume data sets, use LOGINDDNAME or LOGINDYNAM with SELECTMULTI.

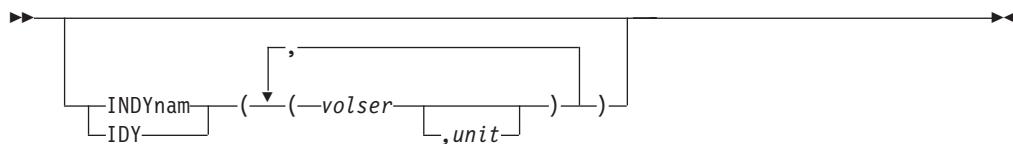
When DFSMSdss invokes IEHMOVE to copy multivolume non-VSAM data sets, IEHMOVE requires that the first DD statement in the job stream must identify all input volumes. DFSMSdss requires separate DD statements for the input volumes. To accommodate both requirements, you must code your JCL as follows:

```
//INDD1 DD UNIT=(SYSDA,2),VOL=SER=(VOL1,VOL2),DISP=SHR  
//INDD2 DD UNIT=SYSDA,VOL=SER=VOL2,DISP=SHR
```

and code your DFSMSdss control statement:

```
COPY LOGINDD(INDD1,INDD2) ...
```

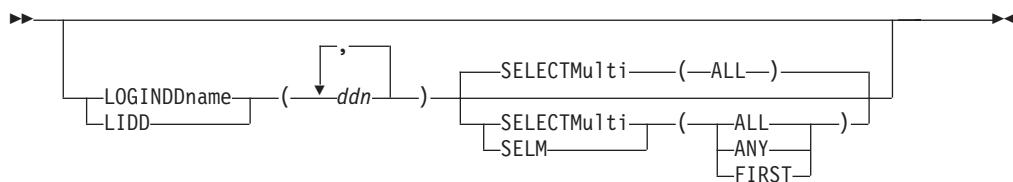
INDYNAM



The INDYNAM keyword specifies the input volumes that are to be dynamically allocated and copied. The volume must be both mounted and online. You cannot specify a nonspecific volume serial number by using an asterisk (*). Only one volume is allowed for a FULL or tracks COPY. Data set copy operations allow multiple volumes. To ease JCL coding and command input without appreciably increasing run time, use the INDYNAM keyword instead of data definition statements to allocate DASD volumes.

- volser* Specifies the volume serial number of a DASD volume that will be copied.
unit Specifies the device type of a DASD volume that will be copied. This parameter is optional.

LOGINDDNAME

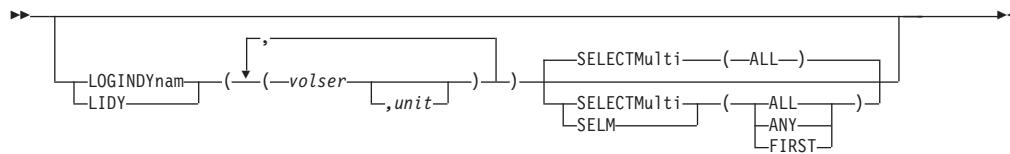


LOGINDDNAME specifies that data sets be selected from the specified volume or volumes (which you can also do by specifying INDDNAME or INDYNAM) and also allows you to specify SELECTMULTI (which cannot be done with INDDNAME or INDYNAM).

- ddn* Specifies the name of the DD statement that identifies a volume that contains the data sets to be copied. For single-volume data sets, each of the DD statements corresponding to a DDNAME (*ddn*) must identify only one volume serial number.

See the **SELECTMULTI** description and the **Notes** under LOGINDYNAM.

LOGINDYNAM



LOGINDYNAM specifies that the volumes that contain the data sets to be copied are dynamically allocated. The volume must be mounted and online. You cannot specify a nonspecific volume serial number using an asterisk (*).

volser Specifies the volume serial number of a DASD volume to be copied.

unit Specifies the device type of a DASD volume to be copied. This parameter is optional.

SELECTMULTI

Specifies the method for determining how cataloged multivolume data sets are to be selected during a logical data set copy operation. SELECTMULTI is accepted only when logical volume filtering is specified with the following keywords:

- LOGINDDNAME
- LOGINDYNAM
- STORGRP

If logical volume filtering is not used, the specification of SELECTMULTI is not accepted.

ALL Specifies that DFSMSdss *not copy* a multivolume data set unless the following criteria is met:

- The volume list created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must list all the volumes that contain a part of the data set.
- The volume list created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must list all the volumes that contain a part of the VSAM cluster.

ALL is the default.

For VSAM data sets, the volume list is affected by the use of the SPHERE keyword as follows:

- Specify SPHERE and you only need to list all parts of the base cluster in the volume list.
- Do not specify SPHERE and you must list all parts of the base cluster and the associated alternate indexes in the volume list.

COPY Command

ANY Specifies that DFSMSdss copy a multivolume data set when the following criteria is met:

- Any volume specified in the volume list that is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must contain a part of the data set.
- Any volume specified in the volume list that is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must contain a part of the VSAM cluster.

For VSAM data sets, the volume list is affected by the use of the SPHERE keyword as follows:

- Specify SPHERE and you only need to list any part of the base cluster in the volume list.
- Do not specify SPHERE and you must list any part of the base cluster and the associated alternate indexes in the volume list.

FIRST Specifies that DFSMSdss copy a multivolume data set only when the volume list designates the volume that contains the first part of the data set. The volume list is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword.

For VSAM data sets, the volume list is affected by the use of the SPHERE keyword as follows:

- Specify SPHERE and you only need to list the volume containing the first extent of the data component in the volume list.
- Do not specify SPHERE and you must list the following in the volume list:
 - The volume containing the first extent of the data component for the base cluster.
 - The volume containing the first extent of the data component for the associated alternate indexes

Notes for LOGINDDNAME, LOGINDYNAM and STORGRP keywords:

1. If the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword is not specified, DFSMSdss selects from all data sets that are cataloged in the catalogs that are accessible through the standard search order.
2. If the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword is specified, DFSMSdss still uses the standard catalog search order, but it selects data sets only from the specified volumes.
3. You must specify the SELECTMULTI keyword to copy a multivolume data set that has extents on volumes which are not identified with the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword.

MAKEMULTI



MAKEMULTI allows DFSMSdss to convert single volume data sets into multivolume data sets. The default is not to convert single volume data sets into multivolume data sets.

This keyword applies only to SMS-managed target data sets. Only single volume, non-VSAM data sets are eligible to be changed into multivolume data sets.

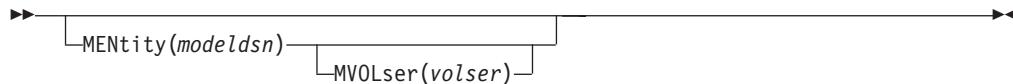
SMS-managed target data sets are given a volume count (VOLCOUNT) that is either:

- The number of SMS output volumes specified in the COPY command, if output volumes are specified through OUTDDNAME or OUTDYNAM
- The number of volumes in the target storage group or 59, whichever is less.

A data set's volume count is the maximum number of volumes to which the data set may extend. At any one time, there may be a mixture of primary volumes (volumes on which space is allocated for the data set) and candidate volumes (volumes on which space may be allocated at a future time). The total sum of the primary volumes and candidate volumes is the data set's volume count.

Note: When MAKEMULTI is specified and VOLCOUNT is also specified with an option other than VOLCOUNT(*), the VOLCOUNT option overrides MAKEMULTI.

MENTITY



MENTITY specifies, for RACF-protected data sets, an entity (*modeldsn*) and, optionally, the serial number of the volume containing that entity (*volser*). These keywords are used to define the data sets to RACF. Specification of MVOLSER is optional for one of the following:

- When the model entity (MENTITY) is cataloged in an integrated catalog facility catalog.
- When a non-VSAM data set is cataloged in the standard catalog search order.

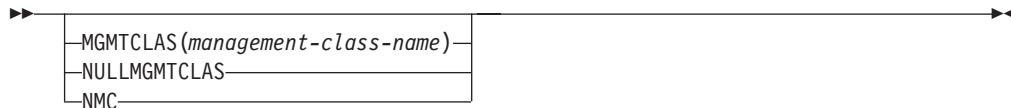
When MVOLSER is specified for a VSAM model entity, *volser* must be the volume serial number of the catalog in which the model entity is cataloged. If these keywords are not specified, DFSMSdss defines the data set to RACF by modeling the target profile after the source data set, if the source is discretely protected. Target data sets (preallocated or nonpreallocated) are RACF-protected only if the corresponding source data sets were so protected. If a source data set is protected with a generic profile, RACF generic profile checking must be activated prior to invoking the copy function.

Restriction: You cannot specify the MVOLSER(*volser*) keyword by itself. It can only be specified in conjunction with the MENTITY(*modeldsn*) keyword.

Related reading: For additional information about data security and data set profile considerations, See Chapter 6, “Data Security and Authorization Checking,” on page 255.

COPY Command

MGMTCLAS



MGMTCLAS specifies the user-desired management class that replaces the source management class as input to the ACS routines. You must have the proper RACF authority for the management class specified. The keyword itself does not require RACF authorization.

NULLMGMTCLAS/NMC specifies that the input to the ACS routines is a null management class rather than the source data set's management class.

MGMTCLAS and NULLMGMTCLAS are mutually exclusive; you cannot specify these keywords together.

Notes:

1. All SMS-managed data sets specified in the BYPASSACS keyword are assigned the specified management class because the ACS routines are not invoked. Non-SMS-managed data sets do not have a management class.
2. See "Assignment of Class Names by Using the RESTORE and COPY Commands" on page 219 for information on the assignment of class names using the copy function.

NOPACKING



NOPACKING specifies that DFSMSdss is to allocate the target data set only to the same or like device types as the source data set is allocated on and that DFSMSdss is to use track level I/O to perform data movement. This results in an exact track-for-track image of the source data set on the target volume.

dsn Specifies the fully or partially qualified names of a PDS to be processed.

NOPACKing is only valid with a PDS. If REBLOCK is specified, REBLOCK is ignored for the data set. If the data set is specified with CONVERT(PDSE()), NOPACKing is ignored for the data set.

A PDS copied or restored by using NOPACKing is not be compressed during data movement.

NOPACKing can be used for a damaged PDS that is currently usable by an application but would be made unusable by compression or other rearrangement of the physical layout of the data.

NOTIFYCONCURRENT

See "CONCURRENT" on page 131.

NULLMGMTCLAS

See "MGMTCLAS" on page 78.

NULLSTORCLAS

See "STORCLAS" on page 88.

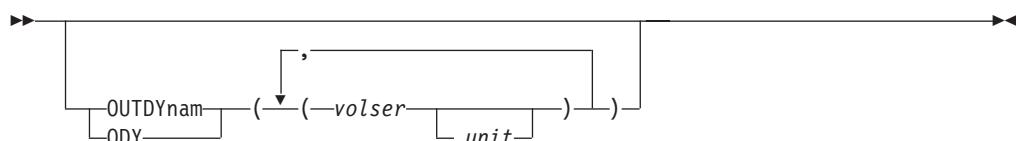
ONLYINCAT

See "INCAT" on page 73.

OUTDDNAME

ddn Specifies the name of the DD statement that identifies the output DASD volume. To assure correct processing, each of the DD statements corresponding to a DDNAME (*ddn*) must identify only one volume serial number. The volume serial number specified must be a valid DASD device; DD DUMMY is not supported. Only one volume is allowed for a full or tracks copy; one or more volumes are allowed for a data set copy operation. Multiple names in a data set copy must be separated by commas.

See the Note under OUTDYNAM for additional information.

OUTDYNAM

OUTDYNAM specifies that the output DASD volume is to be dynamically allocated. The volume must be mounted and online. You cannot specify a nonspecific volume serial number using an asterisk (*). Only one volume is allowed for a full or tracks copy; one or more volumes are allowed for a data set copy.

volser Specifies the volume serial number of the volume.

unit Specifies the device type of the volume. This parameter is optional.

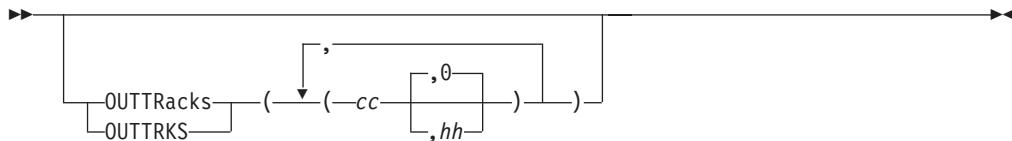
Notes for OUTDDNAME and OUTDYNAM Keywords:

1. DFSMSdss now distinguishes between non-SMS and SMS volumes specified in the OUTDDNAME or OUTDYNAM keywords. For non-SMS allocations, only the volumes that are non-SMS are considered for allocation. Similarly, only SMS volumes are considered for SMS allocations.
2. The above distinction is also used when determining the volume count for a multivolume allocation. Where volume count is determined from the number of specified volumes, only those volumes eligible for the type of allocation being done are counted. If there are no volumes that match the type of allocation

COPY Command

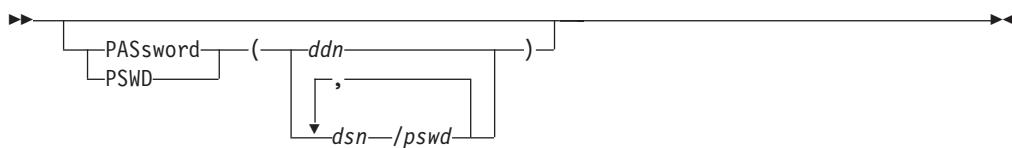
(SMS volumes for SMS allocation, or non-SMS volumes for non-SMS allocation), processing proceeds with a null volume list.

OUTTRACKS



OUTTRACKS specifies, for a tracks copy operation, the cylinder (cc) and head number (hh) on the output volume to which the tracks from the input volume are to be copied. If you do not specify OUTTRACKS operation, the tracks are copied to the same place on the output volume where they were on the input volume. The number of (cc,hh) combinations specified in the OUTTRACKS keyword must be the same as the number of (c1,h1,c2,h2) combinations specified in the TRACKS keyword.

PASSWORD



PASSWORD specifies the passwords that DFSMSdss is to use for password-protected data sets. (Password checking is bypassed for RACF-protected data sets.) This keyword is required only when:

- You do not have the required RACF DASDVOL or RACF data set access.
- The installation authorization exit does not bypass the checks.
- You do not want to be prompted for the password for VSAM data sets.

Note: You should specify the passwords for all data sets that do not have RACF protection but do have password protection. During processing, a utility invoked by DFSMSdss may have to prompt the operator for a password. You can control authorization checking by using the installation authorization exit.

Catalog passwords are not supported to facilitate disaster recovery operations, application data transfers, and data set migration. Catalog protection via an access control facility, such as RACF, is the preferred method of protection.

Passwords for a tracks copy operation are required only for the data sets on which the requested ranges fall.

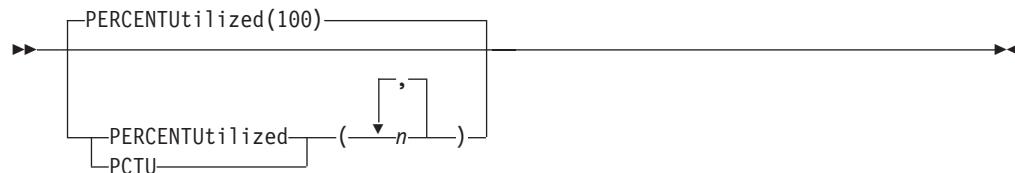
ddn Specifies the name of the DD statement that identifies the sequential data set, or member of a partitioned data set, that contains data set names and their passwords. This data set must contain card-image records in DFSMSdss command syntax format.

dsn/pswd *dsn* is a fully qualified data set name. *pswd* is its password. If no password follows the slash (/), *dsn* is treated as though it were *ddn*.

Printing of actual data set passwords specified in the input command stream is suppressed in the SYSPRINT output.

When a system utility is being used to perform the DFSMSdss copy operation, the user must supply the password for each password-protected data set selected or have the proper RACF data set access authority.

PERCENTUTILIZED

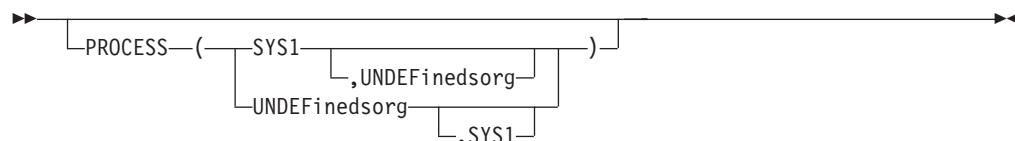


PERCENTUTILIZED specifies that DFSMSdss must stop allocating data sets to the target volumes when the allocated space reaches *n* percent of the total space on the target volume. The default value is 100. Specify more than one *n* if you have more than one target volume (for instance, a volume for overflow). If there are more target volumes than you have values in this keyword, the last value is used for the remaining target volumes. This keyword is used as a guide only and might not be precise for all situations.

Notes:

1. PERCENTUTILIZED is ignored when the target data set is preallocated.
2. PERCENTUTILIZED is not supported in an SMS environment.
3. PERCENTUTILIZED is ignored if no output volume is specified.

PROCESS



SYS1 specifies that DFSMSdss is to allow data sets with a high-level qualifier of SYS1 to be copied to a preallocated target and that SYS1 data sets can be deleted and uncataloged. SYS1.VVDS and SYS1.VTOCIX data sets cannot be copied, deleted, or uncataloged. To specify PROCESS(SYS1), RACF authorization may be required.

UNDEFinedsorg

PROCESS also specifies that DFSMSdss allow data sets with undefined data set organizations to be copied to an unlike target with a larger capacity. Refer to Table 2 on page 103 and Table 3 on page 104 for the action taken by DFSMSdss.

Note: Even though the data is being copied to a device with a larger track capacity, the data may not fit on the output device. For example, if the source device is a 3380, and the output device is a 3390 and the data set's block size is less than 277 bytes, a track on the target cannot contain as much data as a track on the source and the message ADR366W (Invalid Track Format) is issued.

COPY Command

Related reading: For additional information about RACF authorization, see the *z/OS DFSMSdss Storage Administration Guide*.

PURGE



PURGE specifies that unexpired data sets, which reside on the target volume, can be overlaid for a full or track copy operation. If you do not specify PURGE and unexpired data sets exist on the target volume, the copy operation fails.

For data set copy operations, PURGE specifies that unexpired source data sets can be deleted after they have been successfully copied. PURGE is only valid with the DELETE keyword.

Note: You must specify PURGE for a full volume copy operation if the VVDS name on the target volume does not match the target volume serial number (*volser*). This procedure applies to volumes that are created by using full volume copy in conjunction with one of the following conditions:

- When DUMPCONDITIONING is specified
- When you do not specify COPYVOLID or DUMPCONDITIONING

READIOPACING



READIOPACING specifies the pacing (that is, I/O delay) to be used for DFSMSdss DASD read channel programs. You can use this keyword to allow more time for other applications to complete I/O processing. DFSMSdss waits the specified time before issuing each channel program that reads from DASD.

nnn Specifies the amount of time in milliseconds. The maximum delay that can be specified is 999 milliseconds.

Notes:

1. If READIOPACING is not specified, there is no I/O delay.
2. The additional wait time does not apply to error recovery channel programs.
3. READIOPACING does not apply to concurrent copy I/O.

REBLOCK



REBLOCK specifies that DFSMSdss is to reblock one or more of the selected sequential or partitioned data sets.

dsn Specifies the fully or partially qualified names of a sequential or partitioned data set to be copied and reblocked.

COPY Command

The REBLOCK keyword is ignored for:

- Unmovable data sets
- Data sets with record format of U (except for partitioned load modules)
- Data sets with a record format of V, VS, VBS, or F
- Partitioned data sets with note lists (except for partitioned load modules)
- Partitioned data sets that are also specified in the NOPACKING keyword

Additionally, both the installation options exit and the installation reblock exit can override the specification of the REBLOCK keyword. The installation options exit can specify that no data set is to be reblocked. The installation reblock exit can specify whether a given data set is to be reblocked.

Some sequential and partitioned data sets have an attribute indicated in the VTOC making them capable of being reblocked. These data sets may be automatically reblocked by DFSMSdss independent of the REBLOCK keyword.

When copying partitioned load modules to an unlike device, DFSMSdss uses IEBCOPY with COPYMOD specified. This may result in a reblocked data set. When copying to a like device, IEBCOPY with COPY specified is used.

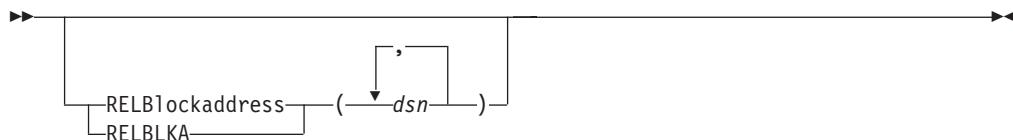
DFSMSdss uses the DASDCALC macro to determine the optimal block size for the target. The reblocking method used, DFSMSdss or DASDCALC, is presented to the installation reblock exit.

Related reading: For additional information about DFSMSdss processing of reblockable data sets, see the *z/OS DFSMSdss Storage Administration Guide*.

RECATALOG

See “CATALOG” on page 52.

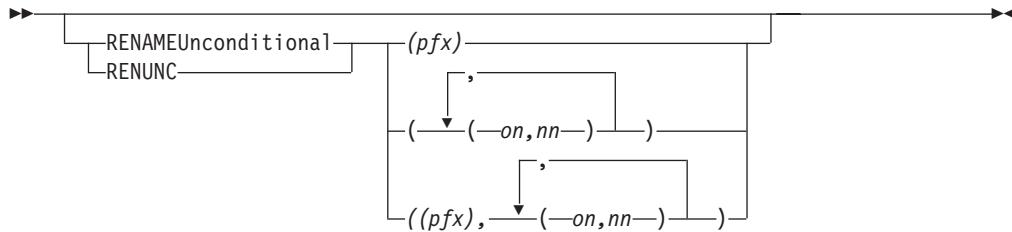
RELBLOCKADDRESS



RELBLOCKADDRESS identifies the direct access data sets whose names match the fully or partially qualified names specified (dsn). These direct access data sets are organized by relative block address instead of TTR and are to be copied block by block. DFSMSdss updates the block reference count (the relative position of the physical record as stored on its track) of dummy records. This keyword applies only to direct access data sets with fixed record formats and without standard user labels.

Restriction: If the data set is actually organized by TTR, the data set might become unusable.

RENAMEUNCONDITIONAL



RENAMEUNCONDITIONAL specifies that the data set must be copied with the new name, regardless of whether the data set exists on DASD with the old name. If the data set exists on the target volume with the new name and the **REPLACEUNCONDITIONAL** keyword is not specified, an error message is issued, and the data set is not copied.

- pfx** Specifies the prefix used to replace the first-level qualifier of the data set name. It is optional, but if specified, must be the first parameter in the list of subkeywords. The prefix is used only if the **(on,nn)** parameters are not specified or the old name filters do not match the data set name.
- on** Specifies the old name to be used as a filtering criterion to check if it matches the data set name.
- nn** Specifies the new name to be used to derive the new data set name when the data set name matches the corresponding old name filtering criterion.

The syntax rules for **pfx** (prefix), **on** (old name), and **nn** (new name) are the same as in the **RENAME** keyword in a restore operation.

Notes:

1. If the **RENAMEU** keyword is specified in conjunction with the **REPLACE** keyword, only one of the keywords take effect for any particular data set. The **RENAMEU** keyword takes precedence over the **REPLACE** keyword. If a source data set name matches the **RENAMEU** criteria, then rename processing will be performed and replace processing will not be performed. If a preallocated target data set exists with the new name as chosen by the rename criteria, then the copy fails even if the **REPLACE** keyword was specified. If you want to replace a preallocated target with the new name, specify the **REPLACEUNCONDITIONAL** keyword. If a source data set name does not match the rename criteria and a preallocated target data set with the source name exists, the preallocated target data set is replaced.
2. If **CICSVRBACKUP** is also specified, DFSMSdss uses the CICSVR-generated new name instead of the new name that you specified. See “**CICSVRBACKUP**” on page 54 for more information.
3. You can have up to 255 entries using **RENAMEUNCONDITIONAL**. You cannot go beyond this limit by trying to use **FILTERDD**. **FILTERDD** can not be used with **RENAMEUNCONDITIONAL**.

Related reading: For additional information about renaming, see “**RENAME**” on page 204 under the **RESTORE** command.

REPLACE

REPLACE specifies that DFSMSdss is to search the target volumes for usable preallocated data sets. If a usable preallocated target data set is found, it is replaced with the source data set. If no preallocated target is found, DFSMSdss attempts to allocate a data set.

DFSMSdss searches for preallocated data sets as follows:

- For SMS-managed data sets, DFSMSdss first searches in the standard order of search for a catalog entry for the data set.
- For VSAM data sets that are not SMS-managed, DFSMSdss searches the output volumes for preallocated data sets. If no output volumes are specified, DFSMSdss searches for a catalog entry for the data set.
- For Non-VSAM data sets that are not SMS-managed, DFSMSdss searches the output volumes for preallocated data sets. If no output volumes are specified, DFSMSdss searches for a catalog entry for the data set.
- If no preallocated target is found, DFSMSdss attempts to allocate a data set. DFSMSdss invokes the ACS routines to determine if the data set should be SMS managed. If it should be SMS managed, then allocation is done according to the SMS constructs. If the data set should not be SMS managed, the output volumes specified are used. If allocation is successful, the data set is copied.

Notes:

1. If REPLACE is specified with the COPY command, the SMS constructs already associated with the preallocated target data set remain the same.
2. CATALOG and RECATALOG are ignored for preallocated data sets.
3. If the source data set is an extended-addressable VSAM data set, then the target must also be an extended-addressable VSAM data set.
4. The target data set name must match the source data set name.
REPLACEUnconditional must be specified to replace a target data set that has a name matching the rename criteria.
5. The REPLACE and REPLACEUnconditional keywords can not be specified together.
6. If the RENAMEU keyword is specified in conjunction with the REPLACE keyword, only one of the keywords take effect for any particular data set. The RENAMEU keyword takes precedence over the REPLACE keyword. If a source data set name matches the RENAMEU criteria, then rename processing will be performed and replace processing will not be performed. If a preallocated target data set exists with the new name as chosen by the rename criteria, then the copy fails even if the REPLACE keyword was specified. If you want to replace a preallocated target with the new name, specify the REPLACEUNCONDITIONAL keyword. If a source data set name does not match the rename criteria and a preallocated target data set with the source name exists, the preallocated target data set is replaced.
7. If the source data set is a large format sequential data set, but the preallocated target is not a large format sequential data set, the preallocated target data set will be used and turned into a large format sequential data set as long as it has enough allocated space for the source data set.

REPLACEUNCONDITIONAL



REPLACEUNCONDITIONAL specifies that DFSMSdss is to search the target volumes for usable preallocated data sets. If a usable preallocated target data set is found, it IS replaced. When used with the RENAMEUnconditional keyword, usable preallocated data sets with the new name are replaced. When used without the RENAMEUnconditional keyword, usable preallocated data sets with the same name as the source data set are replaced. If no preallocated target is found, DFSMSdss attempts to allocate a data set. The REPLACE and REPLACEUnconditional keywords can not be specified together.

See the REPLACE keyword description for information on how target volume selection is performed.

Notes:

1. If REPLACEUNCONDITIONAL is specified with the COPY command, the SMS constructs already associated with the preallocated target data set remain the same. If the preallocated target data set is scratched and reallocated, the SMS constructs used are those returned by the ACS routines for the source data set name.
2. CATALOG and RECATALOG are ignored for preallocated data sets.
3. If the source data set is an extended-addressable VSAM data set, then the target must also be an extended-addressable VSAM data set.
4. If the source is a large format sequential data set, but the preallocated new name target is not a large format sequential data set, the preallocated target will be used and turned into a large format sequential data set as long as it has enough allocated space for the source data set. If the preallocated new name target does not have enough allocated space, it will be scratched and reallocated as a large format sequential data set with enough space for the source data set. If the name of the preallocated target is the same as the source data set, then see the REPLACE keyword.

SELECTMULTI

See “LOGINDDNAME” on page 74 and “LOGINDYNAM” on page 75.

SHARE



SHARE specifies that DFSMSdss is to share the data sets to be copied for read access with other programs.

SHARE and FULL are mutually exclusive; you cannot specify these keywords together.

Do not specify DELETE if you specify SHARE. You must have exclusive control over data sets that are to be deleted; SHARE does not require such exclusive control.

COPY Command

Note: Unlike the RESTORE command, the COPY command honors the SHARE keyword for VSAM data sets. However, the SHARE keyword is only honored for VSAM data sets that were defined with share options other than (1,3) or (1,4).

Specifying the SHARE keyword does not cause DFSMSdss to honor the share options that are defined for VSAM data sets. For VSAM data sets that are defined with share options other than (1,3) or (1,4), specifying the SHARE keyword allows other programs to obtain read access. It does not, however, allow write access to the data sets while they are being copied. For VSAM data sets that are defined with share options (1,3) or (1,4), neither read access nor write access by other programs is allowed while the data set is being copied.

SPHERE

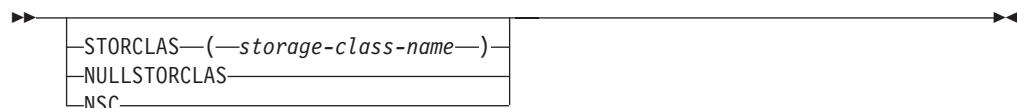


SPHERE specifies that, for any VSAM cluster copied, all associated AIX clusters and paths are to be copied. Individual names of sphere components do not need to be specified. Only the base cluster name is required. If output volumes are specified, the volumes on which the AIX clusters reside do not need to be specified.

Restrictions:

- If the sphere is specified but the base cluster name is not, DFSMSdss processes only those components of the sphere whose names are specified.
- Do not specify the SPHERE keyword with the CICSVRBACKUP keyword.

STORCLAS



STORCLAS specifies the storage class that you want to replace (the source storage class) as input to the ACS routines. You must have the proper RACF authorization for the specified storage class. The keyword itself does not require authorization.

NULLSTORCLAS/NSC specifies that the input to the ACS routines is to be a null storage class rather than the source data set's storage class.

STORCLAS and NULLSTORCLAS are mutually exclusive; you cannot specify both keywords simultaneously. See “Assignment of Class Names by Using the RESTORE and COPY Commands” on page 219 for information on assigning class names using the copy function.

Note: If BYPASSACS(dsn) is specified, all data sets that pass the BYPASSACS selection criteria are guaranteed the specified storage class. The combination of NULLSTORCLAS and BYPASSACS(dsn) forces the selected data sets to be non-SMS-managed.

STORGRP

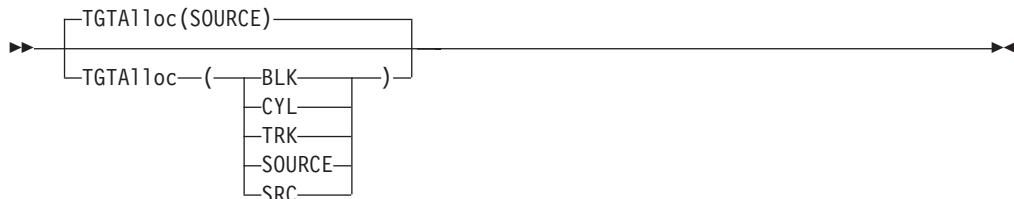
STORGRP specifies that all of the online volumes in the storage group be dynamically allocated. If a volume in the storage group is not online, that volume is not used for processing. Up to 255 storage group names may be specified. Specifying STORGRP with a storage group name is equivalent to specifying LOGINDYNAM with all the online volumes in the storage group included in the list.

You can specify the STORGRP keyword with the SELECTMULTI keyword, but STORGRP is mutually exclusive with the INDDname, INDYnam, LOGINDDname and LOGINDYnam keywords.

Notes for LOGINDDNAME, LOGINDYNAM, and STORGRP keywords:

1. DFSMSdss selects from all data sets cataloged in the catalogs accessible through the standard search order if you do not specify the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword.
2. When you specify the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword, DFSMSdss still uses the standard catalog search order. However, DFSMSdss selects data sets only from the specified volumes.
3. You must specify the SELECTMULTI keyword to copy a multivolume data set that has extents on volumes which are not identified with the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword.

See “LOGINDYNAM” on page 75 for a description of the SELECTMULTI keyword.

TGTALLOC

TGTALLOC specifies how DFSMSdss is to allocate the target data set.

BLK by blocks

CYL by cylinders

TRK by tracks

SOURCE/SRC

with the same space allocation type as that of the source data set

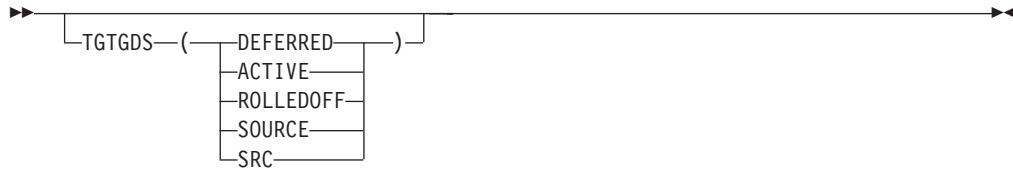
Notes:

1. If the TGTALLOC keyword is omitted, the target allocation defaults to SOURCE.

COPY Command

2. If SRC is specified and if the source data set is allocated by track or if TRK is specified, then the final VSAM allocation might be different from the requested one because of VSAM allocation rules.
3. If BLK is specified for VSAM data sets, TRK is used instead. The final VSAM allocation might be different from the requested one because of VSAM allocation rules.

TGTGDS



TGTGDS specifies in what status, during a data set operation, that DFSMSdss is to place nonpreallocated SMS-managed GDG data sets:

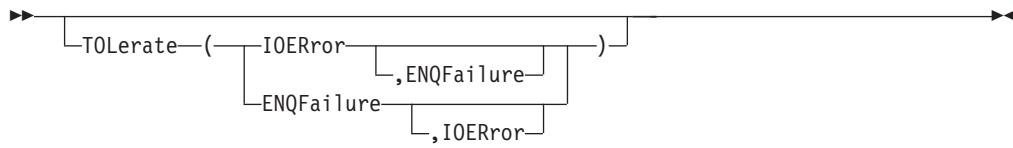
DEFERRED	Specifies that the target data set is to be assigned the DEFERRED status.
ACTIVE	Specifies that the target data set is to be assigned the ACTIVE status, for example, rolled into the GDG base.
ROLLEDOFF	Specifies that the target data set is to be assigned the rolled-off status.
SOURCE/SRC	Specifies that the target data set is to be assigned the same status as that of the source data set.

Notes:

1. If DELETE is specified without RENAMEUNCONDITIONAL and the source data set is an SMS-managed generation data set, the TGTGDS keyword is ignored and the source GDS status is copied to the target.
2. The requested target status of generation data sets must not violate generation data group rules.

Related reading: For additional information about the default status when TGTGDS is not specified, see the *z/OS DFSMSdss Storage Administration Guide*.

TOLERATE



TOLERATE specifies that DFSMSdss tolerates certain error conditions. For a copy operation (except of a user catalog or loadlib) in which a utility performs the copy, this keyword is ignored.

ENQFailure specifies that source and target data sets are to be processed even though shared or exclusive access fails.

Notes:

1. Unlike PDS data sets, PDSE data sets that are open for update cannot be copied even if TOL(ENQF) is specified.
2. If you must copy a PDSE data set and it must be open for update, convert the PDSE back to PDS and then copy the PDS data set with TOL(ENQF).
3. For a logical data set COPY command, TOL(ENQF) is ignored for HFS source data sets.

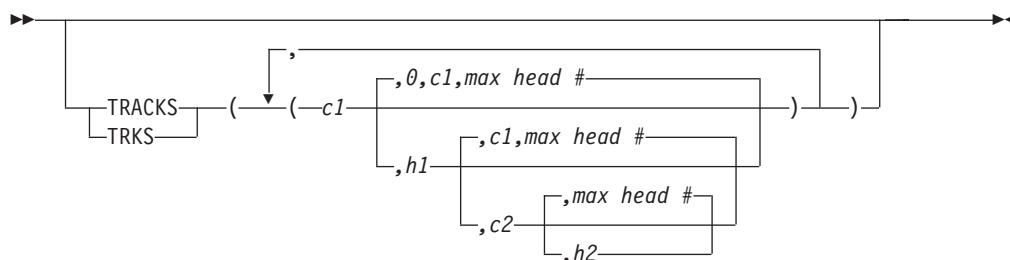
IOERROR

specifies that if the input volume can be opened, DFSMSdss is to continue copying even though permanent input errors (busout parity and equipment checks only) occur. DFSMSdss ends after 100 errors when this keyword is specified. The default ends on permanent input errors. On a data set copy in which a utility performs the copy, DFSMSdss ignores this keyword.

Notes:

1. TOL(IOERror) is ignored if CANcelerror is specified.
2. You cannot use the TOLERATE(ENQF) keyword when performing a logical copy operation with VSAM extended-format data sets.
3. You cannot use the TOLERATE(ENQF) keyword with a COPY FULL or COPY TRACKS operation.

Related reading: For additional information about using TOL(ENQF) keyword, see Appendix B, “Data Integrity—Serialization,” on page 287.

TRACKS

TRACKS specifies ranges of tracks to be copied (that is, a tracks copy).

c1,h1 Specifies the cylinder and head number of the beginning of the range. Specify hexadecimal numbers as X'c1' or X'h1'.

c2,h2 Specifies the cylinder and head number of the end of the range. Specify hexadecimal numbers as X'c2' or X'h2'. The *c2* must be greater than or equal to *c1*. If *c2* equals *c1*, *h2* must be greater than or equal to *h1*.

DFSMSdss verifies that the range is within the limits of the device. If you do not specify all four values for a range, DFSMSdss provides the missing values unless the omitted value causes a syntax error. No intervening values can be omitted. For example:

Specified Results

None	Syntax error
-------------	--------------

COPY Command

<i>c1</i>	c1,0,c1,maximum head number
<i>c1,h1</i>	c1,h1,c1,maximum head number
<i>c1,h1,c2</i>	c1,h1,c2,maximum head number
<i>c1,,c2</i>	Syntax error
<i>,h1</i>	Syntax error

Restriction: You cannot use the TRACKS keyword with the TOL(ENQF) keyword.

Related reading: For additional information about using the TRACKS keyword during physical processing, see the *z/OS DFSMSdss Storage Administration Guide*.

TTRADDRESS



TTRADDRESS identifies the direct access data sets whose names match the fully or partially qualified names specified (dsn). These data sets are organized by TTR rather than by relative block addressing and are to be processed track by track. The target device track capacity must be equal to or greater than the source.

Guideline: The TTRADDRESS keyword takes precedence over the AUTORELBLOCKADDRESS keyword processing for the specified data sets (dsn).

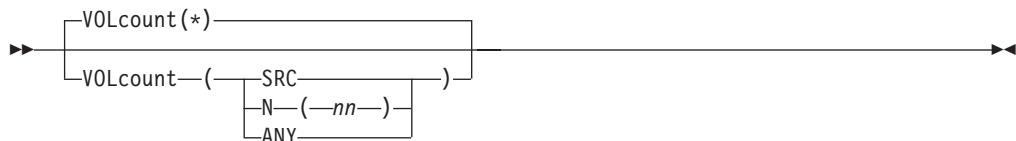
UNCATALOG



UNCATALOG specifies that DFSMSdss is to uncatalog but not scratch successfully copied non-VSAM data sets that are currently cataloged on the source volume. Any non-SMS, non-VSAM data set that has a high-level qualifier of SYS1 cannot be uncatalogued unless PROCESS(SYS1) is specified. UNCATALOG is ignored for VSAM data sets and SMS-managed non-VSAM data sets.

Note: Do not specify UNCATALOG with CONCURRENT, because after the concurrent copy operation has begun, the original data can still be updated.

VOLCOUNT



VOLCOUNT specifies the method DFSMSdss uses to determine the number of volumes (volume count) for allocating the SMS target data set for a copy operation of VSAM or non-VSAM data sets.

*** (asterisk)**

Specifies that DFSMSdss determine the volume count for allocation according to the following conditions:

- If the source data set is a single-volume data set, allocate one volume.
- The source data set is a multivolume data set, and one of the following conditions is present:
 - The OUTDDNAME or OUTDYNAM does not specify a list of volumes.
 - There are no SMS volumes in the list

DFSMSdss allocates the same number of volumes that were in the multivolume source data set.

- The source data set is a multivolume data set. It has an associated volume list (you specified the OUTDDNAME or OUTDYNAM keyword). DFSMSdss designates the volume count as the number of SMS volumes in the list.

DFSMSdss does not adjust the final number of candidate volumes after the allocation is complete.

The * (asterisk) is the default for this keyword.

SRC Specifies that DFSMSdss *rely on the source volume count* to determine the number of volumes to allocate for the target data set as follows:

- If no output volume list is specified, DFSMSdss allocates the same number of volumes that the source data set had.
- If a volume list is specified through OUTDDNAME or OUTDYNAM, the volumes in the list that are SMS-managed must be in the same storage group, and the allocation must be directed to that storage group.

DFSMSdss does not adjust the final number of candidate volumes after the allocation is complete.

N(nn) *nn* represents the number of volumes to be used for SMS data set allocation. Any value between 0 and 59 may be specified with the following conditions:

- If *nn* is not zero and a volume list is specified through OUTDDNAME or OUTDYNAM, DFSMSdss allocates either the number of SMS volumes in the volume list or *nn*, whichever is less.
- If *nn* is zero and a volume list is specified through OUTDDNAME or OUTDYNAM, DFSMSdss allocates either the number of SMS volumes in the volume list or the number of volumes that were allocated for the source data set, whichever is less.
- If a volume list is specified through OUTDDNAME or OUTDYNAM and there are no SMS volumes in the list, or there is no volume list, DFSMSdss allocates either the number of volumes used by the source data set or *nn*, whichever is more.

DFSMSdss does not adjust the final number of candidate volumes after the allocation is complete.

ANY Specifies that DFSMSdss *use a maximum volume count* to allocate the SMS target data set as follows:

- DFSMSdss initially sets a volume count of 59 for the allocation.
- If the data set is allocated on more volumes than were used to allocate the source data set, DFSMSdss reduces the number of volumes used to the number of primary volumes needed to satisfy the allocation.

COPY Command

- If the data set is allocated on the same number or fewer volumes than were used to allocate the source data set, DFSMSdss reduces the number of volumes used to the number of volumes used for allocation of the source data set.

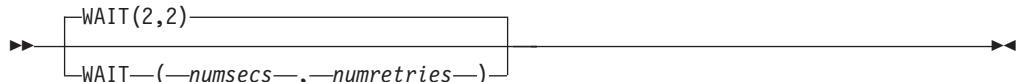
Notes:

1. VOLCOUNT does not convert any of the following data sets to multivolume: PDS or PDSE data sets, single-volume data sets whose organization is undefined, or empty non-VSAM, single-volume data sets.
2. VOLCOUNT does not change the number of volumes for keyrange KSDS data sets.
3. Guaranteed space is not honored when VOLCOUNT(ANY) is used.
4. VOLCOUNT(ANY) does not support keyed VSAM data sets that have an imbedded index. If VOLCOUNT(ANY) is specified and a data set has an imbedded index, the data set is processed as if VOLCOUNT(*) were specified.
5. VOLCOUNT(ANY) does not support any type of striped data set (physical, sequential, extended, or VSAM). If VOLCOUNT(ANY) is specified and a data set is striped, the data set is processed as if VOLCOUNT(*) were specified.
6. When you specify VOLCOUNT(ANY), the &ANYVOL and &ALLVOL read-only variables are not available to the storage group ACS routine.
7. For nonguaranteed-space, striped VSAM data sets: The minimum number of volumes that DFSMSdss allocates is determined by the number of stripes, which is based on the STORCLAS sustained data rate (SDR). DFSMSdss does not consider the number of volumes in the output volume list or any of the VOLCOUNT specifications. If there are not enough enabled volumes in the STORGRP to support the SDR, DFSMSdss reduces the number of stripes. If there are excess volumes specified, those volumes become nonspecific (*) candidates.
8. For guaranteed-space, striped VSAM data sets: DFSMSdss allocates the number of volumes that are specified in the output list, regardless of the SDR. (To be striped, the SDR must be greater than zero.) The VOLCOUNT rules described above apply.

You can override VOLCOUNT keyword settings with the options installation exit routine.

Related reading: For additional information about overriding VOLCOUNT keyword settings, see *z/OS DFSMS Installation Exits*.

WAIT



WAIT Specifies to DFSMSdss the length of the wait in seconds, and the number of passes to be made through the list of selected data sets to obtain control of a data set for the COPY DATASET command.

numsecs Specifies a decimal number (0–255) that designates the interval, in seconds, to wait before attempting another pass through the entire list of selected data sets.

numretries Specifies a decimal number (0–99) that designates the number of passes to make through the list of selected data sets in an attempt to obtain control of a data set.

The default for *numsecs,numretries* is (2,2), which specifies two retries at 2-second intervals. If you do not want to wait for a resource, specify 0 for either *numsecs* or *numretries*.

For a data set copy operation the WAIT keyword has a different meaning when: (1) data sets are being serialized, (2) multiple data sets are being processed, and (3) WAIT(0,0) is not specified. In this case, DFSMSdss makes multiple passes through the list of data sets. On each pass, DFSMSdss processes the data sets that (1) can be serialized without waiting for the resource and (2) were not processed before. At the end of a pass, if none of the data sets can be processed without waiting for a resource, then, in the next pass, at the first occurrence of a data set that was not processed, a WAIT is issued. That data set and the remainder of the list are processed if possible.

The above procedure is repeated until all data sets are processed or the WAIT limits are reached. For example, if WAIT(3,10) is specified and five data sets are left to be processed, up to ten passes are made. On each pass, an unprocessed data set is waited upon for 3 seconds. Thus, only a 30-second maximum is ever waited, not 150 (5 times 3 times 10).

Note: The WAIT keyword does not control wait/retry attempts for system resources (such as the VTOC and the VVDS). For system resources, the default wait time is 3 seconds and the default retry count is 30. This results in a total wait time of 90 seconds. Refer to *z/OS DFSMSdss Storage Administration Guide* for information about controlling the wait/retry attempts for system resources.

WRITECHECK



WRITECHECK specifies that the data copied is to be verified for successful completion. This keyword increases the overall elapsed time.

Notes:

1. On a data set copy in which a utility performs the copy, DFSMSdss ignores this keyword.
2. The WRITECHECK keyword is not supported for extended-format sequential data sets.

Data Integrity Considerations for Full or Tracks Copy Operation

For a full or tracks copy operation, DFSMSdss serializes the VTOC to preclude DADSM functions (such as ALLOCATE, EXTEND, RENAME, and SCRATCH) from changing the contents of the VTOC on the volume during the copy operation. Data sets are *not* serialized on these full or tracks operations. Therefore, some data

COPY Command

sets might be opened by other jobs during the copy, resulting in copies of partially updated data sets. You can minimize this possibility by performing the copy when there is low system activity.

Full data integrity can only be guaranteed by performing copy operations by data set when TOL(ENQF) or SHARE are not specified.

Examples of Full and Tracks Copy Operations

The following examples are for FULL and tracks COPY operations.

Example 1: Data Set Copy Operation

```
//JOB1    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSU
//SYSPRINT DD SYSOUT=A
//DASD1    DD UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//DASD2    DD UNIT=3380,VOL=(PRIVATE,SER=222222),DISP=OLD
//SYSIN    DD *
      command input (see Examples 1A and 1B below)
/*
```

Example 1A: A Full Copy Operation

```
COPY INDDNAME(DASD1) OUTDDNAME(DASD2) -
      ALLEXCP CANCELERROR COPYVOLID
```

Example 1B: A Tracks Copy Operation

```
COPY TRACKS(1,0,1,15) INDDNAME(DASD1) -
      OUTDDNAME(DASD2) CANCELERROR
```

The data from DASD volume 111111 is to be copied to DASD volume 222222. For the full copy operation (example 1A), all allocated space in sequential or partitioned data sets, and in data sets with a data set organization that is null, is copied (ALLEXCP(*)). The volume serial number (VOLID) of the source volume will be copied to the target volume. The result is that both volumes will have the same serial number (111111).

The preceding applies only to data sets that are not empty. For data sets that are empty, DFSMSdss copies all data within the allocated space (ALLEXCP). The copy operation is to be ended if a permanent read error occurs (CANCELERROR).

Example 2: A Tracks Copy with Track Relocation

The following example shows a tracks copy operation in which the contents of cylinder 1, tracks 0 through 14, on source volume 338000 are copied to cylinder 3, tracks 0 through 14, on target volume 338001. The operation stops if a permanent error occurs on the source volume (CANCELERROR). The data written to the target volume is to be verified (WRITECHECK).

```
//JOB2      JOB accounting information,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
      COPY TRACKS(1,0,1,14) /* SOURCE TRACKS */ -
          OUTTRACKS(3,0)    /* TARGET TRACKS */ -
          INDYNAM(338000)   /* ALLOC VOL 338000 DYNAMICALLY */ -
          OUTDYNAM(338001)  /* ALLOC VOL 338001 DYNAMICALLY */ -
          CANCELERROR       /* STOP ON INPUT ERROR */ -
          WRITECHECK        /* VERIFY DATA WRITTEN TO OUT VOL */
/*

```

Examples of Data Set Copy Operations

The following examples are for data set copy operations.

Example 1: A Data Set Move—Only Single Volume Data Sets

```
//JOB3      JOB accounting information,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
      COPY DATASET(
          INCLUDE(USER1.**)           /* FILTER ON DS W/1ST LEV Q USER1 */ -
          BY(MULTI,=,NO)              /* FILTER ON SINGLE VOLUME */ -
          INDYNAM (338000,338002)    /* ALLOC VOL 338000, 338002 DYNAMICALLY */ -
          OUTDYNAM(338001)           /* ALLOC VOL 338001 DYNAMICALLY */ -
          DELETE
/*

```

Example 1 shows a data set copy operation in which all single-volume data sets with the first-level qualifier USER1 on the source volumes that are labeled 338000 and 338002 are copied to the target volume that is labeled 338001. All source data sets that are selected and successfully processed are deleted. The copied non-SMS, non-VSAM data sets are not cataloged.

Example 2: A Data Set Copy to Move Data Sets to a Single Volume—Device Conversion

```
//JOB4      JOB accounting information,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSU
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
      COPY DATASET(
          INCLUDE(USER1.**)) /* FILTER ON DS W/1ST LEV Q USER1 */ -
          OUTDYNAM(338001)   /* ALLOC VOL 338001 DYNAMICALLY */ -
          DELETE CATALOG FORCE -
          TGTALLOC(SOURCE)
/*

```

Example 2 shows a data set copy operation in which all cataloged data sets with a first-level qualifier of USER1 (USER1.**) are consolidated on a single target volume that is labeled 338001. The data sets can reside on multiple source volumes. The target volume can differ in device type from other volumes on which the source data sets reside. A few data sets might already reside on volume 338001. Data sets are cataloged in the standard search order. Expired source data sets are scratched and uncataloged after they are successfully moved to volume 338001. On volume 338001, they have the same allocation type (BLK, TRK, or CYL) that they had on the source volumes. FORCE is specified to include unmovable data sets.

Example 3: A Data Set Copy of a Multivolume Data Set

```
//JOB5      JOB accounting information,REGION=nnnnK
//STEP1     EXEC PGM=ADRDSU
//IVOL1     DD UNIT=(SYSDA,2),VOL=SER=(VOL111,VOL222),DISP=SHR
//IVOL2     DD UNIT=SYSDA,VOL=SER=VOL222,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSIN     DD *
COPY DATASET(      -
INC(USER.MULTI.VOLUME1)) /* SELECT THIS DATA SET      */ -
INDD(IVOL1,IVOL2)        /* IDENTIFY INPUT VOLUMES */ -
OUTDYNAM((338001),(338002),(338003)) /* DYNAM ALLOC VOLS */ -
PCTU(80,80,80)           /* PERCENTUTIL = 80 PERCENT */ -
RECATALOG(USERCAT2)
/*
```

Example 3 shows a data set copy operation in which a multivolume data set is copied to a set of target volumes labeled 338001, 338002, and 338003. The source data set is not deleted. The copied data set is cataloged in a new catalog, USERCAT2. The data set currently exists on multiple source volumes. Multiple output volumes are specified for overflow purposes. However, the output cannot be on more volumes than the source data set. These target volumes might already have data sets that reside on them. Space is left on these volumes to allow expansion of the data sets that remain on the volumes.

To include SELECTMULTI processing, you can change Example 3 as shown later in this section. The INCLUDE keyword specifies that you want to select all data sets on input volumes VOL111 and VOL222. The SELECTMULTI(ANY) keyword specifies that a cataloged data set that resides on volumes VOL111, VOL444, and VOL555 is copied even though VOL444 and VOL555 are omitted from the LOGINDD volume list.

```
//JOB5      JOB accounting information,REGION=nnnnK
//STEP1     EXEC PGM=ADRDSU
//IVOL1     DD UNIT=(SYSDA,2),VOL=SER=(VOL111,VOL222),DISP=SHR
//IVOL2     DD UNIT=SYSDA,VOL=SER=VOL222,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSIN     DD *
COPY DATASET(      -
INC(***)          /* SELECT ALL DATA SETS      */ -
LOGINDD(IVOL1,IVOL2) /* IDENTIFY INPUT VOLUMES */ -
OUTDYNAM((338001),(338002),(338003)) /* DYNAM ALLOC VOLS */ -
SELECTMULTI(ANY)    /* PROCESS MISSING VOLUMES */ -
PCTU(80,80,80)      /* PERCENTUTIL = 80 PERCENT */ -
RECATALOG(USERCAT2)
/*
```

Example 4: A Data Set Copy with DELETE and RENAMEU Options

```
//JOB6      JOB accounting information,REGION=nnnnK
//STEP1     EXEC PGM=ADRDSU
//SYSPRINT DD SYSOUT=A
//SYSIN     DD *
COPY DATASET(
  INCLUDE(USER1.*) /* FILTER ON DS W/1ST LEV Q USER1 */ -
  OUTDYNAM((338001),(338002),(338003)) /* DYNAM ALLOC VOLs */ -
  DELETE -
  RENAMEU(USER2) -
  RECATALOG(USERCAT2)
/*

```

Example 4 shows a data set copy operation. All the data sets with the high-level qualifier USER1 that are in the standard search order are copied to the target volumes that are labeled 338001, 338002, and 338003. The copied data sets are renamed to the high-level qualifier USER2, followed by the second through last qualifiers of the old names. If data sets with the same name as the new names are on the target volumes or if the data sets are already cataloged in USERCAT2, they are not copied. The copied, expired data sets are deleted from the source volumes, uncataloged, and recataloged in the USERCAT2 catalog . This process moves the data sets from one set of volumes to another set of volumes, from one catalog to another catalog, and renames them.

Example 5: A Data Set Copy With REBLOCK Option

```
//JOB7      JOB accounting information,REGION=nnnnK
//STEP1     EXEC PGM=ADRDSU,PARM='UTILMSG=YES'
//SYSPRINT DD SYSOUT=A
//DISK      DD UNIT=3380,VOL=(PRIVATE,,,SER=338001),DISP=SHR
//DISK2     DD UNIT=3390,VOL=(PRIVATE,,,SER=339001),DISP=SHR
//SYSIN     DD *
COPY DATASET(
  INCLUDE(***) /* INCLUDE ALL DATA SETS */ -
  INDDNAME(DISK) /* INPUT VOLUME */ -
  OUTDDNAME(DISK2) /* OUTPUT VOLUME */ -
  REBLOCK(**.USER1.*)
/*

```

Example 5 shows a data set copy operation in which all data sets on the 3380 source volume that is labeled 338001 are copied to the 3390 target volume that is labeled 339001. If data sets with the same name are on the target volume, they are not copied. The sequential and partitioned data sets that meet the filtering criteria that is specified in the REBLOCK keyword are reblocked on the target volume. The block size is selected by DFSMSdss, unless it is modified by the user reblock exit routine.

COPY Command

Example 6: A Data Set Copy to a Preallocated Target Data Set

```
//JOB8      JOB accounting information,REGION=nnnnK
//STEP1      EXEC PGM=ADRDSU
//SYSPRINT  DD SYSOUT=A
//DASD1      DD UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//DASD2      DD UNIT=3380,VOL=(PRIVATE,SER=222222),DISP=OLD
//SYSIN      DD *
      COPY DATASET(   -
                      INCLUDE(USER.TEST.DATA) ) -
                      INDDNAME(DASD1) -
                      OUTDDNAME(DASD2) -
                      REPLACE -
                      DELETE
/*

```

Example 6 shows a data set copy in which a source data set (USER.TEST.DATA) allocated on volume 111111 and cataloged in catalog USERCAT is copied to a preallocated target data set (with the same name as the source) on volume 222222. The REPLACE keyword specifies that you want DFSMSdss to search the target volume for a usable preallocated data set. The data set is deleted from the source volume, if it is expired.

Example 7: Using the COPY Command to Convert to SMS

Example 7 shows non-SMS-managed volumes that are converted to SMS-managed volumes. It is a two-step process.

```
//JOB9      JOB accounting information,REGION=nnnnK
//STEP1      EXEC PGM=ADRDSU
//SYSPRINT  DD SYSOUT=*
//SYSIN      DD *
      COPY -
          DS(INC(**)) -
          LOGINDYNAM ( -
                      (338001) -
                      (338002) -
          ) -
          STORCLAS(DB2PERF) -
          MGMTCLAS(DBBACKUP) -
          BYPASSACS(**) -
          DELETE -
          PURGE
/*

```

In step 1 of Example 7, all data sets on the non-SMS-managed volumes 338001 and 338002 are copied to SMS-managed volumes on the system. DELETE and PURGE processing is used to avoid duplicate catalog entries. ACS routines are not invoked to determine the target data set classes for this copy operation. Instead, users provide storage and management classes with the STORCLAS and MGMTCLAS keywords. In addition, users can suppress calls made to the ACS routines with the BYPASSACS(**) keyword. All data sets that are supported by SMS are given these new storage and management classes. All data sets that cannot be SMS-managed (for example, unmovable data sets) are not copied.

```
//JOB9      JOB accounting information,REGION=nnnnK
//STEP2      EXEC PGM=ADRDSU
//SYSPRINT  DD  SYSOUT=*
//SYSIN      DD  *
COPY -
  DS(INC(**)) -
    LOGINDYNAM ( -
      (338001) -
      (338002) -
    ) -
    RENUNC(AUG0387)
/*

```

In step 2 of Example 7, all data sets on the non-SMS-managed volumes 338001 and 338002 are copied to SMS-managed volumes on the system. The RENUNC command is used to avoid duplicate catalog entries. The ACS routines select a target storage and management class for each data set. Those data sets that cannot be SMS-managed (storage class ACS routine returns a null storage class) are not copied because no output volume is specified. Each data set that is copied is given a new high-level qualifier (AUG0387) and automatically cataloged.

Example 8: A Data Set Copy Using CONVERT PDSE

```
//JOB2      JOB accounting information,REGION=nnnnK
//STEP1      EXEC PGM=ADRDSU
//SYSPRINT  DD  SYSOUT=*
//SYSIN      DD  *
COPY -
  DS(INC(USER.PDS,**)) -
    LOGINDYNAM ( -
      (338001) -
      (338002) -
    ) -
    CONVERT (PDSE(**)) -
    RENUNC (USER.PDS.**, USER.PDSE.**)
/*

```

Example 8 shows all data sets with the first two qualifiers of USER.PDS on non-SMS-managed volumes 338001 and 338002 are copied to SMS-managed volumes on the system. CONVERT PDSE is used to convert data sets to PDSE. The RENUNC keyword is used to avoid duplicate catalog entries. The ACS routines select a target storage and management class for each data set. Each data set that is copied and converted is given a new secondary qualifier (PDSE) and automatically cataloged.

Example 9: A Data Set Copy with CONCURRENT

```
//DSSJOB JOB accounting information,REGION=nnnnK
//COPYSTEP EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=*
//SYSIN      DD  *
COPY DATASET(INCLUDE(USER.LOG,USER.TABLE,USER.XREF)) -
  OUTDYNNAME(OVOL01,OVOL02,OVOL03,OVOL08) -
  ALLDATA(*) ALLEXCP CONCURRENT -
  STORCLAS(BACKUP) RENAMEUNCONDITIONAL(USERX)
/*

```

COPY Command

Example 9 shows the required JCL for DFSMSdss to perform a logical data set copy using the concurrent copy feature. This job continues (with a warning message) if concurrent copy initialization fails.

Example 10: Copying a HFS using logical COPY

```
//DSSJOB JOB accounting information,REGION=nnnnK
//COPYSTEP EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
COPY DATASET(INCLUDE(OMVS.SB.MVS090.JV390.HFS, -
    OMVS.SB.MVS090.XML.HFS)) -
    RENAMEU((OMVS.SB.MVS090.JV390.HFS, -
    OMVS.SB.MVS030.ROOT.HFS) -
    (OMVS.SB.MVS090.XML.HFS, -
    OMVS.SB.MVS030.XML.HFS)) -
    NULLSTORCLAS BYPASSACS(**) -
    ALLDATA(*) ALLEXCP CANCELERROR -
    LOGINDDNAME(MVS091) OUTDDNAME(MVS023) -
    SHARE -
    WRITECHECK
/*
*/
```

Example 10 shows the required JCL for DFSMSdss to perform a logical copy of a HFS.

ALLDATA and ALLEXCP Interactions

Table 2 on page 103 and Table 3 on page 104 describe the functions of the ALLDATA and ALLEXCP keywords during a data set copy to LIKE and UNLIKE devices, respectively.

Table 2. ALldata and ALLEXCP Interactions When Copying to LIKE Device. Read the first eleven columns to find the row that matches your situation. Read the last column to find what DFSMSdss does.

Y	Yes
N	No
X	Either
-	Not Applicable
1	Allocate and copy all the allocated space
2	Allocate and copy only the used space
3	Allocate and copy only one track
4	Allocate all the allocated space, and copy only the used space
5	Allocate all the allocated space, and copy only one track
6	Do not process the data set

Empty data set?	EOF as first record?	Undefined DSORG?	Sequential data set? (a)	Partitioned data set? (b)	Load module?	ALldata(*) used?	ALldata(dsn) used?	Allexcp used?	NOPACKING used?	REBLOCK used?	DFSMSdss action is:
X	X	Y	-	-	-	X	X	X	-	X	1
N	X	-	Y	-	-	N	N	X	-	X	2
N	X	-	Y	-	-	Y	-	X	-	N	1
N	X	-	Y	-	-	Y	-	X	-	Y	4
N	X	-	Y	-	-	-	Y	X	-	N	1
N	X	-	Y	-	-	-	Y	X	-	Y	4
N	X	-	-	Y	X	N	N	X	N	X	2
N	X	-	-	Y	X	N	N	N	Y	X	1
N	X	-	-	Y	N	Y	-	X	N	X	4
N	X	-	-	Y	N	-	Y	X	N	X	4
N	X	-	-	Y	Y	Y	-	X	N	N	1
N	X	-	-	Y	Y	Y	-	X	N	Y	4
N	X	-	-	Y	Y	-	Y	X	N	N	1
N	X	-	-	Y	Y	-	Y	X	N	Y	4
Y	N	-	Y	-	-	X	X	N	-	X	6
Y	N	-	Y	-	-	X	X	Y	-	N	1
Y	N	-	Y	-	-	X	X	Y	-	Y	6
Y	Y	-	Y	-	-	X	X	Y	-	N	1
Y	Y	-	Y	-	-	X	X	Y	-	Y	3
Y	Y	-	Y	-	-	N	N	N	-	X	5
Y	Y	-	Y	-	-	Y	-	N	-	X	3
Y	Y	-	Y	-	-	-	Y	N	-	X	3

Notes:

- (1) When ALldata or Allexcp is specified for a sequential extended format data set, data beyond the last used block pointer is not retained. The target data set is allocated with the same amount of space as the source data set during a logical restore or copy operation.
- (2) Partitioned data sets that have directories (whether or not there are empty members in the directory) are treated as not empty.

COPY Command

Table 3. ALldata and ALLEXCP Interactions When Copying to UNLIKE Device. Read the first twelve columns to find the row that matches your situation. Read the last column to determine what DFSMSdss does.

Y	Yes
N	No
X	Either
-	Not Applicable
1	Allocate the same number of tracks as the source, and make a track image copy.
2	Allocate and copy only the used space
3	Allocate and copy only one track
4	Allocate all the allocated space, and copy only the used space
5	Allocate all the allocated space, and copy only one track
6	Do not process the data set

Empty data set?	EOF as first record?	BLKSIZE=0 data set?	Undefined DSORG?	Sequential data set? (a)	Partitioned data set? (b)	Load module?	Target tracksize > source?	PROCESS (UNDEF) used?	ALldata(*) used?	ALldata(dsn) used?	ALLEXCP used?	DFSMSdss action is:
X	X	X	Y	-	-	-	X	N	X	X	X	6
X	X	X	Y	-	-	-	N	Y	X	X	X	6
X	X	X	Y	-	-	-	Y	Y	X	X	X	1
N	X	N	-	Y	-	-	X	-	N	N	N	2
N	X	N	-	Y	-	-	X	-	Y	-	X	4
N	X	N	-	Y	-	-	X	-	-	Y	X	4
N	X	Y	-	Y	-	-	X	-	X	X	X	6
N	X	X	-	-	Y	N	X	-	N	N	N	2
N	X	X	-	-	Y	N	X	-	Y	-	X	4
N	X	X	-	-	Y	N	X	-	-	Y	X	4
N	X	N	-	-	Y	Y	X	-	N	N	X	2
N	X	N	-	-	Y	Y	X	-	Y	-	X	4
N	X	N	-	-	Y	Y	X	-	-	Y	X	4
N	X	Y	-	-	Y	Y	X	-	X	X	X	6
Y	N	X	-	Y	-	-	X	-	X	X	X	6
Y	Y	N	-	Y	-	-	X	-	N	N	N	3
Y	Y	N	-	Y	-	-	X	-	N	N	Y	5
Y	Y	N	-	Y	-	-	X	-	Y	-	N	3
Y	Y	N	-	Y	-	-	X	-	-	Y	X	5
Y	Y	Y	-	Y	-	-	X	-	X	X	X	6

Notes:

- (a) When ALldata or ALLEXCP is specified for an extended-sequential data set, data beyond the last used block pointer is not retained. The target data set is allocated with the same amount of space as the source data set during a logical restore or copy operation.
- (b) Partitioned data sets that have directories (whether or not there are empty members in the directory) are treated as not empty.

COPYDUMP Command

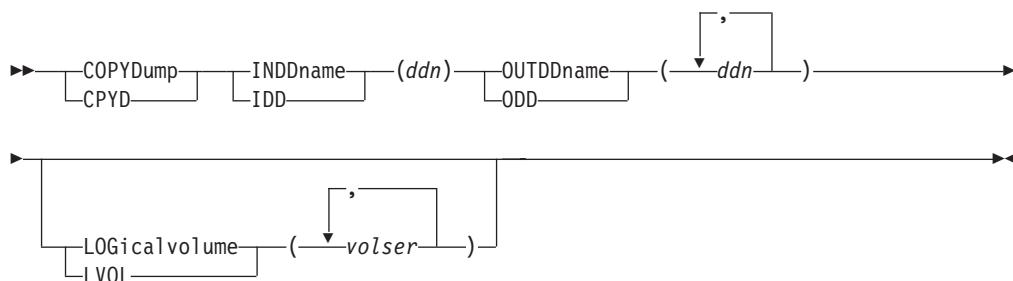
With the COPYDUMP command, you can make from 1 to 255 copies of DFSMSdss-produced dump data. The data to be copied, a sequential data set, can be on a tape or a DASD volume, and copies can be written to a tape or a DASD volume. If the dump data is produced from multiple DASD volumes by using a physical data set dump operation, you can selectively copy the data from one or more of those volumes.

The COPYDUMP command cannot change the block size of the DFSMSdss dump data set. If you are copying a dump data set to a DASD device, the source block size must be small enough to fit on the target device.

Notes:

1. Extra dump tapes can be used for such things as disaster recovery backup or distribution of dumped data (for example, a newly generated system).
2. COPYDUMP is the only supported method for copying DFSMSdss dump data sets. Using a copy produced by any other method or utility as input to a RESTORE operation can produce unpredictable results.

COPYDUMP Syntax



Explanation of COPYDUMP Command Keywords

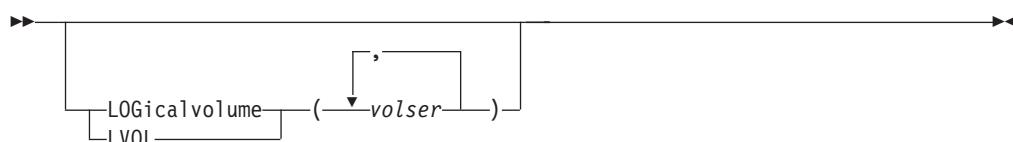
This section describes the keywords for the COPYDUMP command.

INDDNAME



ddn Specifies the name of the DD statement that identifies the sequential data set to be copied. This data set can reside on one or more tapes, or on DASD volumes.

LOGICALVOLUME



COPYDUMP Command

volser Specifies the source DASD volume serial number from which dumped data is to be copied. Omission of the LOGICALVOLUME keyword causes DFSMSdss to copy data from all logical volumes in the dump data set. This keyword is useful only if the data being copied was created by a physical data set dump operation from multiple DASD volumes. When copying a *logical* dump, LOGICALVOLUME is ignored.

OUTDDNAME



ddn Specifies the name of the DD statement that identifies the output sequential data set. This data set can be on a tape or a DASD volume.

Examples of COPYDUMP Operations

The following are examples of the COPYDUMP command.

Example 1: Making Two Copies of a Dump

```
//JOB1    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=A
//BACKUP  DD  UNIT=3480,VOL=SER=TAPE05,DISP=OLD,
//  DSNAME=V111111.BACKUP
//COPY1   DD  UNIT=3480,VOL=SER=TAPE06,
//  DISP=(NEW,CATLG),DSNAME=V111111.BACKUP1
//COPY2   DD  UNIT=3480,VOL=SER=TAPE07,
//  DISP=(NEW,CATLG),DSNAME=V111111.BACKUP2
//SYSIN   DD  *
      COPYDUMP -
      INDD(BACKUP) -
      OUTDD(COPY1,COPY2)
/*
```

In this example, two copies are to be made from a DFSMSdss dump tape (OUTDD(COPY1,COPY2)).

Example 2: Copying a Dump Created by Using Physical Data Set Processing

```
//JOB2    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=A
//TAPE2   DD  UNIT=3480,VOL=SER=TAPE20,
//  LABEL=(1,SL),DISP=(OLD,KEEP),DSN=USER.BACKUP.REL3A
//OUTT2   DD  UNIT=3480,VOL=SER=TAPE21,
//  LABEL=(1,SL),DISP=(NEW,CATLG),DSN=USER.BACKUP.REL3A.A
//SYSIN   DD  *
      COPYDUMP -
      INDD(TAPE2)          /* DUMP TAPE TO BE COPIED    */ -
      OUTDD(OUTT2)          /* NEW DUMP TAPE           */ -
      LVOL(338001)          /* SER NO OF VOL TO BE COPIED */-
/*
```

Assume that a *physical* data set dump operation was used to create a dump tape, volume TAPE20. Also assume that source DASD volumes 338000, 338001, and so

on were specified, resulting in VTOCs being used for data set selection. Only the dump data from DASD volume 338001 is to be copied (LVOL(338001)).

Example 3: Copying a Dump Created by Using Logical Data Set Processing

```
//JOB3      JOB    accounting information,REGION=nnnnK
//STEP1     EXEC   PGM=ADRDSSU
//SYSPRINT DD     SYSOUT=A
//TAPE3     DD     UNIT=3480,VOL=SER=TAPE30,
//          LABEL=(1,SL),DISP=(OLD,KEEP),DSN=USER.BACKUP.REL3B
//OUTT3     DD     UNIT=3480,VOL=SER=TAPE31,
//          LABEL=(1,SL),DISP=(NEW,CATLG),DSN=USER.BACKUP.REL3B.A
//SYSIN     DD     *
          COPYDUMP -
          INDD(TAPE3)           /* DUMP TAPE TO BE COPIED */ -
          OUTDD(OUTT3)           /* NEW DUMP TAPE */ -
/*
/*
```

Assume that a *logical* data set dump operation was used to create a dump tape, volume TAPE30. All dump data is copied.

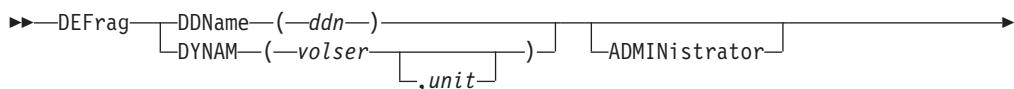
DEFrag Command

When you issue the DEFrag command, DFSMSdss relocates data set extents on a DASD volume to reduce or eliminate free space fragmentation. A summary report is printed that lists the before and after statistics of the volume. The report includes the fragmentation index, the size of the largest free space extent, and so forth. You can specify which data sets, if any, should be excluded from relocation. Before the free space defragmentation is performed for the volume, specify that DFSMSdss combine the extents of any data sets on the volume that have multiple extents.

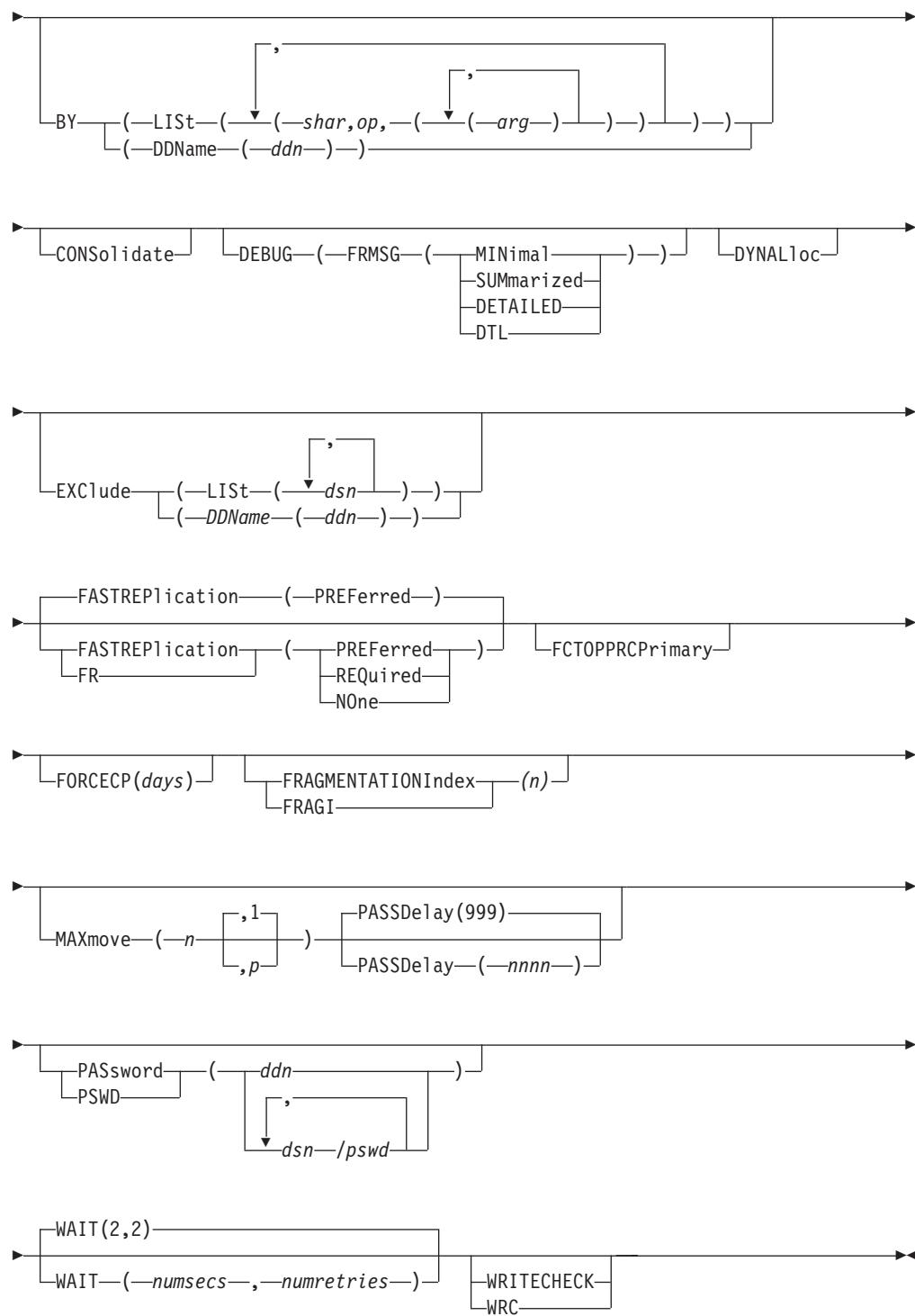
The amount of time it takes for a DEFrag operation to complete is determined by the size and fragmentation of the volume being processed. Larger volumes or more fragmented volumes take longer to complete.

Attention: Canceling the DEFrag command is strongly discouraged. Canceling an in-process DEFrag can damage data in numerous and unpredictable ways. Before initiating the DEFrag command, consider how long the operation will take by evaluating the size and the fragmentation of the volume being processed.

DEFrag Syntax



DEFrag Command



Explanation of DEFrag Command Keywords

This section describes the keywords for the DEFrag command.

ADMINISTRATOR

Administrator

ADMINISTRATOR lets you act as a DFSMSdss-authorized storage administrator for the DEFrag command. If you are not authorized to use the ADMINISTRATOR keyword, the command is ended with an error message. Otherwise, access checking to data sets and catalogs is bypassed.

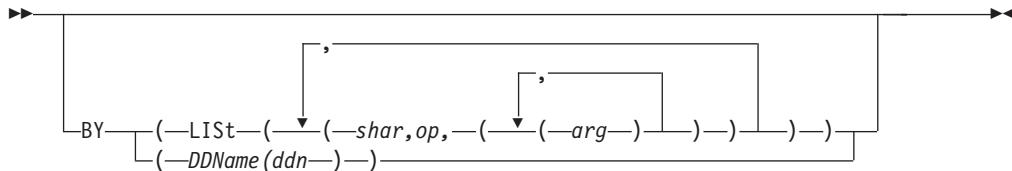
DEFFRAG Command

To use the ADMINISTRATOR keyword all of the following must be true:

- FACILITY class is active.
- The applicable FACILITY-class profile is defined.
- You have READ access to that profile.

Related reading: For additional information about using the ADMINISTRATOR keyword see “ADMINISTRATOR Keyword” on page 265.

BY



BY specifies data set filtering criteria.

DDNAME(ddn)

Specifies the name of the data definition (DD) statement that identifies a sequential data set or member of a partitioned data set that contains the filtering criteria to use. This is in the form of card-image records, in DFSMSdss command syntax, that contain the BY keywords that are described below.

LIST((schar,op,((arg))))

Specifies data set filtering. To select the data set for inclusion in the DEFrag operation, *all* BY criteria must be met. See “Filtering by Data Set Characteristics” on page 14 for a full discussion of *schar*, *op*, and *arg*.

CONSOLIDATE



CONSOLIDATE specifies that DEFrag perform extent reduction by combining the extents of data sets with multiple extents, if possible.

When CONSOLIDATE is specified, DFSMSdss consolidates data set extents where possible and then continues with normal free space defragmentation processing.

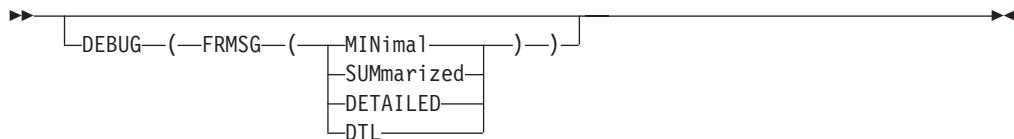
Attention: The process of combining data set extents can cause the freespace to be more fragmented than it was before the operation began. And, although DFSMSdss performs freespace defragmentation following the consolidation of data set extents, there is a possibility that the fragmentation index may be higher following a defrag operation with CONSOLIDATE specified, than before the operation began.

DDNAME



ddn Specifies the name of the DD statement that describes the volume to be processed.

DEBUG



FRMSG is a subkeyword for the DEBUG keyword. DEBUG(FRMSG(MINIMAL | SUMMARIZED | DETAILED)) specifies that DFSMSdss issue an informational message about why one of the fast replication methods cannot be used during a DEFFRAG operation. DEBUG(FRMSG) cannot be specified by itself. You must use one of the subkeywords (MINimal | SUMmarized | DETAILED) when you designate this keyword.

The DEBUG(FRMSG(MINIMAL | SUMMARIZED | DETAILED)) keyword overrides the DEBUG=FRMSG parameter specified in the JCL EXEC statement.

FRMSG(MINIMAL)

Specifies that DFSMSdss issue an informational message with a minimal level of information that explains why a fast replication method could not be used. See the following examples for messages that are issued when you use this keyword:

Example 1:

ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR
VOLUME SRCV01, RETURN CODE F

Return code F indicates that the volume does not support data set fast replication.

Example 2:

ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR
VOLUME SRCV01, RETURN CODE 3

Return code 3 indicates that the source device is not eligible for fast replication at this time.

FRMSG(SUMMARIZED)

Specifies that DFSMSdss issue an informational message with summary information that explains why a fast replication method could not be used. When applicable, summary information regarding ineligible volumes is provided in the message text. See the following examples for messages that are issued when you use this keyword:

Example 1:

ADR918I (ttt)-mmmmm(yy), FAST REPLICATION COULD NOT BE USED FOR
VOLUME SRCV01, RETURN CODE F

Return code F indicates that the volume does not support data set fast replication.

DEFRAG Command

In this example, the DEBUG(FRMSG(MINIMAL)) keyword provides the same level of informational message as when you specify the DEBUG(FRMSG(SUMMARIZED)) or DEBUG(FRMSG(DETAILED)) keyword.

Example 2:

```
ADR918I (ttt)-mmmm(yy), FAST REPLICATION COULD NOT BE USED FOR  
VOLUME SRCV01, RETURN CODE 3  
VOLUME SRCV01 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE 7 - VERSION 1 FC  
RELATION EXISTS
```

In this example, the DEBUG(FRMSG(SUMMARIZED)) keyword provides the same level of informational message as when you specify the DEBUG(FRMSG(DETAILED)) keyword.

Guideline: When the FASTREPLICATION(REQUIRED) keyword is specified and the DEBUG(FRMSG(MIN | SUM | DTL)) keyword is not specified, DFSMSdss still issues an informational message when a data set fast replication method cannot be used in DEFrag operation. It is as though the DEBUG(FRMSG(SUMMARIZED)) keyword had been specified.

FRMSG(DETAILED)

Specifies that DFSMSdss issue an informational message with detailed information that explains why a fast replication method could not be used. When applicable, detailed information regarding ineligible volumes is provided in the message text. See the following examples for messages that are issued when you use this keyword:

Example 1:

```
ADR918I (ttt)-mmmm(yy), FAST REPLICATION COULD NOT BE USED FOR  
VOLUME SRCV01, RETURN CODE 3  
VOLUME SRCV01 WAS REJECTED FOR QFRVOLS VOLUME REASON CODE 7 - VERSION 1 FC  
RELATION EXISTS
```

In this example, the DEBUG(FRMSG(DETAILED)) keyword provides the same level of informational message as when you specify the DEBUG(FRMSG(SUMMARIZED)) keyword.

DYNALLOC



DYNALLOC specifies dynamic allocation, instead of enqueue, to serialize the use of data sets. The data sets whose extents are relocated are serialized throughout the DEFrag operation. This allows cross-system serialization with the following considerations:

- The serialization is of value only when the dynamic allocation/JES3 interface is not disabled.
- Run time increases when you use the DYNALLOC keyword to serialize data sets (as opposed to enqueue) because overhead is involved in dynamic allocation and serialization across multiple processors.

- If you are running on a system using JES3 with MDS enabled and are not using multisystem GRS (or an equivalent function), you can use the DEFFRAG command DYNALLOC keyword to provide serialization for data sets on shared DASD. However, not all data sets allocated within a JES3 environment are known to the global. The following are two cases where the use of the DYNALLOC keyword does not provide cross-system serialization for these data sets:
 - Allocation of existing (old) data sets whose names appear in the RESDSN and DYNALDSN lists are *not* protected by the DYNALLOC serialization mechanism of DFSMSdss. You can prevent DEFFRAG processing for these data sets by placing their names (or filters for the names) in the EXCLUDE list for the DEFFRAG command.
 - New data sets created with nonspecific allocation (no volume serial supplied) are *not* protected by the DYNALLOC serialization mechanism of DFSMSdss. However, you can use BY filtering with the DEFFRAG command to specifically include or exclude data sets from processing.

DYNAM



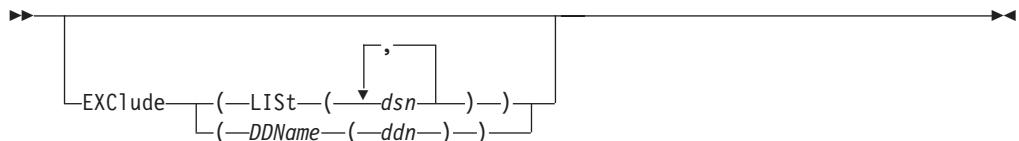
DYNAM specifies that the volume to be processed is to be dynamically allocated. The volume must be mounted and online. You cannot specify a nonspecific volume serial number using an asterisk (*).

Using the DYNAM keyword instead of DD statements to allocate DASD volumes does not appreciably increase run time and permits easier coding of JCL and command input.

volser Specifies the volume serial number of a DASD volume to be processed.

unit Specifies the device type of a DASD volume to be processed. This parameter is optional.

EXCLUDE



LIST(*dsn*)

Specifies a fully or partially qualified name of a data set to be excluded from the DEFFRAG operation. You can specify either a cluster or component name for VSAM data sets.

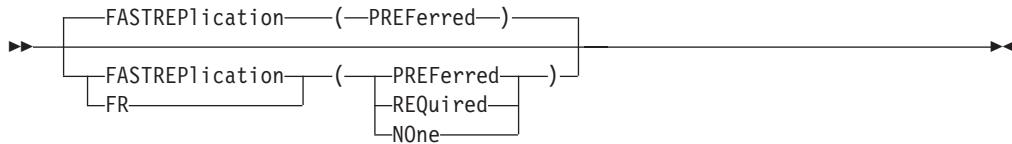
DDNAME(*ddn*)

Specifies the name of the DD statement that identifies a sequential data set or member of a partitioned data set, which contains the list of data sets to be excluded.

DEFFRAG Command

Related reading: For additional information about specific types of data sets that must be placed in the EXCLUDE list if they are present on the volume being defragmented, see the *z/OS DFSMSdss Storage Administration Guide*.

FASTREPLICATION



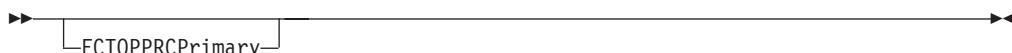
FASTREPLICATION specifies whether the use of fast replication is required, preferred, or not desired. This keyword applies to fast replication methods such as FlashCopy and SnapShot.

PREFERRED specifies that you want to use a fast replication method, if possible. If fast replication cannot be used, DFSMSdss completes the operation using traditional data movement methods. PREFERRED is the default (unless changed to none by the installation).

REQUIRED specifies that fast replication must be used. If fast replication cannot be used, DFSMSdss fails the operation. DFSMSdss issues an informational message regarding why a fast replication method cannot be used even if the DEBUG(FRMSG(MIN | SUM | DTL)) keyword is not specified.

NONE specifies that DFSMSdss not use fast replication methods for the operation. Instead, DFSMSdss uses traditional data movement methods to complete the operation.

FCTOPPRCPrimary



FCTOPPRCPrimary specifies that if FlashCopy is used to perform the defrag operation, a Peer-toPeer Remote Copy (PPRC) primary volume is allowed to become a FlashCopy target volume. To specify FCTOPPRCPrimary, RACF authorization may be required.

When FlashCopy is not used to perform the defrag operation, the FCTOPPRCPrimary keyword is ignored.

When FCTOPPRCPrimary is not specified or if the capability is not supported by the ESS, a PPRC primary volume is not eligible to become a FlashCopy target volume.

Attention: When FCTOPPRCPrimary is specified the FlashCopy operation will cause a PPRC primary volume to become a FlashCopy target volume. The PPRC-SYNC volume pair currently in full duplex state will go into a duplex pending state when the FlashCopy relationship is established. When PPRC completes the operation, the PPRC_SYNC volume pair will go to full duplex state. For more information on PPRC options and volume states see: *z/OS DFSMS Advanced Copy Services*.

Notes:

1. Do not specify the FCTOPPRCPrimary keyword with the FASTREPLICATION(NONE) keyword.

Related reading:

- For additional information about RACF authorization see: *z/OS DFSMSdss Storage Administration Guide*.
- For additional information about RACF FACILITY class profiles see: *z/OS Security Server RACF Security Administrator's Guide*.
- For additional information about PPRC (PPRC-SYNC), PPRC-XD, and PPRC V2 see: *z/OS DFSMS Advanced Copy Services* and the IBM TotalStorage Enterprise Storage Server Implementing ESS Copy Services redbook.

FORCECP

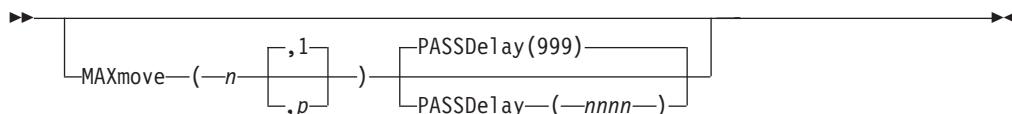
FORCECP specifies that extents of checkpointed data sets on the SMS volume or volumes can be moved. The checkpoint indication is left in place for IMS™ generalized sequential access method (GSAM) data sets, as the data set is still usable for a restart. The checkpoint indication is removed from all volumes for MVS checkpointed data sets, as the data sets are no longer usable for a restart.

days Specifies a one-to-three digit number in the range of zero to 255. It also specifies the number of days that must have elapsed since the last referenced date before the data set can be defragmented.

FRAGMENTATIONINDEX

FRAGMENTATIONINDEX specifies that the DEFrag operation is to end if the fragmentation index is less than *n*, where *n* is a 1- to 3-digit number. DFSMSdss prefixes your number, *n*, with a decimal point. For example, 1 becomes .1, 999 becomes .999, 001 becomes .001, and so forth.

Related reading: For additional information about the fragmentation index of a volume, see the *z/OS DFSMSdss Storage Administration Guide*.

MAXMOVE

n Specifies a 1- to- 6 digit number that specifies that DFSMSdss is to attempt to assemble up to *n* free tracks in a contiguous area. If *n* is greater than the total number of free tracks on the volume, for example MAXMOVE(999999), a message is issued and the DEFrag operation adjusts *n* to the number of free tracks.

DEFRAG Command

<i>p</i>	Specifies a 1- to- 2 digit number that specifies that DFSMSdss is to make up to <i>p</i> passes in attempting to assemble the tracks. If <i>p</i> is not specified, for example MAXMOVE(99), only one pass is made, that is, MAXMOVE(99,1).
PASSDelay	specifies the time delay between the passes (<i>p</i>) specified in MAXMOVE (<i>n,p</i>). PASSDELAY is only meaningful if (<i>p</i>) is greater than one.
<i>nnnn</i>	Specifies a 1- to- 4 digit number from 0 to 9999 that specifies the time delay in milliseconds (1/1000 of a second). When MAXMOVE (<i>n,p</i>) is specified and PASSDELAY is not specified, a default PASSDELAY value of 999 (almost a second) is used to allow access to the volume between MAXMOVE passes.

DEFRAG processing attempts to move a fraction of the total free tracks during each pass. DEFrag processing calculates an integer limit that is the larger of *n* divided by *p* or the value of 15. The limit is compared to the cumulative number of tracks moved after each extent is moved. If the limit is reached or exceeded, the current pass ends, and a new pass starts. Between each pass, the DEFrag function releases and re-obtains volume serialization. This action reduces the overall time that a DEFrag operation makes a volume unavailable to other applications.

The DEFrag operation may end before completing *p* passes. If the DEFrag operation cannot assemble *n* contiguous free tracks without moving more than *n* tracks, DFSMSdss issues a message and ends the DEFrag operation. If *n* contiguous free tracks exist, the DEFrag operation still attempts to reduce the fragmentation of the volume if it can do so without relocating more than *n* tracks.

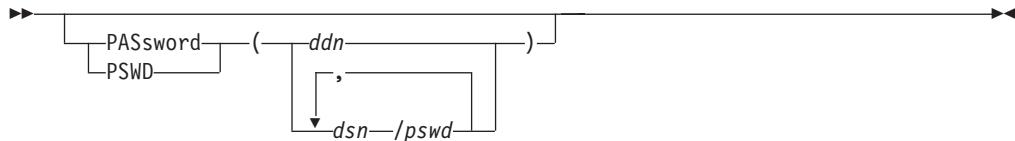
The operation or the current pass also ends when DEFrag processing criteria (for example, FRAGI) are satisfied. If you specified more than one pass, only the current pass ends. DFSMSdss issues message ADR233W for the current pass when the FRAGI criteria has been met. The DEFrag function continues to run and attempts to complete the specified number of passes. DFSMSdss issues message ADR233W for each pass that meets the FRAGI criteria. The function continues because activity on the volume between the DEFrag passes may change the fragmentation index and the subsequent DEFrag passes may further reduce the fragmentation of the volume.

In order to re-obtain volume serialization between DEFrag passes, DFSMSdss uses a default value of WAIT(3,30). This means that DFSMSdss is to retry the volume serialization 30 times at 3 second intervals for a total wait time of 90 seconds. The DEFrag operation issues a message and ends if DFSMSdss cannot obtain volume serialization; any remaining passes do not run. "The WAIT Option" on page 289 explains how to change the wait/retry values for the volume serialization if the default is not satisfactory.

If MAXMOVE is not specified, DFSMSdss tries to assemble a contiguous free area of a size equal to the total number of free tracks on the volume in a single pass. The DEFrag function uses two methods to assemble free tracks on the volume. The first method attempts to assemble the largest contiguous amount of free space with a minimum amount of data movement. The second method attempts to assemble multiple groups of large areas of contiguous free space and generally moves more data around than the first method. When MAXMOVE is specified, only the first method is used during each pass.

PASSDELAY

See "MAXMOVE" on page 115.

PASSWORD

PASSWORD specifies the passwords DFSMSdss is to use for password-protected data sets. (Password checking is bypassed for RACF-protected data sets.) This keyword must be specified only if:

- You do not have the required RACF DASDVOL or RACF DATASET access.
- The installation authorization exit does not bypass the checks.
- You do not want to be prompted for the password for VSAM data sets.

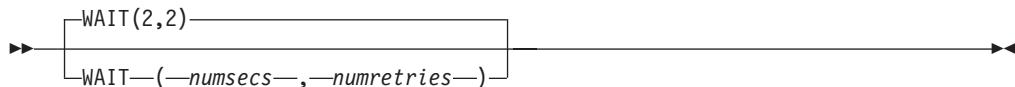
Note: You should specify the passwords for all data sets that do not have RACF protection but do have password protection. During processing, a utility invoked by DFSMSdss may have to prompt the operator for a password. You can control authorization checking by using the installation authorization exit.

For VSAM data sets, password checking is done only at the cluster level.

ddn Specifies the name of the DD statement that identifies the sequential data set, or member of a partitioned data set that contains data set names and their passwords. This data set must contain card-image records in DFSMSdss command syntax format.

dsn/pswd *dsn* is a fully qualified data set name. *pswd* is its password. If no password follows the slash (/), *dsn* is treated as though it were *ddn*.

Printing of actual data set passwords specified in the input command stream is suppressed in the SYSPRINT output.

WAIT

WAIT specifies to DFSMSdss the length of a wait in seconds and the number of retries to obtain control of a data set.

numsecs Specifies a decimal number from 1 to 255 that specifies the interval, in seconds, between retries.

numretries Specifies a decimal number (0–99) that specifies the number of times an attempt to gain control of a data set is to be retried.

The default for *numsecs,numretries* is (2,2), which specifies two retries at 2-second intervals. If you do not want to wait for a data set, specify 0 for either *numsecs* or *numretries*.

DEFFRAG Command

Note: The WAIT keyword does not control wait/retry attempts for system resources (such as the VTOC and the VVDS). For system resources, the default wait time is 3 seconds and the default retry count is 30. This results in a total wait time of 90 seconds.

Related reading: For additional information about controlling the wait/retry attempts for system resources, see *z/OS DFSMSdss Storage Administration Guide*.

WRITECHECK



WRITECHECK specifies that the data moved by the DEFFRAG operation is to be verified for successful completion. This keyword increases the overall elapsed time.

Examples of DEFFRAG Operations

Example 1: A DEFFRAG Operation with Excluded Data Sets

```
//JOB1    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC  PGM=ADRDSU
//SYSPRINT DD   SYSOUT=A
//DASD     DD   UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//A1       DD   DSN=USER2.EXCLUDE,DISP=SHR
//SYSIN   DD   *
      command input (see examples 1A and 1B below)
/*
```

Example 1A: With the Names of Excluded Data Sets in the Input Stream

```
DEFFRAG  DDNAME(DASD) -
          EXCLUDE(LIST(USER2.**.LIST,*.LOAD))
```

Example 1B: With the Names of Excluded Data Sets in a Data Set

```
DEFFRAG  DDNAME(DASD) -
          EXCLUDE(DDNAME(A1))
```

In examples 1A and 1B, DASD volume 111111 is defragmented. All data sets whose first and last qualifiers are USER2 and LIST, respectively, are to be excluded from this operation, as are data sets with two qualifiers whose second qualifier is LOAD. In example 1B, cataloged data set USER2.EXCLUDE contains a single card-image record with the following in columns 2 through 72:

```
USER2.**.LIST,*.LOAD
```

Example 2: A DEFrag Operation Using a BY Criterion

```
//JOB2      JOB  accounting information,REGION=nnnnK
//STEP1    EXEC PGM=ADRDSU
//SYSPRINT DD   SYSOUT=A
//DASD      DD   UNIT=3380,VOL=(PRIVATE,SER=11111),DISP=OLD
//SYSIN    DD   *
      DEFrag DDNAME(DASD) /* VOLUME TO BE PROCESSED */
                           BY(LIST(REFDT LT *, -1)) /* DATE LAST REF LT RUN DATE -1 */
/*
/*
```

In Example 2, only data sets that were last referenced more than one day before the run date are included in the DEFrag operation. That is, those that were last referenced one day before or on the run date are excluded.

Results of a Successful DEFrag Operation

Figure 1 on page 120 is the printout from a DEFrag run for a DASD volume. It gives an indication of the free space fragmentation before and after a DEFrag operation, as well as the distribution of data set extents by size.

The following JCL was used for this job:

```
//STEPT010 EXEC PGM=ADRDSU,PARM='RACFLOG=YES,TRACE=YES'
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
      DEFrag DYNAM(D9S060)
/*
/*
```

DEFrag Command

```
PAGE 0001      5695-DF175  DFSMSDSS V2R10.0 DATA SET SERVICES   1999.211 14:55
  DEFrag
    DYNAM(D9S060)
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'DEFrag'
ADR109I (R/I)-RI01 (01), 1999.211 14:55:33 INITIAL SCAN OF USER CONTROL
STATEMENTS COMPLETED.
ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR061I (001)-STEND(01), 1999.211 14:55:33 EXECUTION BEGINS
ADR208I (001)-EANAL(01), 1999.211 14:55:34 BEGINNING STATISTICS ON D9S060:
          FREE CYLINDERS          000058
          FREE TRACKS            000003
          FREE EXTENTS           000002
          LARGEST FREE EXTENT (CYL,TRK) 000053,0003
          FRAGMENTATION INDEX     0.050
          PERCENT FREE SPACE      97
ADR220I (001)-EANAL(01), INTERVAL BEGINS AT CC:HH 00001:0000 AND ENDS AT
CC:HH 00006:000C
ADR209I (001)-EFRAG(01), 1999.211 14:55:34 MOVED EXTENT 001 FROM
00006:0000-00006:0004 TO 00001:0000-00001:0004
FOR PUBSEXMP.ESDS.S01.DATA
ADR209I (001)-EFRAG(01), 1999.211 14:55:34 MOVED EXTENT 001 FROM
00006:0005-00006:0006 TO 00001:0005-00001:0006
FOR PUBSEXMP.KSDS.S01.DATA
ADR209I (001)-EFRAG(01), 1999.211 14:55:35 MOVED EXTENT 001 FROM
00006:0007-00006:0008 TO 00001:0007-00001:0008
FOR PUBSEXMP.SAM.S01
ADR209I (001)-EFRAG(01), 1999.211 14:55:35 MOVED EXTENT 001 FROM
00006:0009-00006:000A TO 00001:0009-00001:000A
FOR PUBSEXMP.PDS.S01
ADR209I (001)-EFRAG(01), 1999.211 14:55:35 MOVED EXTENT 002 FROM
00006:000B-00006:000B TO 00001:000B-00001:000B
FOR PUBSEXMP.PDS.S01
ADR213I (001)-EANAL(01), 1999.211 14:55:35 ENDING STATISTICS ON D9S060:
          DATA SET EXTENTS RELOCATED      000005
          TRACKS RELOCATED                000012
          FREE CYLINDERS                 000058
          FREE TRACKS                   000003
          FREE EXTENTS                  000001
          LARGEST FREE EXTENT (CYL,TRK) 000058,0003
          FRAGMENTATION INDEX          0.000
PAGE 0002      5695-DF175  DFSMSDSS V2R10.0 DATA SET SERVICES   1999.211 14:55
ADR212I (001)-EANAL(01), EXTENT DISTRIBUTION MAP FOR D9S060:
          EXTENT *FREE SPACE BEFORE* *FREE SPACE AFTER* *ALLOCATED*
          SIZE
          IN      NO.    CUM.      NO.    CUM.      NO.    CUM.
          TRACKS  EXTS  PCT/100  EXTS  PCT/100  EXTS  PCT/100
            1          4    0.148
            2          4    0.444
            5          1    0.629
            10         1    1.000
            75         1    0.085
            >499        1    1.000      1    1.000
ADR061I (001)-STEND(02), 1999.211 14:55:35 EXECUTION ENDS
ADR013I (001)-CLTSK(01), 1999.211 14:55:35 TASK COMPLETED WITH RETURN CODE 0000
ADR012I (SCH)-DSSU (01), 1999.211 14:55:35 DFSMSDSS PROCESSING COMPLETE. HIGHEST
RETURN CODE IS 0000
```

Figure 1. Printed Output Resulting from a Successful DEFrag Run

The value in the first column of message ADR212I is the size of the extent in tracks (free space or data set) and is printed only if an extent of that size occurs. The second and third columns show the number of free space extents existing *before* processing that were of the size shown in the first column, along with their cumulative percentage divided by 100. The fourth and fifth columns give the same information for free space, but *after* processing. The sixth and seventh columns give the distribution of allocated data set extents, which do not change during a run.

DUMP Command

With the DUMP command, you can dump DASD data to a sequential data set. The storage medium for the sequential data set can be a tape or DASD. You can dump data sets, an entire volume, or ranges of tracks.

Note: The FULL keyword for the DUMP command specifies that an entire DASD volume is to be dumped. The TRACKS keyword with the DUMP command specifies ranges of tracks to be dumped.

DFSMSdss offers two ways to process DUMP commands:

- *Logical processing* is data set-oriented, which means it operates against data sets independently of physical device format.
- *Physical processing* can operate against data sets, volumes, and tracks, but is oriented toward moving data at the track-image level.

The processing method is determined by the keywords specified on the command.

DFSMSdss logical dump processing cannot be used to process partitioned data sets containing location-dependent information that does not reside in note lists or in the directory. Furthermore, DFSMSdss cannot be used to dump migrated data sets.

Integrated catalog facility catalogs should not have a high-level qualifier of SYSCTLG because this causes DFSMSdss to treat them as CVOLs.

Related reading: For additional information about using the DUMP command, see the *z/OS DFSMSdss Storage Administration Guide*.

Special Considerations for Dump

The following special considerations apply when you are performing a dump operation:

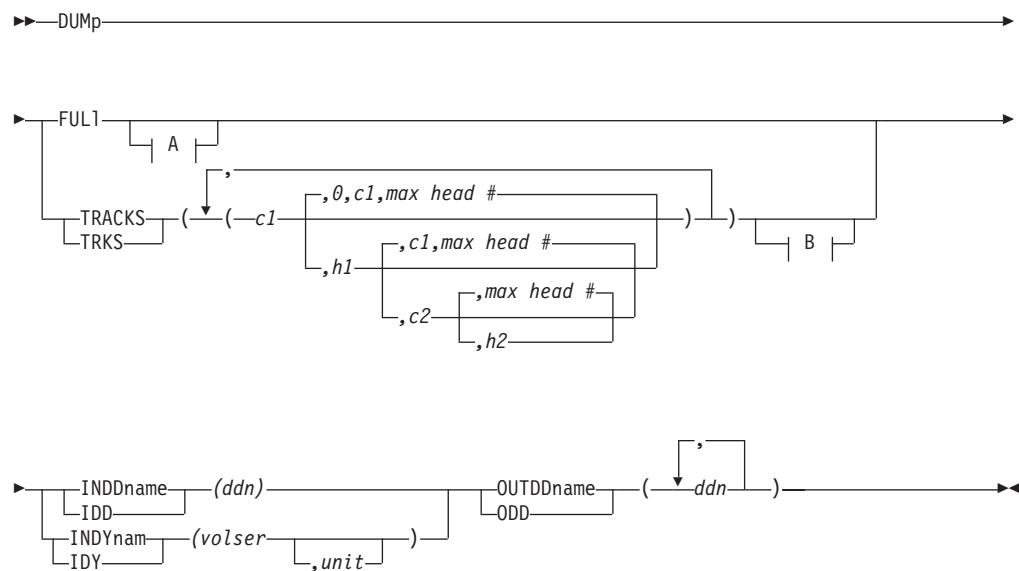
- A logical data set dump cannot be performed on the following data sets:
 - VSAM data sets not cataloged in an integrated catalog facility catalog
 - Page, swap, and SYS1.STGINDEX data sets
 - VSAM Volume Data Sets (VVDS)
 - Partitioned data sets containing location-dependent information that does not reside in note lists or in the directory
- A physical data set dump cannot be performed on the following data sets:
 - KSDSs with key ranges. Use logical processing for this type of data set.
 - VSAM data sets not cataloged in an integrated catalog facility catalog.
 - Page, swap, and SYS1.STGINDEX data sets.

Note: DFSMSdss cannot be used to dump data sets with a volume serial of MIGRAT. The recommended method of dumping migrated data sets is ABARS.

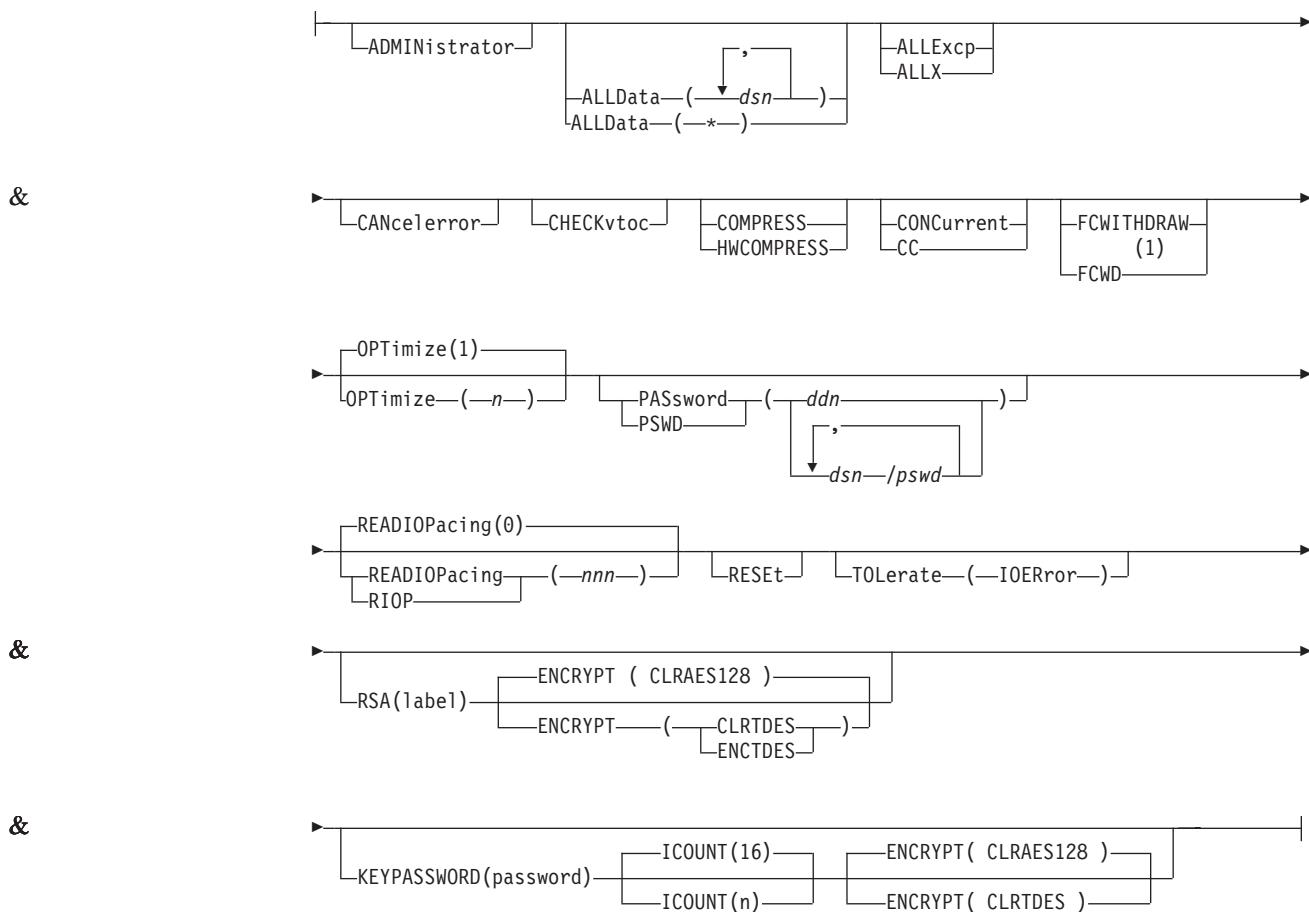
Related reading: For additional information about dumping data sets, see the *z/OS DFSMSdss Storage Administration Guide*.

DUMP Command

DUMP FULL and DUMP TRACKS Syntax

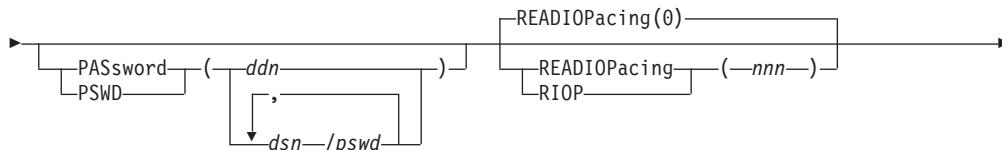
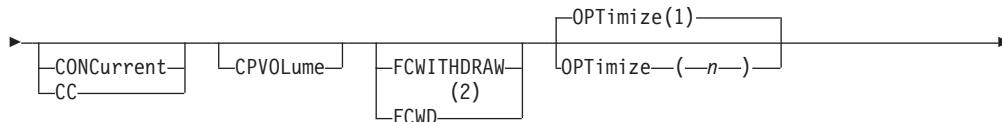


A: Optional Keywords with DUMP FULL:

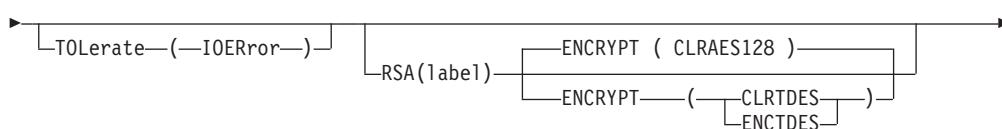


B: Optional Keywords with DUMP TRACKS:

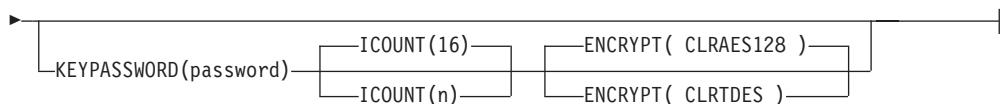
&



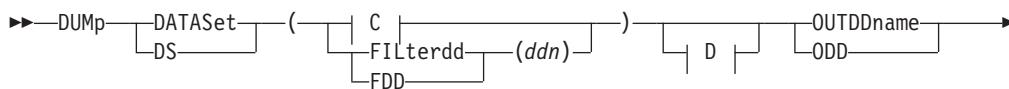
&



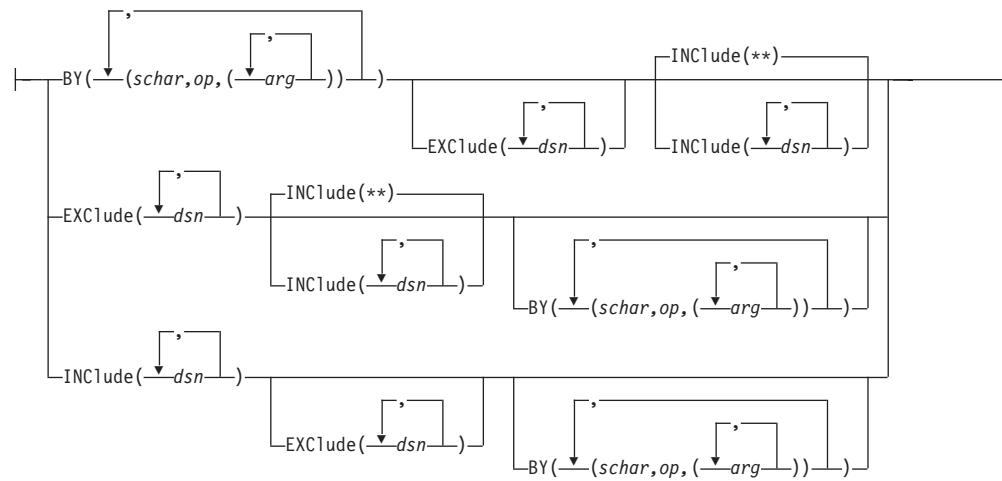
&

**Notes:**

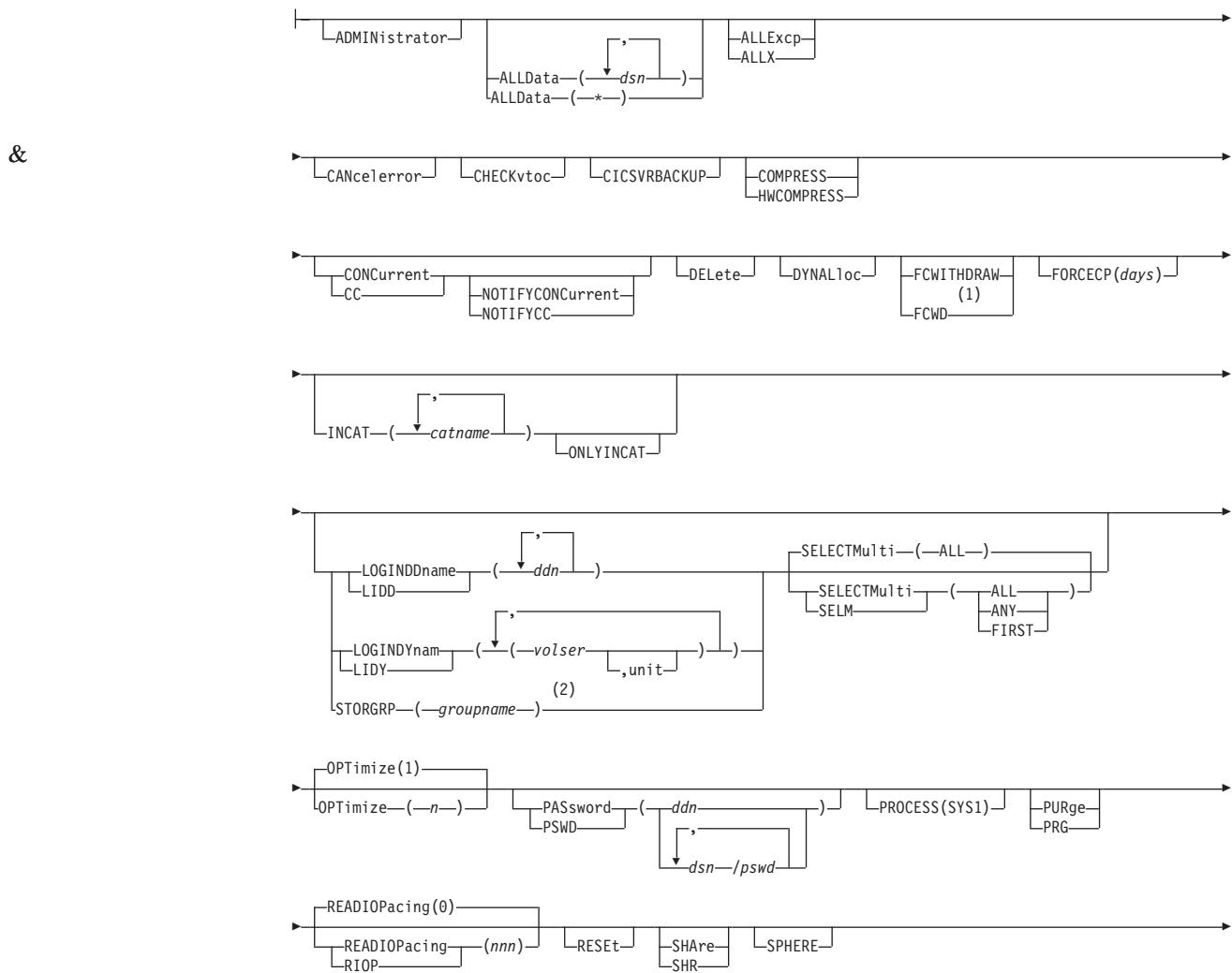
- 1 You cannot specify the FCWITHDRAW keyword with the CONCURRENT or RESET keywords at the same time.
- 2 You cannot specify the FCWITHDRAW keyword with the CONCURRENT or RESET keywords at the same time.

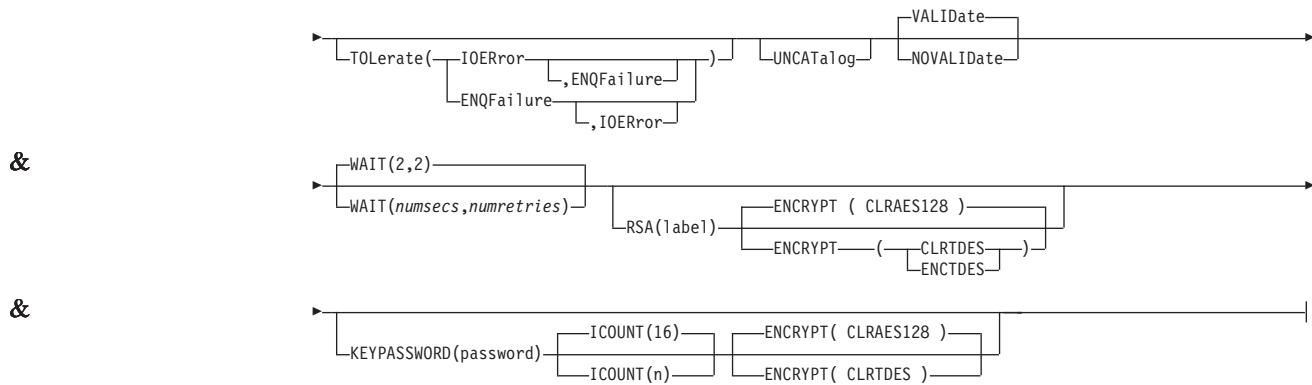
DUMP DATASET Syntax for Logical Data Set**C: Additional Keywords with DUMP DATASET for Logical Data Set:**

DUMP Command



D: Optional Keywords with DUMP DATASET for Logical Data Set:

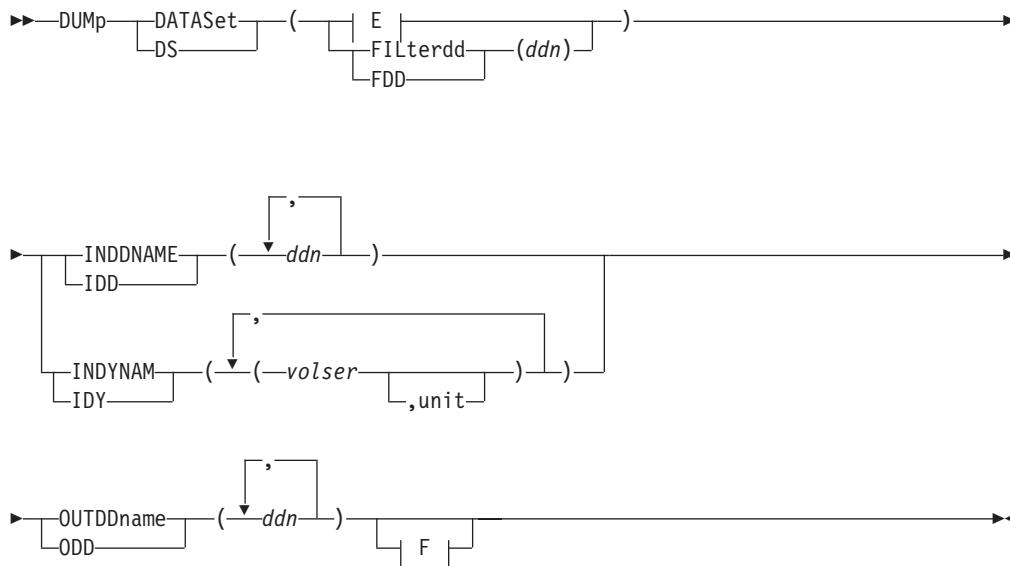




Notes:

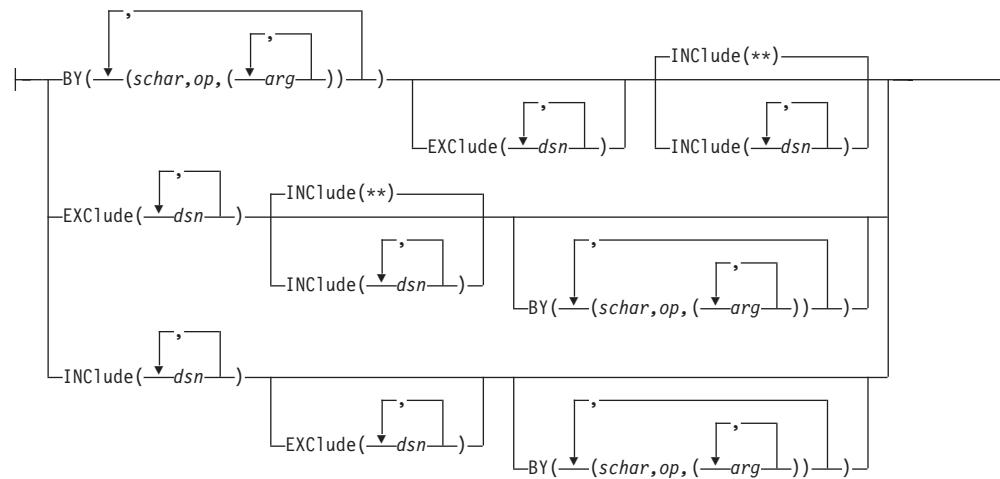
- 1 You cannot specify the FCWITHDRAW keyword with the CONCURRENT or RESET keywords at the same time.
- 2 You cannot specify the STORGRP keyword with the INDDNAME, INDYDYNAM, LOGINDDNAME or LOGINDYDYNAM keywords at the same time.

DUMP DATASET Syntax for Physical Data Set

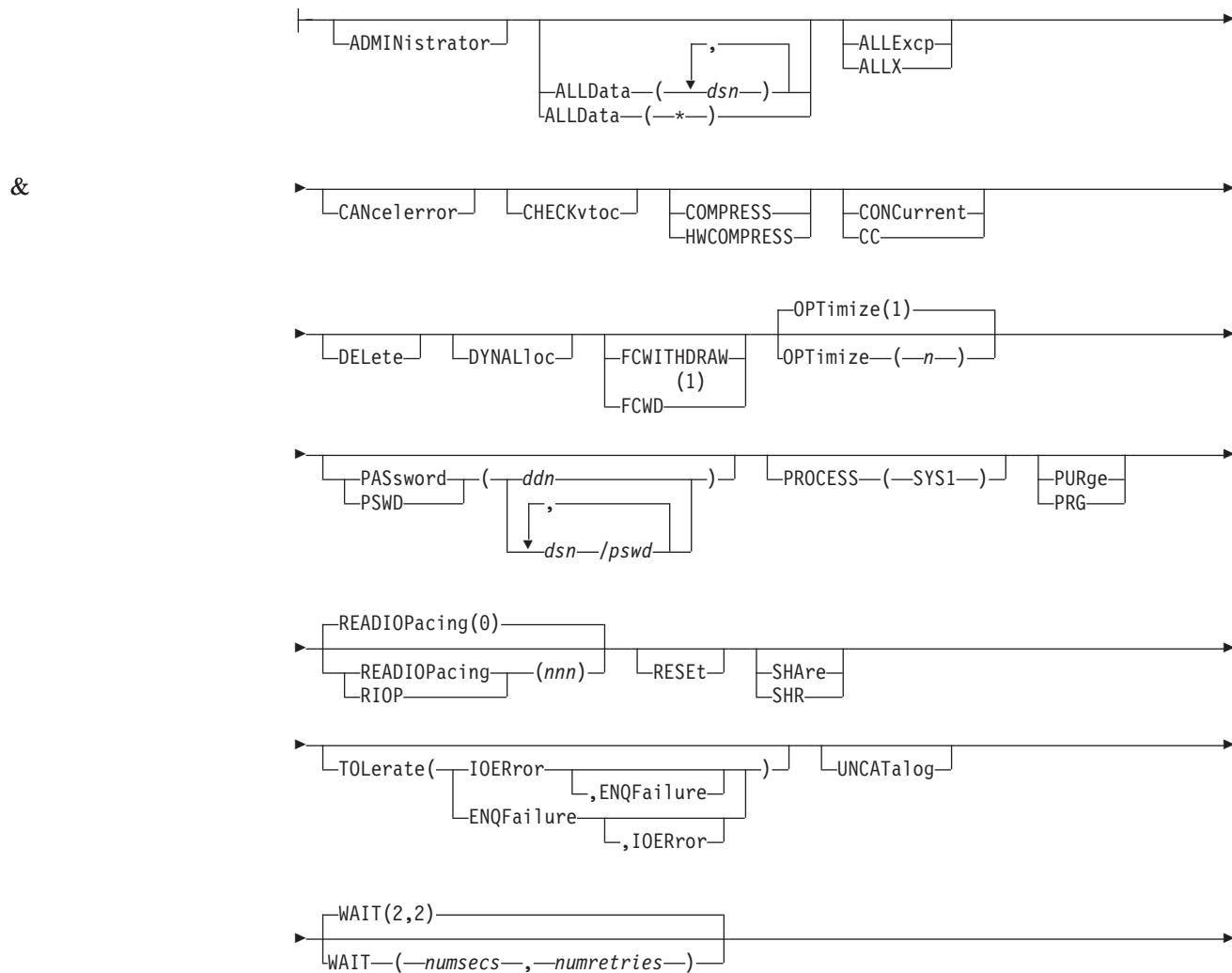


E: Additional Keywords with DUMP DATASET for Physical Data Set:

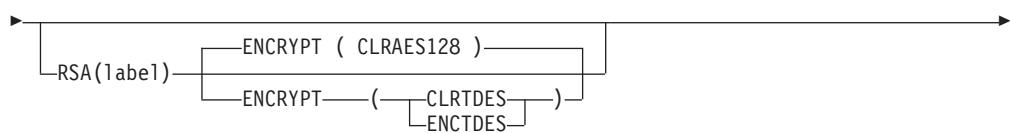
DUMP Command



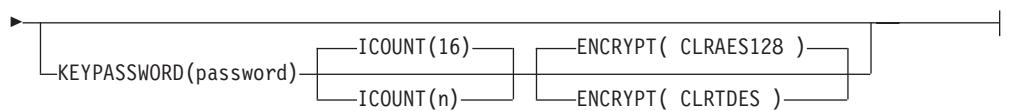
F: Optional Keywords with DUMP DATASET for Physical Data Sets:



&



&

**Notes:**

- 1 You cannot specify the FCWITHDRAW keyword with the CONCURRENT or RESET keywords at the same time.

Explanation of DUMP Command Keywords

This section describes the keywords for the DUMP command.

ADMINISTRATOR



ADMINISTRATOR lets you act as a DFSMSdss-authorized storage administrator for the DUMP command. If you are not authorized to use the ADMINISTRATOR keyword, the command is ended with an error message. Otherwise, access checking to data sets and catalogs that are initiated by DFSMSdss are bypassed.

To use the ADMINISTRATOR keyword all of the following must be true:

- FACILITY class is active.
- Applicable FACILITY-class profile is defined.
- You have READ access to that profile.

Related reading: For additional information about how to use the ADMINISTRATOR keyword, see “ADMINISTRATOR Keyword” on page 265.

ALLDATA



ALLDATA applies to full and data set dump operations.

- dsn** Specifies the fully qualified name of a data set whose data set organization is PS, PSU, PO, POU, or null, for which all allocated space is to be dumped.
- * (asterisk)** Specifies that all allocated space is to be dumped for the following data sets:
 - All sequential and partitioned data sets that are not empty, and
 - Data sets whose data set organization is null and are not empty.
(The last used block pointer in the data set's volume table of contents (VTOC) entry is not zero.)

Notes:

1. Data beyond the last used block pointer is not retained when ALLDATA or ALLEXCP is specified for an extended-format sequential data set during a logical dump operation. During a subsequent restore operation, the target data set is allocated with the same amount of space as the source data set.
2. All of the allocated space is always dumped for a physical dump of PDSE and HFS data sets.

3. For a logical dump of PDSE or HFS data sets, the following conditions are true:
 - If DFSMSdss can determine the amount of used space, DFSMSdss dumps only the used space, regardless of the ALLDATA keyword.
 - If you specify ALLDATA, DFSMSdss remembers the amount of allocated space. During a subsequent restore operation, DFSMSdss will allocate the target data set with the same amount of space as the source data set.
 - If DFSMSdss cannot determine the amount of used space, it assumes that the used space is equal to the allocated space. DFSMSdss then dumps all of the allocated space.

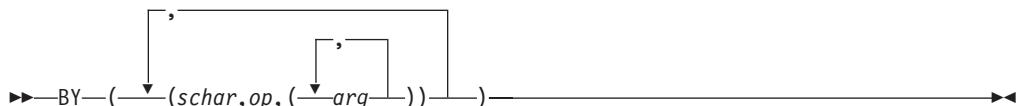
ALLEXCP



ALLEXCP is an option for full and data set dump operations. It instructs DFSMSdss to dump all allocated space for data sets whose data set organization is PS, PSU, PO, POU, or null and are empty (the last used block pointer in the data set's VTOC entry is zero).

Note: Using the ALLEXCP keyword will result in all the allocated space being dumped for applicable data sets. However, whether or not all the allocated space is restored depends on data set characteristics and device characteristics during the restore. If you require that all of the unused space is restored, then you should be sure that the data set is restored to a like device type and not reblocked or compressed. Compress is the default for PDS data sets on restore unless you use the NOPACKING keyword.

BY



BY specifies that the data sets selected up to this point by the processing of the INCLUDE and EXCLUDE keywords are to be further filtered. To select the data set, *all* BY criteria must be met. See "Filtering by Data Set Characteristics" on page 14 for a full discussion of *schar*, *op*, and *arg*. See the separate discussions of INCLUDE and EXCLUDE for information on how these keywords are specified.

Note: You must use FILTERDD when you have more than 255 entries in INCLUDE, EXCLUDE, or BY list keywords.

DUMP Command

CANCELERROR

```
>>> [ CANCELERROR ] <<<
```

CANCELERROR specifies that the dump operation is to be ended if a track-related permanent read error occurs. If this keyword is *not* specified and a permanent read error occurs, the track image record is flagged on the output volume as having an I/O error, and the dump operation continues. This track is not restored in a restore operation. When CANCELERROR is specified, the TOLERATE(IOERROR) keyword is ignored.

This keyword may be used in conjunction with the CHECKVTOC keyword to specify whether or not an operation is to continue in the event of terminating VTOC errors discovered during VTOC checking. Refer to the CHECKVTOC keyword.

Note: CANCELERROR has no effect on the following types of errors on a DASD volume:

- Equipment check
- Command reject
- Intervention required
- Busout parity

CHECKVTOC

```
>>> [ CHECKVTOC ] <<<
```

CHECKVTOC specifies that a VTOC analysis of the source volume be performed during dump processing. In the event of terminating VTOC errors found during analysis, operation continues unless the CANcelerror keyword is specified. CHECKVTOC is ignored if CPVOLUME is also specified.

CICSVRBACKUP

```
>>> [ CICSVRBACKUP ] <<<
```

CICSVRBACKUP specifies that for a logical data set dump operation, DFSMSdss creates backups for use by CICSVR. DFSMSdss notifies the CICSVR server address space when a CICSVR backup is made for a VSAM base cluster. This enables CICSVR to manage backups made by DFSMSdss.

Notes:

1. CICSVRBACKUP is intended to be used in conjunction with CICSVR. The minimum required CICSVR release is Version 3 Release 1. To use CICSVRBACKUP, the CICSVR server address space must be active.
2. CICSVRBACKUP applies to a logical data set dump only.
3. CICSVR manages VSAM base clusters backed up using the DFSMSdss DUMP command. The CICSVRBACKUP keyword is ignored for non-VSAM data sets

and VSAM alternate indexes. When you specify CICSVRBACKUP, DFSMSdss DUMP processes AIXs as usual but does not notify CICSVR of backups that are made for AIXs.

Related reading: For additional information about using DFSMSdss DUMP to create CICSVR backups, see the *CICSVR Implementation Guide*.

&

COMPRESS

&



&

COMPRESS specifies that the dumped data is to be written in compressed form to the output medium. This decreases the space occupied by the dump data at the expense of increased processor and elapsed times.

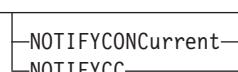
&

Notes:

1. The COMPRESS keyword is ignored if it is specified during a logical data set dump for either compressed-format sequential data sets or compressed-format VSAM data sets.
2. If you have a tape drive with the compaction feature and you want to use hardware data compaction, you do not need to specify the COMPRESS keyword. If software data compression is desired, you do not need to specify DCB=TRTCH=COMP in the JCL. However, you may specify both the COMPRESS keyword and DCB=TRTCH=COMP.
3. To improve performance and save dump space, specify the OPTIMIZE keyword with the COMPRESS keyword.

&

CONCURRENT



CONCURRENT specifies that the data is to be processed with concurrent copy if it is possible, otherwise, the data will be processed as if CONCURRENT was not specified.

If the source volume is a RAMAC Virtual Array and CONCURRENT is specified, DFSMSdss uses the SnapShot capability of the RVA to provide a function equivalent to concurrent copy. This function is called virtual concurrent copy and is transparent to the user.

If a concurrent copy operation fails after signaling that the concurrent copy initialization was complete (and update activity on the data has resumed), it is not possible to recover the data at the point-in-time at which the concurrent copy operation was started. This is because the data may have been updated while the dump operation was progressing.

DUMP Command

Attention: Performing concurrent copy operations against many large data sets when there is also heavy update activity (such as reorganizing data sets or initializing the volume the data sets reside on) can result in a shortage of storage. The shortage occurs because data is transferred to data-space storage faster than DFSMSdss can process it.

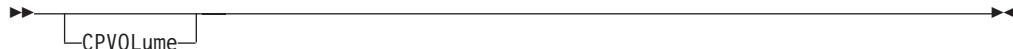
Restriction: The CONCURRENT keyword cannot be specified with the DELETE, UNCATALOG, or FCWITHDRAW keywords.

NOTIFYCONCURRENT

Specifies (logical data set dump operation only) that DFSMSdss issue message ADR767I for every data set that is successfully included in the concurrent copy operation. If NOTIFYCONCURRENT is not specified, messages are issued only for data sets that are not successfully included in the concurrent copy operation.

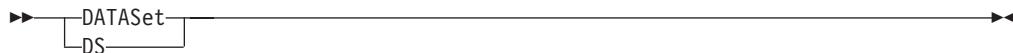
Related reading: For additional information about determining concurrent copy storage requirements, see the *z/OS DFSMSdss Storage Administration Guide*.

CPVOLUME



CPVOLUME specifies that the input volume is a VM-format volume and that the OS-compatible VTOC must begin on track zero, record five. You must specify the track range to be copied with the TRACKS keyword, as the OS-compatible VTOCs do not describe the extents of any data on the volume. You must also specify the ADMINISTRATOR keyword with CPVOLUME because DFSMSdss cannot check access authorization for VM data.

DATASET



DATASET specifies a data set dump operation, using filtering. See Chapter 2, “Filtering—Choosing the Data Sets You Want Processed,” on page 11 for an explanation of the filtering process used. Unless the ALLDATA or ALLEXCP keyword is specified, only *used tracks* are dumped for sequential and partitioned data sets and for data sets with no defined data set organization (for example, JES2/JES3 data sets). If the free space map in the VTOC is invalid, all tracks are dumped.

Note: Either the FILTERDD, INCLUDE, EXCLUDE, or BY keyword must be specified when DATASET is selected.

DELETE



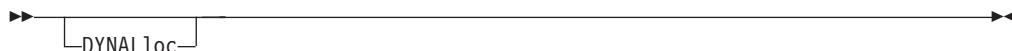
For a *physical data set dump*, DELETE instructs DFSMSdss to delete expired single-volume, non-VSAM data sets that are successfully serialized and dumped. In addition, DFSMSdss is to uncatalog successfully deleted data sets. DELETE is ignored for VSAM data sets.

For a *logical data set dump*, DELETE can be used to delete expired single- and multivolume VSAM and non-VSAM data sets that are successfully serialized and dumped. Unmovable data sets can also be deleted. User catalogs cannot be deleted. *Unexpired* source data sets are deleted only if you also specify PURGE.

Notes:

1. For both a *physical* and a *logical* data set dump operation, you cannot delete data sets with a high-level qualifier of SYS1 unless you specify PROCESS(SYS1). Even if PROCESS(SYS1) is specified, SYS1.VVDS and SYS1.VTOCIX data sets cannot be dumped and deleted. To delete or scratch a password-protected data set, the operator must supply the password for DADSM scratch password checking.
2. Do not specify SHARE if you specify DELETE.
3. Do not specify DELETE with CONCURRENT, because after the concurrent copy operation has begun, the original data can still be updated.
4. Do not specify DELETE for a mounted HFS data set. DFSMSdss can delete an HFS data set only if DFSMSdss can enqueue the data set exclusively. This is only possible if the data set is unmounted.

DYNALLOC



DYNALLOC specifies dynamic allocation, instead of enqueue, to serialize the use of data sets. This allows cross-system serialization in a JES3/MVS environment.

Notes:

1. The serialization is of value only when the dynamic allocation/JES3 interface is not disabled.
2. Run time increases when you use DYNALLOC to serialize data sets (as opposed to enqueue) because overhead is involved in dynamic allocation and serialization across multiple processors.
3. If a data set passes INCLUDE/EXCLUDE filtering and is migrated before BY filtering and DYNALLOC is used, the dynamic allocation causes the data set to be recalled. DFSMSdss waits for the recall processing to complete. If the data set is recalled to a different volume, a message is issued indicating that the VTOC entry was not found.
4. For an HFS data set, DFSMSdss ignores DYNALLOC and attempts to get a SYSZDSN enqueue. If DFSMSdss cannot enqueue the HFS data set, DFSMSdss attempts to quiesce the data set.

&
&
&

ENCRYPT

ENCRYPT is a subparameter of the RSA and KEYPASSWORD keywords. See the "RSA" or "KEYPASSWORD" keywords for further explanation.

EXCLUDE

DUMP Command



dsn Specifies the name of a data set to be excluded from the data sets selected by INCLUDE. Either a fully or a partially qualified data set name can be used. See INCLUDE and BY for information on how these keywords are specified.

Note: You must use FILTERDD when you have more than 255 entries in INCLUDE, EXCLUDE, or BY list keywords.

FCWITHDRAW



Notes:

- 1 You cannot specify the FCWITHDRAW keyword with the CONCURRENT or RESET keywords at the same time.

FCWITHDRAW specifies that if the volume that is dumped is the target volume of a FlashCopy relationship, then the FlashCopy relationship is withdrawn when the dump has successfully completed. Withdrawing the FlashCopy relationship releases the subsystem resources that maintain the FlashCopy relationship.

When the volume that is dumped is not FlashCopy capable, the FCWITHDRAW keyword is ignored.

Notes:

1. Be aware that when the FlashCopy relationship is withdrawn, the data on the volume that was dumped becomes invalid. Therefore, only use FCWITHDRAW if you no longer need the data on the volume that is dumped, after the dump has completed.
2. Be aware that if you use the FCNOCOPY keyword, you must withdraw the FlashCopy relationship when the copy is no longer needed to free up the subsystem resources that maintain the FlashCopy relationship. You can withdraw the FlashCopy relationship by performing one of the following tasks:
 - Initiate a dump of the target of the copy and specify the FCWITHDRAW keyword on the DUMP command.
 - Initiate the TSO FCWITHDR command.

FILTERDD



FILTERDD specifies the name of the DD statement that identifies the sequential data set or member of a partitioned data set that contains the filtering criteria to use. The filtering criteria are in the form of card-image records, in DFSMSdss command syntax, containing the INCLUDE, EXCLUDE, and BY keywords that complete the DUMP command syntax.

Note: You must use FILTERDD when you have more than 255 entries in the INCLUDE, EXCLUDE, or BY list of subkeywords.

FORCECP

```
► [ ] FORCECP(days) ◄
```

FORCECP specifies that checkpoint data sets resident on the SMS volume or volumes can be logically dumped. Checkpoint indications are not removed from the data set during conversion.

days Specifies a one-to-three digit number in the range of zero to 255. It also specifies the number of days that must have elapsed since the last referenced date before the data set can be dumped.

FULL

```
► [ ] FUL1 ◄
```

FULL specifies that an entire DASD volume is to be dumped. This is the default. Unallocated tracks are not dumped. Unless the ALLDATA or ALLEXCP keyword is specified, only *used tracks* are dumped for sequential and partitioned data sets and for data sets with no defined data set organization. (for example, JES2/JES3 data sets). If the free space map in the VTOC is invalid, all tracks are dumped. Used tracks consist of the tracks from the beginning of the data set to the last-used track (as indicated by the last used block pointer in the data set's VTOC entry).

Note: You cannot specify SHARE or TOL(ENQF) for FULL operations.

&

HWCOMPRESS

```
► [ ] COMPRESS  
[ ] HWCOMPRESS ◄
```

&

&

&

&

HWCOMPRESS specifies that DFSMSdss writes the dumped data in compressed form to the output medium. This decreases the space occupied by the dump data. Hardware assisted compression is performed on the dumped data using the zSeries CMPSC instruction.

&
&

Logical Data Set DUMP: Data sets that are less than five tracks in size (used space or allocated space when ALLDATA is specified) are not compressed.

&
&
&
&

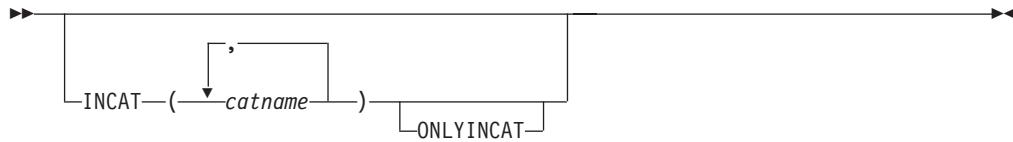
Physical Full Volume, Tracks, and Data Set DUMP: DFSMSdss rebuilds the compression dictionary after 15 consecutive compressions that do not have a new size of at most 94% the original size. The values 15, and 94% may be tuned with patch bytes as described in the DFSMSdss Storage Administration Guide.

&
&

Restriction: The HWCOMPRESS keyword cannot be specified with the COMPRESS keyword.

DUMP Command

INCAT



INCAT(*catname*) specifies that DFSMSdss search the user catalogs specified by the INCAT(*catname*) keyword, then follow the standard search order to locate data sets. INCAT allows you to identify specific source catalogs. To specify INCAT, RACF authorization may be required.

catname Specifies a fully qualified catalog name.

ONLYINCAT Specifies that DFSMSdss only searches catalogs specified in the INCAT catalog name list.

DFSMSdss does not process an SMS-managed data set that is cataloged outside the standard search order. This is the case even if it is cataloged in one of the catalogs that is specified with the INCAT(*catname*) keyword. Ensure that the SMS-managed data sets are cataloged under standard catalog search order.

Related reading: For additional information about RACF authorization, see the *z/OS DFSMSdss Storage Administration Guide*.

INCLUDE



dsn Specifies the name of a data set eligible to be dumped. Either a fully or a partially qualified data set name can be used. See “Filtering by Data Set Names” on page 12. If INCLUDE is omitted (but EXCLUDE or BY is specified) or if INCLUDE(**) is specified, *all* data sets are eligible to be selected for dumping. See the separate discussions of EXCLUDE and BY for information on how these keywords are specified.

Guidelines:

- You must use FILTERDD when you have more than 255 entries in a list that is created using the INCLUDE, EXCLUDE, or BY keywords.
- DFSMSdss does not support INCLUDE filtering of non-VSAM data sets using an alias.

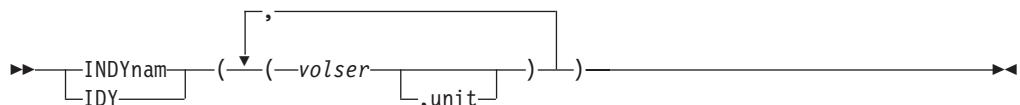
INDDNAME



ddn Specifies the name of the DD statement that identifies the input volume to be dumped. To assure correct processing, each of the DD statements corresponding to a DDNAME (*ddn*) must identify only one volume serial number.

Notes:

1. Only one *ddn* can be specified for INDDname when you use a full or tracks dump operation. One or more are allowed for a physical data set dump operation.
2. Specifying INDDNAME or INDYNAM results in a physical dump. To do a logical data set dump, either do not specify any input volumes with the DATASET keyword or use the LOGINDDNAME or LOGINDYNAM keywords.

INDYNAM

INDYNAM specifies that volumes to be dumped are to be dynamically allocated for a full, tracks, or physical data set dump.

volser Specifies the volume serial number of a DASD volume to be dumped.

unit Specifies the device type of a DASD volume to be dumped. This parameter is optional.

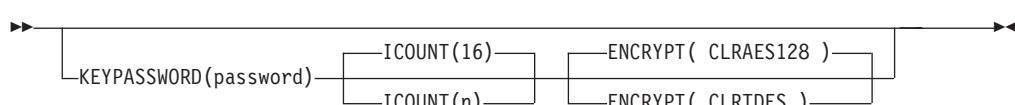
Notes:

1. The volume must be mounted and online. You cannot specify a nonspecific volume serial number using an asterisk (*).
2. Only one volume is allowed for a full or tracks dump operation; one or more volumes are allowed for a physical data set dump operation.
3. Using INDYNAM instead of DD statements to allocate DASD volumes does not appreciably increase run time and permits easier coding of JCL and command input.
4. If either INDDNAME or INDYNAM is specified, *physical* processing is used to perform the dump. If both INDDNAME and INDYNAM are omitted, a *logical* data set dump is performed. A logical data set dump is also performed if you specify the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword.

&

KEYPASSWORD

&



&

KEYPASSWORD

&

&

&

&

The KEYPASSWORD keyword allows you to specify an 8 to 32 EBCDIC character password that is used to generate a clear TDES triple-length key or a clear 128-bit AES key. DFSMSdss removes leading and trailing blanks. Acceptable characters are upper and lower-case letters A through Z, numerals 0-9, and the following characters: !@#\$%&*&_-=:<>?|{}.

&

&

Imbedded spaces, commas (,), forwardslash (/), parentheses (()), and semi-colons are unacceptable characters.

&

&

You can specify this keyword on systems that do not have secure cryptographic hardware (for example, z890 or z990 processors that only

DUMP Command

& use CPACF) or on systems that use the clear key options for
& encryption/decryption (for example, System z9 109 with CPACF).

ICOUNT

& The ICOUNT optional parameter specifies how many times DFSMSdss
& performs the SHA-1 hash algorithm in the generation of the data key and
& initial chaining vector for encryption. n is an integer between 1 and 10000.
& If you do not specify ICOUNT, the default number of iterations is 16.

ENCRYPT

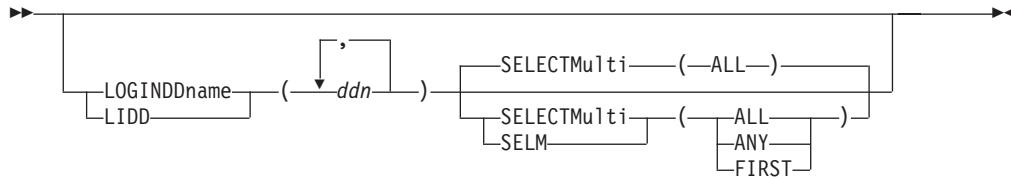
& The ENCRYPT keyword allows you to specify the type of encryption to
& use. The data key used is generated from the password you specified on
& the KEYPASSWORD keyword. If the same password is specified on
& separate DUMP commands, the same data key will be generated for a
& particular encryption type.

& See the "RSA" keyword explanation for more information on the ENCRYPT
& keyword.

Note:

1. When you specify KEYPASSWORD, the only types of encryption that are allowed are CLRTDES and CLRAES128. Secure Triple DES (ENCTDES) is not allowed.
2. When using the KEYPASSWORD keyword, you must take care to ensure that the password is not lost or forgotten. If you lose or forget the password, DFSMSdss will not be able to decrypt the encrypted data on the dump data set. No password recovery mechanism exists. Neither the password or the generated data key is stored on the output medium.
3. Use of the HWCOMPRESS keyword is recommended when using the ENCRYPT keyword.
4. The KEYPASSWORD keyword is mutually exclusive with the RSA keyword.

LOGINDDNAME



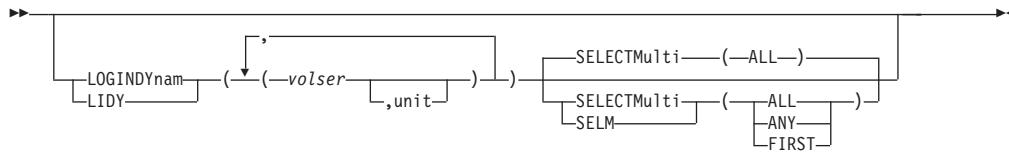
ddn Specifies the name of the DD statement that identifies the input volume that contains the data sets for a logical dump operation. To ensure correct processing, each of the DD statements corresponding to a DDNAME (*ddn*) must identify only one volume serial number.

Notes for LOGINDDNAME, LOGINDYNAM, and STORGRP keywords:

1. When you specify the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword, DFSMSdss uses logical processing to perform the dump operation. Logical processing also occurs when no input volume is specified.
2. A multivolume data set that has extents on volumes that are not specified with the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword will not be dumped unless you specify SELECTMULTI.

See LOGINDYNAM for a description of the SELECTMULTI keyword.

LOGINDYNAM



LOGINDYNAM specifies that volumes that contain the data sets to be dumped using logical processing are to be dynamically allocated.

Note: The volume must be mounted and online. You cannot specify a nonspecific volume serial number using an asterisk (*).

volser Specifies the volume serial number of a DASD volume to be dumped.

unit Specifies the device type of a DASD volume to be dumped. This parameter is optional.

SELECTMULTI

Specifies the method for determining how cataloged multivolume data sets are to be selected during a logical data set dump operation. SELECTMULTI is accepted only when logical volume filtering is specified with one of the following keywords:

- LOGINDDNAME
- LOGINDYNAM
- STORGRP

If logical volume filtering is not used, the specification of SELECTMULTI is not accepted.

ALL Specifies that DFSMSdss *not dump* a multivolume data set unless the following criteria are met:

- The volume list that is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must list all the volumes that contain a part of the data set.
- The volume list that is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must list all the volumes that contain a part of the VSAM cluster.

ALL is the default.

For VSAM data sets, the volume list is affected by the use of the SPHERE keyword as follows:

- Specify SPHERE and you only need to list all parts of the base cluster in the volume list.
- Do not specify SPHERE and you must list all parts of the base cluster and the associated alternate indexes in the volume list.

ANY Specifies that DFSMSdss dump a multivolume data set when the following criteria are met:

- Any volume specified in the volume list that is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must contain a part of the data set.

DUMP Command

- Any volume specified in the volume list that is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must contain a part of the VSAM cluster.

For VSAM data sets, the volume list is affected by the use of the SPHERE keyword as follows:

- Specify SPHERE and you only need to list any part of the base cluster in the volume list.
- Do not specify SPHERE and you must list any part of the base cluster and the associated alternate indexes in the volume list.

FIRST Specifies that DFSMSdss dump a multivolume data set only when the volume list designates the volume that contains the first part of the data set. The volume list is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword.

For VSAM data sets, the volume list is affected by the use of the SPHERE keyword as follows:

- Specify SPHERE and you only need to list the volume that contains the first extent of the data component in the volume list.
- Do not specify SPHERE and you must list the following in the volume list:
 - The volume containing the first extent of the data component for the base cluster
 - The volume containing the first extent of the data component for the associated alternate indexes

Notes for LOGINDDNAME, LOGINDYNAM, and STORGRP keywords:

1. If the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword is specified, DFSMSdss uses logical processing to perform the dump operation. Logical processing is also used if no input volume is specified.
2. A multivolume data set that has extents on volumes not specified with the LOGINDDNAME, LOGINDYNAM or STORGRP keyword will not be dumped unless you specify SELECTMULTI.

NOTIFYCONCURRENT

See "CONCURRENT" on page 54.

NOVALIDATE

See "VALIDATE" on page 149.

ONLYINCAT

See "INCAT" on page 136.

OPTIMIZE



OPTIMIZE specifies the number of tracks to be read at a time, as follows:

- If n is 1, DFSMSdss reads one track at a time.
- If n is 2, DFSMSdss reads two tracks at a time.
- If n is 3, DFSMSdss reads five tracks at a time.
- If n is 4, DFSMSdss reads one cylinder at a time.

If OPTIMIZE is not specified, OPTIMIZE(1) is the default. Specifying OPTIMIZE (2), (3), or (4) reduces the time for a dump. Notice that this keyword uses more real and virtual storage. It also keeps the channel busy for longer blocks of time.

Recommendation: To improve performance and save dump space, specify the OPTIMIZE keyword with the COMPRESS keyword.

OUTDDNAME



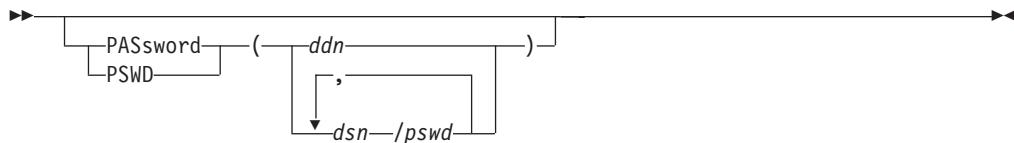
ddn Specifies the name of the DD statement that identifies the (output) dump data set. This data set can be on a tape or a DASD volume. Up to 255 DDNAMEs can be specified; that is, up to 255 dump copies can be made.

Notes:

1. The default block size for output records written to tape is 65 520 bytes. You can change this default to 32 760 bytes by using the installation options exit routine.
2. The default block size for output records written to DASD is the track length for devices whose track length is less than 32KB (KB equals 1 024 bytes), or one-half the track length for devices whose track length is greater than 32KB.
3. If the DCB keyword BLKSIZE is specified on the DD statement for tape or DASD, it must be in the range of 7 892 to 32 760 inclusive.
4. The COPYDUMP command cannot change the block size of the DFSMSdss dump data set. If you intend to copy the dump data set to a DASD device, you must ensure that the block size will be small enough to fit on the target device.
ATTENTION: COPYDUMP is the only supported method for copying DFSMSdss dump data sets. Using a copy produced by any other method or utility as input to a RESTORE operation can produce unpredictable results.
5. If the DCB keyword RECFM is specified on the DD statement, it must have a value of "U".
6. If the DCB keyword LRECL is specified on the DD statement, it must have a value of "0" (zero).
7. The output data set must be a standard format sequential data set and cannot use any extended-format features, such as compression.

Related reading: For additional information about the installation options exit routine, see *z/OS DFSMS Installation Exits*.

PASSWORD



DUMP Command

PASSWORD specifies the passwords that DFSMSdss uses for password-protected data sets for all dump operations. (DFSMSdss bypasses password checking for RACF-protected data sets.) DFSMSdss requires this keyword only when the following apply:

- You do not have the required access for volume-level RACF DASDVOL or RACF DATASET.
- The installation authorization exit does not bypass the checks.
- You do not want a prompt for the VSAM data sets password.

Notes:

1. Specify the passwords for all data sets that do not have RACF protection but do have password protection. A utility invoked by DFSMSdss may prompt the operator for a password during processing. You can control authorization checking by using the installation authorization exit.
2. Actual data set passwords that are specified in your input command stream are not printed in the SYSPRINT output.

The preferred method of protection is catalog protection through an access control facility, such as RACF. Catalog passwords are not supported to facilitate disaster recovery operations, application data transfers, and data set migration.

<i>ddn</i>	Specifies the name of the DD statement that identifies the sequential data set or member of a partitioned data set that contains data set names and their passwords. This data set must contain card-image records in DFSMSdss command syntax format.
<i>dsn/pswd</i>	<i>dsn</i> is a fully qualified data set name. <i>pswd</i> is its password. If no password follows the slash (/), <i>dsn</i> is treated as though it were <i>ddn</i> .

PROCESS



SYS1 specifies that DFSMSdss allows data sets with a high-level qualifier of SYS1 to be dumped and that SYS1 data sets can be deleted and uncataloged. SYS1.VVDS and SYS1.VTOCIX data sets can be physically, but not logically, dumped. SYS1.VVDS and SYS1.VTOCIX data sets cannot be deleted or uncataloged. To specify PROCESS(SYS1), RACF authorization may be required.

Related reading: For additional information about RACF authorization, see the *z/OS DFSMSdss Storage Administration Guide*.

PURGE



PURGE for a data set dump operation, specifies deletion of unexpired data sets that are dumped successfully. This keyword is valid only when DELETE has been specified.

READIOPACING



READIOPACING specifies the pacing (that is, I/O delay) to be used for DFSMSdss DASD read channel programs. You can use this keyword to allow more time for other applications to complete I/O processing. DFSMSdss waits the specified time before issuing each channel program that reads from DASD.

nnn Specifies the amount of time in milliseconds. The maximum delay that can be specified is 999 milliseconds.

Notes:

1. If READIOPACING is not specified, there will be no I/O delay.
2. The additional wait time does not apply to error recovery channel programs.
3. READIOPACING does not apply to concurrent copy I/O.

RESET



RESET specifies that the data set changed flag in the VTOC entry is reset for all data sets that are serialized and dumped successfully. This applies to both a full dump and a data set dump operation.

DUMP Command

Notes:

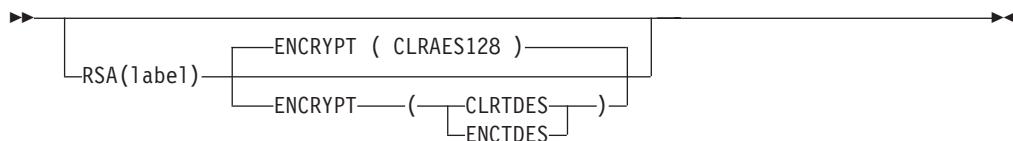
1. Do not specify SHARE or FCWITHDRAW if you specify RESET.
2. You might not want to specify RESET if you use a storage management program, such as DFSMSHsm.
3. DFSMSdss ignores the RESET keyword when a data set is dumped using record-level sharing (RLS) access.
4. Using the RESET keyword for a logical data set dump operation causes the enqueue on a data set to be held until all data sets are dumped. DFSMSdss does not reset the data set change indicator until after all data sets are dumped. This may be of particular interest when dumping user catalogs. That is because delays for other jobs that need access to the user catalog may be caused by DFSMSdss holding the enqueue for the user catalog until all the data sets are dumped. This may cause a lockout condition when another job is dumping the same catalog at the same time.
5. You may specify both CONCURRENT and RESET, but RESET is ignored and a warning message is printed unless your installation uses a patch to tell DFSMSdss to accept RESET with CONCURRENT and to reset the data set changed indicator after the concurrent copy initialization is complete. If your installation uses the patch, it is possible that the data set changed indicator is left reset (off) even when the dump of the data set is not successful. If this is unacceptable, do not specify RESET with CONCURRENT.

Related reading: For additional information about DFSMSdss patches, see the *z/OS DFSMSdss Storage Administration Guide*.

&

RSA

&



&

RSA

&

&

&

&

The RSA keyword allows you to specify the label of an existing RSA public key that is present in the ICSF PKDS. The RSA public key is used to encrypt a randomly generated data key, so that the encrypted data key can be stored on the output medium.

&

&

&

ICSF only allows labels for RSA keys to be up to 64 characters long. The first character must be alphabetic or a national character (#, \$, @). The remaining characters may be alphabetic, numeric, national, or a period.

&

Note:

1. You can also specify the label of an RSA public/private key pair. ICSF uses the public key when encrypting the data key.
2. The RSA keyword cannot be specified with the KEYPASSWORD keyword.
3. When using ENCTDES, or running on z800/z900 hardware, ensure that the RSA key is an internal key. Under these scenarios, an external RSA key will not be accepted by ICSF during the restore of the data.

&

ENCRYPT

&

&
&
&
&
&
&
&
&
&
&
&

The ENCRYPT keyword allows you to specify the type of encryption key and the type of encryption that DFSMSdss performs on the dumped data. You can specify one of the following options. If you do not specify the ENCRYPT keyword, CLRAES128 is the default. If you specify ENCRYPT with the RSA keyword, the data key is randomly generated for each DUMP command.

- CLRTDES - This option specifies that the dumped data is encrypted with a clear triple-length DES key.
- CLRAES128 - This option specifies that the dumped data is encrypted with a clear 128-bit AES key
- ENCTDES - This option specifies that the dumped data is encrypted with a secure triple-length DES key.

SELECTMULTI

See “LOGINDYNAM” on page 139.

SHARE



SHARE specifies that DFSMSdss is to share the data sets to be dumped for read access with other programs. Do not specify the DELETE, RESET, or UNCATALOG keyword if you specify SHARE. Use SHARE carefully to ensure that the contents of the dumped copy of the data set are valid.

Restriction: You cannot use the SHARE and FULL keywords at the same time.

Notes:

1. Unlike the RESTORE command, the DUMP command honors the SHARE keyword for VSAM data sets. However, the SHARE keyword is only honored for VSAM data sets that were defined with share options other than (1,3) or (1,4). Specifying the SHARE keyword does not cause DFSMSdss to honor the share options that are defined for VSAM data sets. For VSAM data sets that are defined with share options other than (1,3) or (1,4), specifying the SHARE keyword allows other programs to obtain read access, but not write access, to the data sets while they are being dumped. For VSAM data sets that are defined with share options (1,3) or (1,4), neither read access nor write access by other programs is allowed while the data set is being dumped, regardless of whether SHARE was specified.
2. Do not use the SHARE keyword during a physical dump of HFS or zFS data sets.
3. The SHARE keyword is required to logically dump mounted HFS data sets in DFSMSdss releases prior to DFSMSdss Release 1.5. The SHARE keyword is no longer required to logically dump mounted HFS data sets beginning in DFSMSdss Release 1.5.
4. For an HFS data set, DFSMSdss obtains both an ADRDSN enqueue and a SYSZDSN enqueue. SHARE determines only whether the ADRDSN enqueue is shared or exclusive.
5. For a zFS data set, DFSMSdss obtains an ADRDSN enqueue, a SYSDSN enqueue, and a number of SYSVSAM enqueues. SHARE determines only whether the ADRDSN enqueue is shared or exclusive. When specifying

DUMP Command

DELETE, DFSMSdss attempts to obtain exclusive SYSDSN and SYSVSAM enqueues. If you do not specify DELETE, DFSMSdss attempts to obtain shared SYSDSN and SYSVSAM enqueues.

Related reading:

- For additional information about dumping HFS or zFS data sets, see the section about backing up data sets with special requirements in the *z/OS DFSMSdss Storage Administration Guide*.
- For additional information about dumping HFS data sets, also see “Dumping HFS Data Sets” on page 290.
- For additional information about dumping zFS data sets, see “Dumping zFS Data Sets” on page 291.

SPHERE

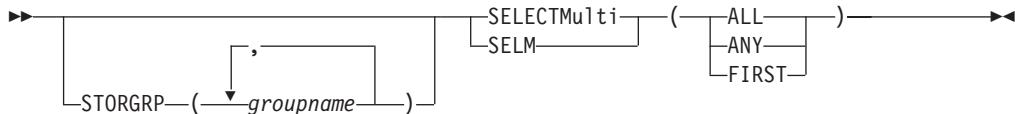


SPHERE is an option for a logical data set dump. SPHERE specifies that for any VSAM cluster dumped DFSMSdss must also dump all associated AIX clusters and paths. Individual sphere components need not be specified, only the base cluster name.

Notes:

1. The base cluster name must be specified to process the entire sphere. If the SPHERE keyword is specified but the base cluster name is not, none of the associations will be processed.
2. If an AIX is dumped without the SPHERE keyword, during a restore DFSMSdss treats the AIX as a normal VSAM KSDS.

STORGRP



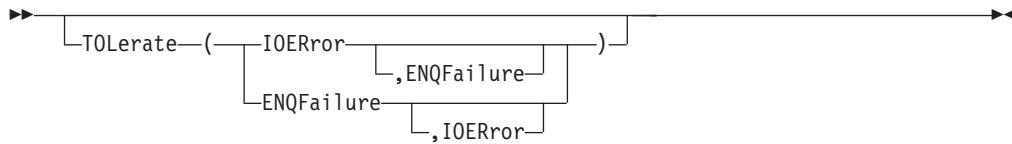
STORGRP specifies that all of the online volumes in the storage group be dynamically allocated. If a volume in the storage group is not online, that volume is not used for processing. You can specify up to 255 storage group names. Specifying STORGRP with a storage group name is equivalent to specifying LOGINDYNAM with all the online volumes in the storage group included in the list.

You can specify the STORGRP keyword with the SELECTMULTI keyword, but STORGRP cannot be used at the same time with the INDDname, INDYnam, LOGINDDname, or LOGINDYnam keywords.

See “LOGINDYNAM” on page 75 for a description of the SELECTMULTI keyword.

Notes for LOGINDDNAME, LOGINDYNAM, and STORGRP keywords:

1. If the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword is specified, DFSMSdss uses logical processing to perform the dump operation. Logical processing is also used if no input volume is specified.
2. A multivolume data set that has extents on volumes not specified with the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword will not be dumped unless you specify SELECTMULTI.

TOLERATE

ENQFailure specifies that data sets are to be processed even though shared or exclusive access fails.

TOL(ENQF) and FULL or TRACKS are mutually exclusive; you cannot specify these keywords together.

Notes:

1. Unlike PDS data sets, PDSE data sets that are open for update cannot be dumped even if TOL(ENQF) is specified.
2. If you must dump a PDSE data set and it must be open for update, process the data set with concurrent copy (specify CONCURRENT). If you cannot use concurrent copy, convert the PDSE back to PDS and then dump the PDS data set with TOL(ENQF).
3. TOL(ENQF) cannot be used when processing a **logical** dump of HFS or zFS data sets. HFS or zFS data sets cannot be dumped when the HFS or zFS data set is mounted on a system other than the system where the dump is being performed. TOL(ENQF) is not required when dumping an HFS data set or zFS data set is mounted on the same system where the dump is being performed.
4. Exercise caution if you use TOL(ENQF) during a **physical** dump of HFS data sets. Unlike other types of data sets, if an HFS data set is updated during a physical dump with TOL(ENQF), a subsequent restore can likely result in an unusable data set.

Related reading:

- For additional information about dumping HFS or zFS data sets, see the section about backing up data sets with special requirements in the *z/OS DFSMSdss Storage Administration Guide*.
- For additional information about TOL(ENQF), see Appendix B, "Data Integrity—Serialization," on page 287.

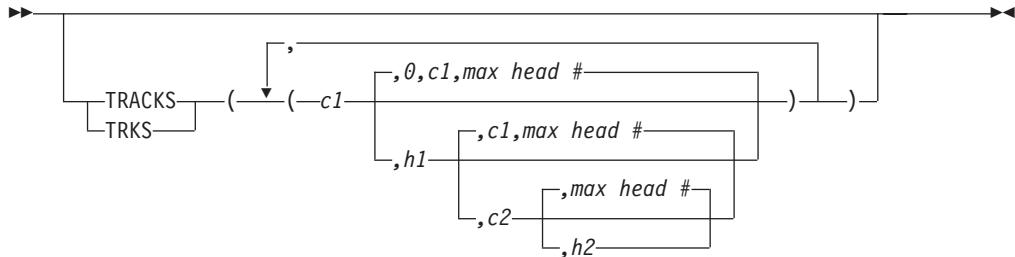
IOERror

specifies that DFSMSdss is to continue processing even though input errors occur, but is to end after 100 errors. This applies only to input errors and only to equipment check and busout parity.

Notes:

1. TOL(IOERror) is ignored if CANcelerror is specified.
2. If a permanent read error occurs, the track image record is flagged on output as having an I/O error and the dump processing continues.
3. This track is cleared in a restore operation.

TRACKS



TRACKS specifies ranges of tracks to be dumped. When you restore the data, this entire range or its subset must be specified with the RESTORE command.

Restriction: You cannot use the TRACKS keyword with the TOL(ENQF) keyword.

c1,h1 Specifies the cylinder and head number of the beginning of the range.

Specify hexadecimal numbers as X'c1' or X'h1'.

c2,h2 Specifies the cylinder and head number of the end of the range. Specify hexadecimal numbers as X'c2' or X'h2'.

Notes:

1. The *c2* must be greater than or equal to *c1*.
2. If *c2* equals *c1*, *h2* must be greater than or equal to *h1*.

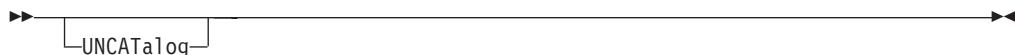
DFSMSdss verifies that the range is within the limits of the device. If you do not specify all four values for a range, DFSMSdss provides the missing values unless the omitted value causes a syntax error. No intervening values can be omitted. For example:

Specified	Results
None	Syntax error
<i>c1</i>	<i>c1,0,c1, maximum head number</i>
<i>c1,h1</i>	<i>c1,h1,c1, maximum head number</i>
<i>c1,h1,c2</i>	<i>c1,h1,c2, maximum head number</i>

<i>c1,,c2</i>	Syntax error
<i>,h1</i>	Syntax error

Related reading: For additional information about using TRACKS during a physical dump operation, see the *z/OS DFSMSdss Storage Administration Guide*.

UNCATALOG



UNCATALOG applies to physical and logical data set dump operations.

For a *physical* data set dump operation, UNCATALOG instructs DFSMSdss to uncatalog any single-volume, non-VSAM, cataloged data sets successfully dumped from the current volume.

For a *logical* data set dump operation, UNCATALOG instructs DFSMSdss to uncatalog any successfully dumped single or multivolume non-VSAM data sets that are currently cataloged. (For VSAM or SMS-managed data sets, use the DELETE keyword.)

Notes:

1. UNCATALOG is ignored for VSAM and SMS-managed data sets.
2. Do not specify UNCATALOG with CONCURRENT, because after the concurrent copy operation has begun, the original data can still be updated.
3. For a logical data set dump operation, the use of the UNCATALOG keyword causes the enqueue in a data set to be held until all data sets are dumped. DFSMSdss does not uncatalog the data set until after all data sets are dumped.
4. Any non-SMS, non-VSAM data set that has a high-level qualifier of SYS1 cannot be uncataloged unless PROCESS(SYS1) is specified.

Related reading: For information about a patch to modify the UNCATALOG algorithm, see the appendix in *z/OS DFSMSdss Storage Administration Guide*.

VALIDATE



VALIDATE on a logical data set dump, specifies that all indexed VSAM data sets are to be validated as they are dumped. If the NOVALIDATE keyword is specified, the indexed VSAM data sets are dumped without validation. VALIDATE is the default.

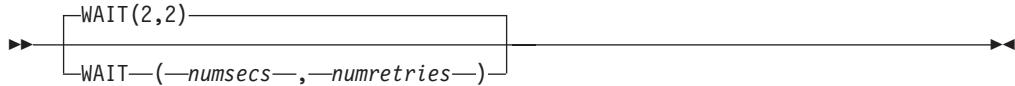
Notes:

1. If an indexed VSAM data set is dumped using VALIDATE, it must be restored on a system that supports VALIDATE. Otherwise, an error message is issued, and the restore fails.
2. Do not specify the NOVALIDATE keyword when processing VSAM extended-format or extended-addressable data sets.

DUMP Command

3. When a data set is restored, the free space in the control areas and control intervals are reset to the values in the catalog entry. You can override the values in the catalog entry by specifying the FREESPACE keyword on the restore.
4. Use the NOVALIDATE keyword on the DUMP command if you wish to restore to a DFDSS Version 2 Release 5 system that does not have the appropriate VALIDATE support, or to any level of DFDSS prior to Release 2 Version 5.

WAIT



For Physical Data Set Dump Processing: WAIT specifies to DFSMSdss the length of a wait and the number of retries to obtain control of a data set.

numsecs Specifies a decimal number from 0 to 255 that designates the interval, in seconds, between retries.

numretries Specifies a decimal number from 0 to 99 that designates the number of times DFSMSdss must retry to gain control of a data set.

For Logical Data Set Dump Processing: WAIT specifies to DFSMSdss the length of wait and the number of passes to obtain control of a data set.

numsecs Specifies a decimal number from 0 to 255 that designates the interval, in seconds, to wait before attempting another pass through the list of selected data sets.

numretries Specifies a decimal number from 0 to 99 that designates the number of passes to make through the list of selected data sets. Each pass is an attempt to obtain control of a data set.

The default for *numsecs,numretries* is (2,2), which specifies two retries at 2-second intervals. If you do not want to wait for a data set, specify 0 for either numsecs or numretries.

For logical data set dump operation, the WAIT keyword has a different meaning when: (1) data sets are being serialized, (2) multiple data sets are being processed, and (3) WAIT(0,0) is not specified. In this case, DFSMSdss makes multiple passes through the list of selected data sets. On each pass, DFSMSdss processes the data sets that (1) can be serialized without waiting for the resource and (2) were not processed before. At the end of a pass, if none of the data sets could be processed without waiting for a resource, then, in the next pass, at the first occurrence of a data set that was not processed, a WAIT will be issued. That data set and the remainder of the list will be processed if possible. The above procedure will be repeated until all data sets are processed or the WAIT limits are reached. For example, if WAIT(3,10) is specified and 5 data sets are left to be processed, up to 10 passes are made. On each pass, an unprocessed data set is waited upon for 3 seconds. Thus, only a 30-second maximum will ever be waited, not 150 (5 times 3 times 10).

Note: The WAIT keyword does not control wait/retry attempts for system resources (such as the VTOC and the VVDS). For system resources, the default wait time is 3 seconds and the default retry count is 30. This results in a total wait time of 90 seconds.

Related reading: For additional information about controlling the wait/retry attempts for system resources, see the *z/OS DFSMSdss Storage Administration Guide*

Data Integrity Considerations for Full or Tracks Dump Operation

For a full or tracks dump operation, DFSMSdss serializes the VTOC to preclude DADSM functions (such as ALLOCATE, EXTEND, RENAME, and SCRATCH) from changing the contents of the VTOC on the volume during the dump operation. Data sets are *not* serialized on these full or tracks operations. Therefore, some data sets might be opened by other jobs during the dump operation, resulting in copies of partially updated data sets. You can minimize this possibility by performing the dump operation when there is low system activity.

Full data integrity can only be guaranteed by performing dump operations by data set when TOL(ENQF) or SHARE keywords are not specified.

Format of the Output Data Set

Refer to *z/OS DFSMSdss Storage Administration Guide* for the format of the output data set.

Examples of Full and Tracks Dump Operations

In the following example, data from DASD volume 111111 is to be dumped to the first data set of standard label tape volumes TAPE01 and TAPE02.

The command input to be substituted for a full and tracks dump are shown below in Example 1A and Example 1B, respectively. To restore the same volume, refer to Examples 1, 1A, and 1B of the RESTORE command.

Example 1: A Data Set Dump

```
//JOB1      JOB    accounting information,REGION=nnnnK
//STEP1     EXEC   PGM=ADRDSU
//SYSPRINT DD     SYSOUT=A
//DASD      DD     UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//TAPE      DD     UNIT=(3480,2),VOL=SER=(TAPE01,TAPE02),
//          LABEL=(1,SL),DISP=(NEW,CATLG),DSNAME=USER2.BACKUP
//SYSIN     DD     *
          command input (See Examples 1A and 1B below)
/*
```

Example 1A: A Full Dump Operation

```
DUMP INDDNAME(DASD) OUTDDNAME(TAPE)
```

Example 1B: A Tracks Dump Operation

```
DUMP TRACKS(1,0,1,5) INDDNAME(DASD) -
          OUTDDNAME(TAPE)
```

DUMP Command

Example 1C: Full Volume Dump Operation with CONCURRENT

```
//DSSJOB JOB      accounting information,REGION=nnnnK
//DUMPSTEP EXEC  PGM=ADRDSU
//SYSPRINT DD    SYSOUT=*
//DASD    DD    UNIT=SYSDA,VOL=SER=(SSDASD),DISP=OLD
//TAPE    DD    UNIT=TAPE,VOL=SER=(TAPE01,TAPE02,TAPE03),LABEL=(1,SL),
//    DISP=(NEW,KEEP),DSN=USER.BACKUP
//SYSIN   DD    *
      DUMP FULL INDDNAME(DASD) OUTDDNAME(TAPE) -
                  COMPRESS CONCURRENT
/*
```

This JCL does a DFSMSdss full-volume dump using concurrent copy. This job continues (with a warning message) if concurrent copy initialization fails. No special action is required to perform a restore operation after a concurrent copy dump operation.

Examples of Physical Data Set Dump Operations

Example 2 depicts specified data sets on DASD volumes (numbered 111111 and 222222) that are being dumped to the first data set of standard label tape volume called TAPE02.

Examples 2A through 2G below complement examples 2A through 2D in the restore section, in any combination; for example, the dump tape produced in example 2C can be used as the input tape for example 2A under the RESTORE command.

Example 2: Depicting DASD Volume DUMP

```
//JOB2    JOB      accounting information,REGION=nnnnK
//STEP1   EXEC  PGM=ADRDSU
//SYSPRINT DD    SYSOUT=A
//DASD1   DD    UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//DASD2   DD    UNIT=3380,VOL=(PRIVATE,SER=222222),DISP=OLD
//TAPE    DD    UNIT=3480,VOL=SER=TAPE02,
//    LABEL=(1,SL),DISP=(NEW,CATLG),DSNAME=USER2.BACKUP
//SYSIN   DD    *
      command input (see Examples 2A, 2B, ... 2G that follow)
/*
```

Example 2A: Using the INCLUDE Subkeyword

```
DUMP INDDNAME(DASD1,DASD2) OUTDDNAME(TAPE) -
      DATASET(INCLUDE(USER2.**,USER3.*))
```

Example 2B: Using the INCLUDE and EXCLUDE Subkeywords

```
DUMP INDDNAME(DASD1,DASD2) OUTDDNAME(TAPE) -
      DATASET(INCLUDE(USER2.**,USER3.*)) -
      EXCLUDE(USER2.**.REP))
```

Example 2C: Using the INCLUDE, EXCLUDE, and BY Subkeywords

```
DUMP INDDNAME(DASD1,DASD2) OUTDDNAME(TAPE) -
      DATASET(INCLUDE(USER2.**,USER3.*)) -
      EXCLUDE(USER2.**.REP) -
      BY((DSCHA,EQ,1))
```

Example 2D: With Filtering Data in a Data Set

```
DUMP INDDNAME(DASD1,DASD2) OUTDDNAME(TAPE) -
      DATASET(FILTERDD(A1))
```

Note: The following DD statement must be added to the JCL shown above:

```
//A1 DD DSNAME=USER2.FILTER,DISP=SHR
```

This cataloged data set (USER2.FILTER) contains three card-image records. The information shown is positioned in columns 2 through 72 of each record:

```
INCLUDE(USER2.**,USER3.*)) -
      EXCLUDE(USER2.**.REP) -
      BY((DSCHA,EQ,1))
```

Example 2E: With Passwords in the Input Stream

```
DUMP INDDNAME(DASD1,DASD2) OUTDDNAME(TAPE) -
      DATASET(INCLUDE(USER2.**,USER3.*)) -
      PASSWORD(USER2.ABC.DEF/PSWD1,USER2.XYZ/PSWD2)
```

Example 2F: With Passwords in a Data Set

```
DUMP INDDNAME(DASD1,DASD2) OUTDDNAME(TAPE) -
      DATASET(INCLUDE(USER2.**,USER3.*)) -
      PASSWORD(PDD)
```

Note: The following DD statement must be added to the JCL shown above:

```
//PDD DD DSNAME=USER2.PASSWORD,DISP=SHR
```

This cataloged data set (USER2.PASSWORD) contains a single card-image record. The information shown is positioned in columns 2 through 72:

```
USER2.ABC.DEF/PSWD1,USER2.XYZ/PSWD2
```

DUMP Command

Example 2G: Wait for Data Sets if They or Other Data Sets with the Same Name are in Use by Other Jobs

```
DUMP INDDNAME(DASD1) OUTDDNAME(TAPE) -
      DATASET(INCLUDE(**)) -
      WAIT(1,99)
```

If a data set is in use, DFSMSdss waits for one second, then tries to gain access to the resource again. This is done as many as 99 times for each data set.

Example 2H: Clearing Volumes of Uncataloged Data Sets

```
DUMP DATASET(INCLUDE(**)) -
      BY((DSORG NE VSAM) -
          (CATLG EQ NO))) -
      INDDNAME(DASD1,DASD2) -
      OUTDDNAME(TAPE) -
      DELETE PURGE
```

If you do not want a dump of the uncataloged data sets, the DD named TAPE can be a dummy. DASD1 and DASD2 identify the input volumes. A physical data set dump can handle multiple uncataloged single-volume data sets with the same name if multiple volumes are specified. This is because each volume is processed in order, one at a time. The dump cannot handle a multivolume data set even if all the volumes on which it resides are specified as input volumes.

Examples of Logical Data Set Dump Operations

This section contains examples of logical data set dump operations.

Example 1: Dumping Data Sets Constantly in Use

```
//JOB1    JOB accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSU
//SYSPRINT DD SYSOUT=A
//DASD1   DD UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//DASD2   DD UNIT=3380,VOL=(PRIVATE,SER=222222),DISP=OLD
//TAPE    DD UNIT=3480,VOL=SER=TAPE02,
//        LABEL=(1,SL),DISP=(NEW,CATLG),DSNAME=USER2.BACKUP
//SYSIN   DD *
DUMP INDDNAME(DASD1) OUTDDNAME(TAPE) -
      DATASET(INCLUDE(**)) TOL(ENQF) WAIT(0,0)
/*
```

DFSMSdss does not wait (WAIT(0,0)) if a data set is in use. Instead, it processes the data set without serialization or enqueueing (TOL(ENQF)).

Example 2: Dumping a User Catalog and its Aliases

To dump a user catalog, you perform a logical data set dump with the fully qualified user catalog name as the data set name. No filtering is allowed. If the user catalog has any aliases, the aliases are automatically dumped.

```
//JOB2    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=A
//DASD1   DD  UNIT=3380, VOL=(PRIVATE,SER=111111),DISP=OLD
//TAPE    DD  UNIT=3480, VOL=SER=TAPE02,
//  LABEL=(1,SL),DISP=(NEW,CATLG),DSN=USER2.BACKUP
//SYSIN   DD  *
      DUMP OUTDDNAME(TAPE) -
      DS(INCLUDE(MY.USER.CAT))
/*

```

Example 3: Logical Data Set Dump Operation with Catalog Filtering

```
//JOB3    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSU
//TAPE    DD  UNIT=3480, VOL=SER=TAPE04,
//  LABEL=(1,SL),DISP=(NEW,CATLG),DSN=USER3.BACKUP
//SYSPRINT DD  SYSOUT=A
//SYSIN   DD  *
      DUMP OUTDD(TAPE) -
      DS(INCL(USER1.**))
/*

```

All data sets cataloged in the standard search order whose first-level qualifier is USER1 are to be dumped. Because some of these data sets are multivolume, source DASD volumes are not specified, resulting in data set selection by catalog.

Example 3 can be modified as follows to dump only data sets changed since the last backup. In addition, data sets that end with a qualifier of LISTING are not to be dumped (EXCL(**.LISTING)).

```
//SYSIN   DD  *
      DUMP OUTDD(TAPE) -
      DS(INCL(USER1.**)) -
      EXCL(**.LISTING) -
      BY((DSCHA EQ 1)))
/*

```

Example 4: Logical Data Set Dump Operation with VTOC Filtering

```
//JOB4    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=A
//DASD1   DD  VOL=SER=338001,UNIT=3380,DISP=OLD
//TAPE    DD  UNIT=3480, VOL=SER=TAPE04,
//  LABEL=(1,SL),DISP=(NEW,CATLG),DSN=USER3.BACKUP
//SYSIN   DD  *
      DUMP DATASET(INCLUDE(USER3.**)) -
      LOGINDDNAME(DASD1) -
      OUTDDNAME(TAPE) -
      DELETE PURGE
/*

```

DUMP Command

All data sets on volume 338001 whose first qualifier is USER3 are included in a logical data set DUMP. DFSMSdss filters using the VTOC of volume 338001. Catalogs are also used, as needed, for multivolume and VSAM data sets.

The previous example can be modified as follows to dynamically allocate volume 338001 and to do SELECTMULTI processing. All single volume data sets on volume 338001 are included in a logical data set dump operation. SELECTMULTI(ANY) specifies that a cataloged data set residing on volumes 338001, 338003, and 338005 will be dumped even though 338003 and 338005 are not in the LOGINDYNAM volume list.

```
//JOB4      JOB  accounting information,REGION=nnnnK
//STEP1      EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=A
//TAPE       DD  UNIT=3480,VOL=SER=TAPE04,
//  LABEL=(1,SL),DISP=(NEW,CATLG),DSN=USER3.BACKUP
//SYSIN      DD  *
          DUMP DATASET(INCLUDE(**)) -
          SELECTMULTI(ANY) -
          LOGINDYNAM(338001) -
          OUTDDNAME(TAPE) -
          DELETE PURGE
/*
```

Example 5: Logical Data Set Dump Operation for Storage Management Subsystem (SMS)

```
//JOB5      JOB  accounting information,REGION=nnnnK
//STEP1      EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=A
//TAPE       DD  DSN=BACKUP(+1),DISP=(,CATLG),
//  DCB=(SYS1.DFDSS.DSCB)
//SYSIN      DD  *
          DUMP LOGINDYNAM(338001) -
          SELECTMULTI(FIRST) -
          DATASET(INCLUDE(**)) -
          OUTDDNAME(TAPE) -
          DELETE
/*
```

This example backs up the volume to a generation data set. You can use generation data set groups to create and manage multiple backup versions of a volume or data sets.

A volume is backed up for converting to or from SMS management by using a logical data set dump function.

Example 6: Logical Dump Operation with CONCURRENT

```
//JOB6      JOB  accounting information,REGION=nnnnK
//DUMPSTEP EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=*
//TAPE       DD  UNIT=TAPE,VOL=SER=(TAPE01,TAPE02,TAPE03),LABEL=(1,SL),
//  DISP=(NEW,KEEP),DSN=USER.BACKUP
//SYSIN      DD  *
          DUMP DATASET(INCLUDE(USER.LOG,USER.TABLE,USER.XREF)) -
          OUTDDNAME(TAPE) OPTIMIZE(4) CONCURRENT
/*
```

This JCL does a DFSMSdss logical data set dump of three fully qualified data sets using concurrent copy. This job continues with a warning message if concurrent copy initialization fails. No special action is required to perform a restore operation after a concurrent copy dump operation.

Example 7: Clearing Volumes of Uncataloged Data Sets

```
//JOB7    JOB accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSSU
//SYSPRINT DD SYSOUT=A
//DASD1   DD VOL=SER=MYVOL1,UNIT=SYSDA,DISP=OLD
//DASD2   DD VOL=SER=MYVOL2,UNIT=SYSDA,DISP=OLD
//TAPE    DD UNIT=3480,VOL=SER=TAPE04,LABEL=(1,SL),
//        DISP=(NEW,CATLG),DSN=USER3.BACKUP
//SYSIN   DD *
      DUMP DATASET(INCLUDE(**) -
                  BY((DSORG NE VSAM) -
                      (CATLG EQ NO))) -
                  LOGINDDNAME(DASD1,DASD2) -
                  OUTDDNAME(TAPE) -
                  DELETE PURGE
/*

```

Logical data set dump cannot be used to dump SMS data sets that are not catalogued or are catalogued outside the standard order of search. DFSMSdss physical data set dump or IDCAMS DELETE NVR can be used for such cleanup operations.

If you do not want a dump of the uncatalogued data sets, the DD named TAPE can be a dummy. DASD1 and DASD2 identify the input volumes. A logical data set dump cannot handle multiple uncatalogued data sets with the same name in the same job even if all the volumes on which they reside are specified as input volumes.

A logical dump can handle a legitimate multivolume uncatalogued data set if all the volumes on which it resides are specified as input volumes and if no catalogued data set by the same name exists on the system.

PRINT Command

With the PRINT command, you can print:

- A single-volume non-VSAM data set, as specified by a fully qualified name. You must specify the volume where the data set resides, but you do not need to specify the range of tracks it occupies.
- A single-volume VSAM data set component (not cluster). The component name specified must be the name in the VTOC, not the name in the catalog.
- Ranges of tracks.
- All or part of the VTOC. The VTOC location need not be known.

Note: In order to print a multivolume data set, multiple PRINT commands with the appropriate INDD/INDY keywords must be used.

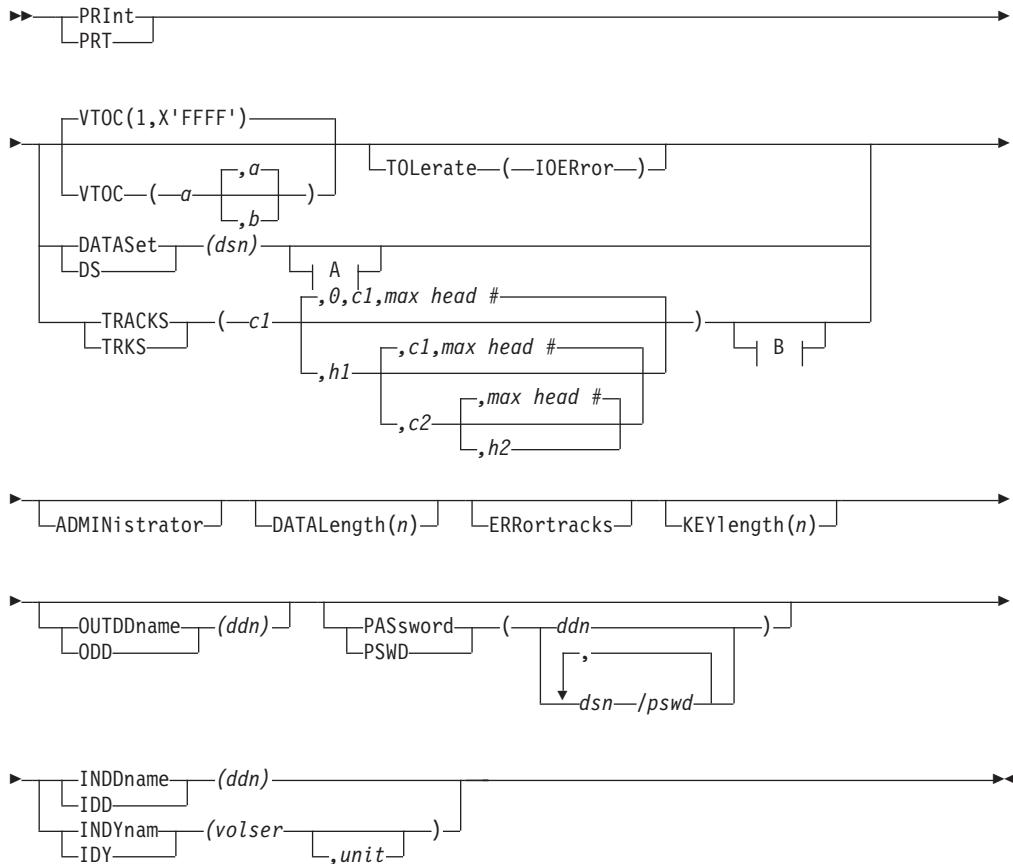
Unless the ALLDATA keyword is specified, only the used space is printed for sequential or partitioned data sets or data sets with data set organizations of null.

PRINT Command

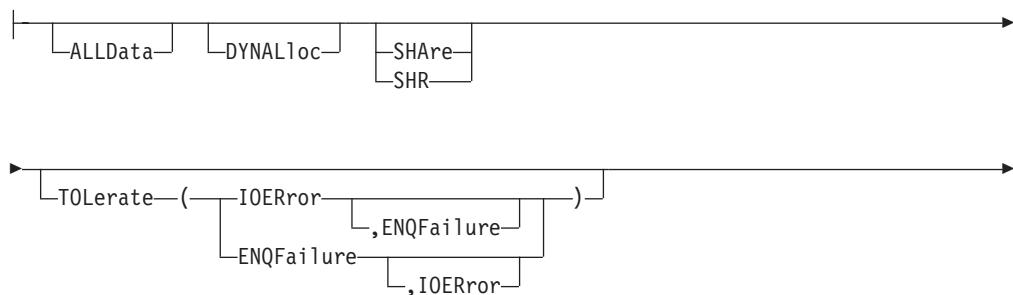
If an error occurs in reading a record, DFSMSdss attempts to print the record in error. You can print all requested tracks or just a subset of the tracks that have data checks.

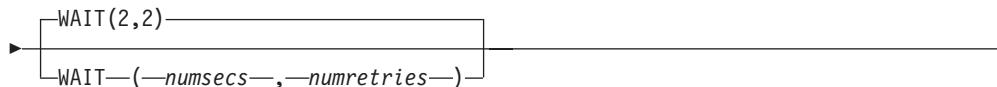
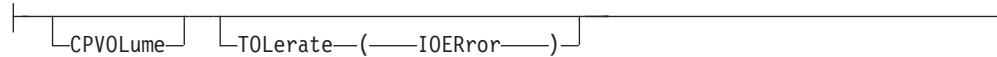
Related reading: For additional information about authorization checking, see Chapter 6, “Data Security and Authorization Checking,” on page 255.

PRINT Syntax

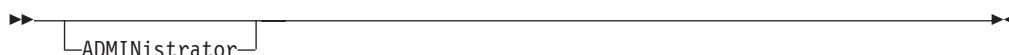


A: Optional Keywords with PRINT DATASET:



**B: Optional Keywords with PRINT TRACKS:****Explanation of PRINT Command Keywords**

This section describes the keywords for the PRINT command.

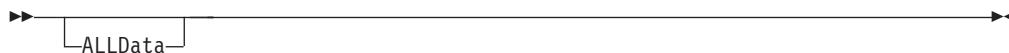
ADMINISTRATOR

ADMINISTRATOR lets you act as a DFSMSdss-authorized storage administrator for the PRINT command. If you are not authorized to use the ADMINISTRATOR keyword, the command is ended with an error message. Otherwise, access checking to data sets and catalogs is bypassed.

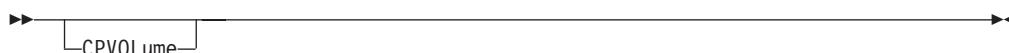
To use the ADMINISTRATOR keyword all of the following must be true:

- FACILITY class is active.
- The applicable FACILITY-class profile is defined.
- You have READ access to that profile.

Related reading: For additional information about the use of the ADMINISTRATOR keyword, see “ADMINISTRATOR Keyword” on page 265.

ALLDATA

ALLDATA specifies, when the DATASET keyword is also specified, that *all* allocated space in the data set is to be printed.

CPVOLUME

CPVOLUME specifies that the volume is VM-formatted and that the OS-compatible VTOC must begin on track zero, record five. The OS-compatible VTOC does not describe the extents of any data on the volume. Therefore, you must specify the track ranges to be printed with the TRACKS keyword. CPVOLUME is only allowed with the ADMINISTRATOR keyword because DFSMSdss cannot check access authorization for VM data.

PRINT Command

DATALENGTH

```
►► [DATALength] (—n—) ►►
```

- n* Specifies the logical length, in decimal format, of the data portion of a record. It is used only if the count field of a record on any track has a data check.

DATASET

```
►► [DATASet] (—dsn—) ►►
```

- dsn* Specifies the fully qualified name of the data set to be printed. The data set is printed in logical sequence.

Note: Data set filtering is not allowed with the PRINT command.

DYNALLOC

```
►► [DYNALloc] ►►
```

DYNALLOC specifies dynamic allocation, instead of enqueue, to serialize the use of data sets. This allows cross-system serialization in a JES3/MVS environment.

Consider:

- The serialization is of value only when the dynamic allocation/JES3 interface is not disabled.
- Run time increases when you use the DYNALLOC keyword to serialize data sets (as opposed to enqueue) because overhead is involved in dynamic allocation and serialization across multiple processors.

ERRORTRACKS

```
►► [ERRortracks] ►►
```

ERRORTRACKS specifies that only tracks on which data checks occur are to be printed.

INDDNAME

```
►► [INDDname] (—ddn—) ►►
```

- ddn* Specifies the name of the DD statement that identifies a volume that contains the data set, range of tracks, or the VTOC to be printed. If you want to print a multivolume data set, you must print one volume at a time.

INDYNAM

```
► [INDYnam] (volser [,unit]) ►
```

INDYNAM specifies that the volume that contains the data set, range of tracks, or the VTOC to be printed is to be dynamically allocated. The volume must be mounted and online. You cannot specify a nonspecific volume serial number using an asterisk (*).

Using INDYNAM instead of DD statements to allocate DASD volumes does not noticeably increase run time and permits easier coding of JCL and command input.

volser Specifies the volume serial number of a DASD volume to be printed.

unit Specifies the device type of a DASD volume to be printed. This parameter is optional.

KEYLENGTH

```
► [KEYlength(n)] ►
```

n Specifies the key length, in decimal format, of a record. It is used only if the count field of a record on any track has a data check.

OUTDDNAME

```
► [OUTDDname(ddn)] ►
```

ddn Specifies the name of the DD statement that identifies the (output) print data set. Each of the DD statements corresponding to a DDNAME (ddn) must identify only one volume serial number. If this keyword is not specified, the default is SYSPRINT.

Notes:

1. If the DCB keyword LRECL is specified on the DD statement, it must be in the range of 84 to 137 inclusive. If BLKSIZE is specified, it must be at least four greater than the LRECL.
2. If an LRECL less than 84 is chosen, the return code is 8 and an error message is issued.
3. If the specified LRECL is greater than 137, LRECL and BLKSIZE are set to 137 and 141, respectively.

PASSWORD

```
► [PASSword(ddn,dsn—/pswd)] ►
```

PRINT Command

PASSWORD specifies the passwords DFSMSdss is to use for password-protected data sets. (Password checking is bypassed for RACF-protected data sets.) This is required only if:

- You do not have the required volume-level RACF DASDVOL or RACF DATASET access.
- The installation authorization exit does not bypass the checks.
- You do not want to be prompted for the password for VSAM data sets.

Note: You should specify the passwords for all data sets that do not have RACF protection but do have password protection. During processing, a utility invoked by DFSMSdss may have to prompt the operator for a password. You can control authorization checking by using the installation authorization exit.

For VSAM data sets, passwords are checked at the cluster level only.

Note: The PASSWORD keyword is not valid for a PRINT VTOC command.

ddn Specifies the name of the DD statement that identifies the sequential data set, or member of a partitioned data set, that contains data set names and their passwords. This data set must contain card-image records in DFSMSdss command syntax format.

dsn/pswd *dsn* is a fully qualified data set name. *pswd* is its password. If no password follows the slash (/), *dsn* is treated as though it were *ddn*.

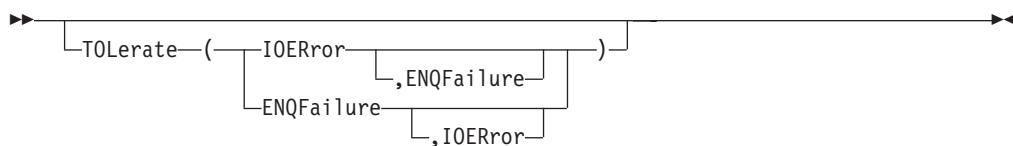
Printing of actual data set passwords specified in your input command stream is suppressed in the SYSPRINT output.

SHARE



SHARE specifies that DFSMSdss is to share, for read access with other programs, the data set that is to be printed.

TOLERATE

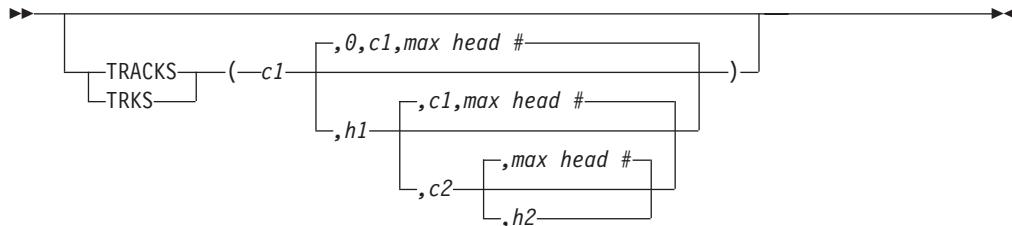


ENQFailure Specifies that data sets are to be processed even though shared or exclusive access fails. TOLERATE(ENQFAILURE) is ignored if it is specified in a PRINT TRACKS or PRINT VTOC operation.

IOError Specifies that DFSMSdss is to continue processing even though I/O errors occur, but is to end after 100 errors.

Related reading: For additional information about TOL(ENQF), see Appendix D, “Examples of the Application Program with the User Interaction Module (UIM),” on page 345.

TRACKS



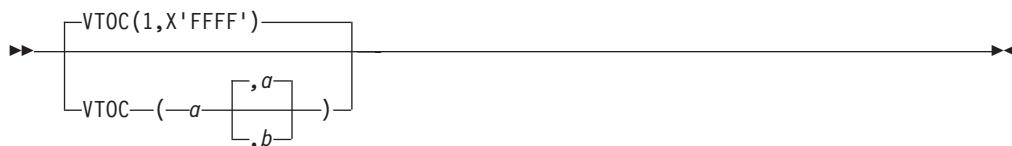
TRACKS specifies ranges of tracks to be printed.

- c1,h1* Specifies the cylinder and head number of the beginning of the range. Specify hexadecimal numbers as X'c1' or X'h1'.
- c2,h2* Specifies the cylinder and head number of the end of the range. Specify hexadecimal numbers as X'c2' or X'h2'. The *c2* must be greater than or equal to *c1*. If *c2* equals *c1*, *h2* must be greater than or equal to *h1*.

DFSMSSdss verifies that the range is within the limits of the device. If you do not specify all four values for a range, DFMSdss provides the missing values unless the omitted value causes a syntax error. No intervening values can be omitted. For example:

Specified	Results
None	Syntax error
<i>c1</i>	<i>c1,0,c1, maximum head number</i>
<i>c1,h1</i>	<i>c1,h1,c1, maximum head number</i>
<i>c1,h1,c2</i>	<i>c1,h1,c2, maximum head number</i>
<i>c1,c2</i>	Syntax error
<i>,h1</i>	Syntax error

VTOC



VTOC specifies that all or part of the VTOC is to be printed. Omitting the VTOC, DATASET, and TRACKS keywords causes the entire VTOC to be printed. Part of the VTOC can be printed by specifying:

VTOC(a,b)

where *a* and *b* are the relative track numbers (1 is the first track) of the first and last tracks to be printed. The value of *b* must be equal to or greater than the value of *a* and equal to or less than 65535 (X'FFFF'). Either of these numbers can be specified in decimal or hexadecimal. To specify a hexadecimal number, code X'nn'.

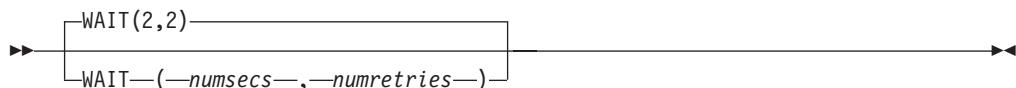
PRINT Command

If only the first value (a) is specified, only that track is printed.

If the second value (b) is greater than the relative track number of the last physical track of the VTOC, DFSMSdss prints up to and including the last track of the VTOC. Therefore, another way of printing the entire VTOC would be to specify: VTOC(1,X'FFFF')

Note: The PASSWORD keyword is not valid for a PRINT VTOC command.

WAIT



WAIT specifies to DFSMSdss the length of a wait in seconds and the number of retries to obtain control of a data set.

numsecs Specifies a decimal number from 1 to 255 that designates the interval, in seconds, between retries.

numretries Specifies a decimal number from 0 to 99 that designates the number of retries to gain control of a data set.

The default for *numsecs,numretries* is (2,2), which specifies two retries at 2-second intervals. If you do not want to wait for a data set, specify 0 for either numsecs or numretries.

Note: The WAIT keyword does not control wait/retry attempts for system resources (such as the VTOC and the VVDS). For system resources, the default wait time is 3 seconds and the default retry count is 30. This results in a total wait time of 90 seconds.

Related reading: For more information about controlling the wait/retry attempts for system resources, see the *z/OS DFSMSdss Storage Administration Guide*.

Examples of Print Operations

The following are examples of the PRINT command.

Example 1: Printing a Range of Tracks

```
//JOB2    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSU
//SYSPRINT DD   SYSOUT=A
//DASD     DD   UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//SYSIN   DD   *
      PRINT  TRACKS(1,0,1,5) INDDNAME(DASD)
/*
```

Prints a hard copy of tracks 0 through 5 from cylinder 1 on volume 111111.

Example 2: Printing a Component of a Virtual Storage Access Method (VSAM) Cluster

```
//JOB3      JOB    accounting information,REGION=nnnnK
//STEP1     EXEC   PGM=ADRDSU
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
      PRINT INDYNAM(338000) /* ALLOC VOL 338000 DYNAMICALLY */ -
                  DS(PARTS.VSAM1.INDEX) /* DATA SET THAT HAS BAD TRACK */ -
                  WAIT(0,0)           /* DO NOT WAIT IF ENQ FAILS */ -
                  TOL(ENQF)           /* IGNORE ENQ FAILURES */ -
                  PSWD(PARTS.VSAM1/USERPSWD) /* PASSWORD FOR CLUSTER */ -
/*
/*
```

A component of a VSAM data set is to be printed. The printing proceeds even if the component cannot be serialized (enqueued).

Example 3: Printing a Data Set

```
//STEPT003 EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
      PRINT DATASET(PUBSEXMP.SAM.S01) INDYNAM(D9S060)
/*
/*
```

The output shown in Figure 2 on page 166 was produced from the example above.

RELEASE Command

```
PAGE 0001      5695-DF175  DFMSDSS V2R10.0 DATA SET SERVICES    1999.211 14:55
PRINT          -
DATASET(PUBSEXMP.SAM.S01) -
INDYNAME(D9S060)
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'PRINT '
ADR109I (R/I)-RI01 (01), 1999.211 14:55:44 INITIAL SCAN OF USER CONTROL
STATEMENTS COMPLETED.
ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR006I (001)-STEND(01), 1999.211 14:55:44 EXECUTION BEGINS
*** TRACK(CCHH) 0000000E           R0 DATA 0000000E00000000
COUNT 0000000E01000190
0000 F0F0F0F0 F0F1D7E4 C2E2C5E7 D4D74BE2 C1D44BE2 F0F1C1C2 C3C4C5C6 C7C8C9D1
*000001PUBSEXMP.SAM.S01ABCDEFGHIJ*
0020 D2D3D4D5 D6D7D8D9 E2E3E4E5 E6E7E8E9 C1C2C3C4 C5C6C7C8 C9D1D2D3 D4D5D6D7
*KLMNOPQRSTUVWXYZABCDEFGHIJKLMNOP*
0040 D8D9E2E3 E4E5E6E7 E8E9C1C2 C3C4C5C6 F0F0F0F0 F0F2D7E4 C2E2C5E7 D4D74BE2
*QRSTUVWXYZABCDEF000002PUBSEXMP.S*
0060 C1D44BE2 F0F1C1C2 C3C4C5C6 C7C8C9D1 D2D3D4D5 D6D7D8D9 E2E3E4E5 E6E7E8E9
*AM.S01ABCDEFGHIJKLMN0PQRSTUVWXYZ*
0080 C1C2C3C4 C5C6C7C8 C9D1D2D3 D4D5D6D7 D8D9E2E3 E4E5E6E7 E8E9C1C2 C3C4C5C6
*ABCDEFGHIJKLMN0PQRSTUVWXYZABCDEF*
00A0 F0F0F0F0 F0F3D7E4 C2E2C5E7 D4D74BE2 C1D44BE2 F0F1C1C2 C3C4C5C6 C7C8C9D1
*000003PUBSEXMP.SAM.S01ABCDEFGHIJ*
00C0 D2D3D4D5 D6D7D8D9 E2E3E4E5 E6E7E8E9 C1C2C3C4 C5C6C7C8 C9D1D2D3 D4D5D6D7
*KLMNOPQRSTUVWXYZABCDEFGHIJKLMNOP*
00E0 D8D9E2E3 E4E5E6E7 E8E9C1C2 C3C4C5C6 F0F0F0F0 F0F4D7E4 C2E2C5E7 D4D74BE2
*QRSTUVWXYZABCDEF000004PUBSEXMP.S*
0100 C1D44BE2 F0F1C1C2 C3C4C5C6 C7C8C9D1 D2D3D4D5 D6D7D8D9 E2E3E4E5 E6E7E8E9
*AM.S01ABCDEFGHIJKLMN0PQRSTUVWXYZ*
0120 C1C2C3C4 C5C6C7C8 C9D1D2D3 D4D5D6D7 D8D9E2E3 E4E5E6E7 E8E9C1C2 C3C4C5C6
*ABCDEFGHIJKLMN0PQRSTUVWXYZABCDEF*
0140 F0F0F0F0 F0F5D7E4 C2E2C5E7 D4D74BE2 C1D44BE2 F0F1C1C2 C3C4C5C6 C7C8C9D1
*000005PUBSEXMP.SAM.S01ABCDEFGHIJ*
0160 D2D3D4D5 D6D7D8D9 E2E3E4E5 E6E7E8E9 C1C2C3C4 C5C6C7C8 C9D1D2D3 D4D5D6D7
*KLMN0PQRSTUVWXYZABCDEFGHIJKLMNOP*
0180 D8D9E2E3 E4E5E6E7 E8E9C1C2 C3C4C5C6
*QRSTUVWXYZABCDEF*
ADR006I (001)-STEND(02), 1999.211 14:55:44 EXECUTION ENDS
ADR013I (001)-CLTSK(01), 1999.211 14:55:44 TASK COMPLETED WITH RETURN CODE 0000
ADR012I (SCH)-DSSU (01), 1999.211 14:55:44 DFMSDSS PROCESSING COMPLETE. HIGHEST
RETURN CODE IS 0000
```

Figure 2. Output Resulting from a PRINT Command

RELEASE Command

The RELEASE command releases allocated but unused space from all eligible sequential, partitioned, and extended-format VSAM data sets that pass INCLUDE, EXCLUDE, and BY filtering criteria. The RELEASE command does not release space from guaranteed-space VSAM extended-format data sets. DFMSDss only selects data sets that have space that can be released.

DFMSDss offers two ways to process RELEASE commands:

- **Logical processing** operates on a single selected data set at a time. The data set can be multivolume. The user has the option to not specify volumes or to specify one or more volumes with the LOGDDNAME, LOGDYNAM, or STORGRP keywords. Releasing unused space from extended-format VSAM data sets requires logical processing.

If no input volumes are specified and the INCAT keyword is not specified, DFMSDss selects from all data sets cataloged in the catalogs accessible through the standard search order. For INCAT keyword details, see the INCAT keyword description in this section.

- **Physical processing** operates on all selected data sets that reside on a single volume. The user must specify one or more volumes with the DDNAME or DYNAM keywords. You cannot use physical processing to release unused space from extended-format VSAM data sets.

To process multivolume data sets, do one of the following:

- Specify no input volumes.
- Specify LOGDDNAME, LOGDYNAM, or STORGRP with an appropriate SELECTMULTI option or a volume list.
- Specify DDNAME or DYNAM with the volume or volumes on which the data set has space that can be released.

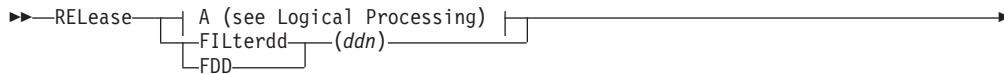
You must exclude, by using the EXCLUDE keyword, data sets whose last used block pointer in the data set's VTOC entry are not properly maintained. This can occur if you use an access method other than BSAM, QSAM, or BPAM. DFSMSdss does not release any space for data sets that are empty (the last used block pointer in the data set's VTOC entry is zero). This restriction does not apply to PDSE data sets; used space in PDSE data sets is maintained internally, rather than by reference to the data set's VTOC entry.

The following apply to the RELEASE command:

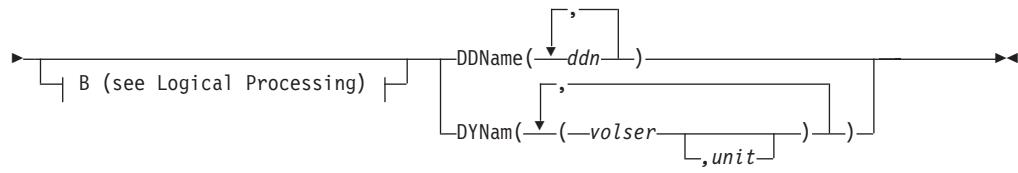
- A data set with the maximum number of extents already in use will not have any space released. For an extended-format VSAM data set, the maximum is 255 extents. For a partitioned data set extended (PDSE) and an extended-format sequential data set, the maximum is 123 extents. For other partitioned and sequential data sets, the maximum is 16 extents.
- For extended-format VSAM data sets, DFSMSdss releases space from only the data component of a base cluster or an alternate index (AIX).
- For striped VSAM data sets, DFSMSdss releases eligible space from each stripe.
- Free tracks in cylinder-allocated extents are not released. Only free cylinders are released.
- DFSMSdss excludes system data sets beginning with SYS1, unless the PROCESS keyword is used.
- DFSMSdss logical processing releases space from each volume on which an extended-format sequential data set contains data.
- DFSMSdss does not support the release of HFS data sets, as the DADSM PARTREL macro no longer supports HFS data sets.
- DFSMSdss does not support the release of zFS data sets.

Related reading: For additional information about filtering, see Chapter 2, “Filtering—Choosing the Data Sets You Want Processed,” on page 11.

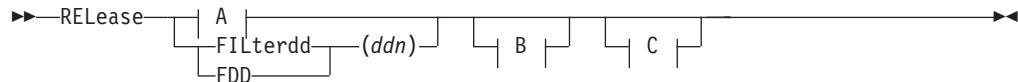
RELEASE Syntax for Physical Processing



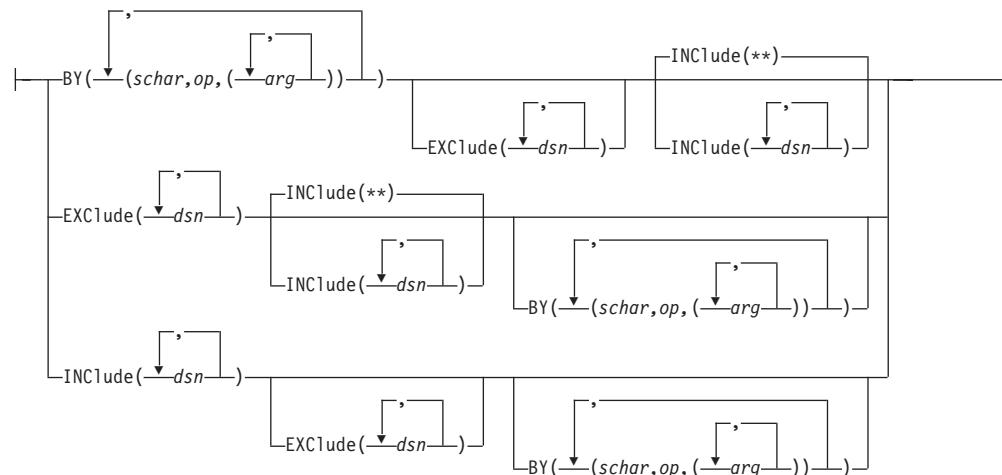
RELEASE Command



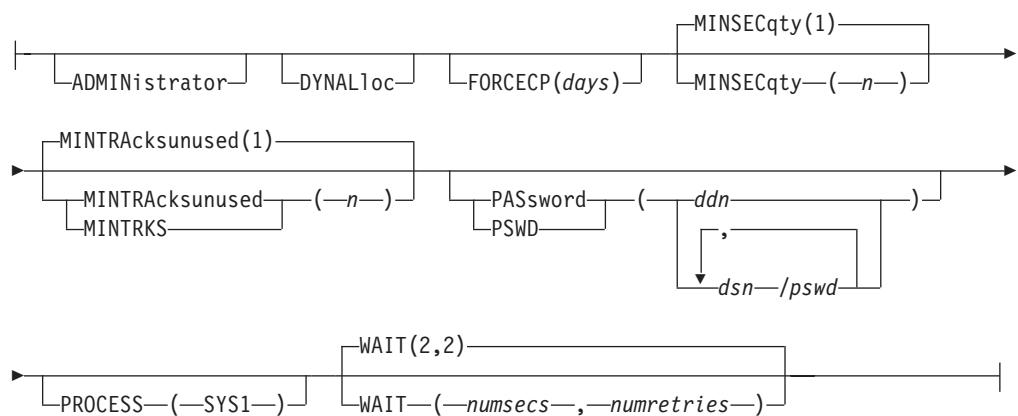
RELEASE Syntax for Logical Processing



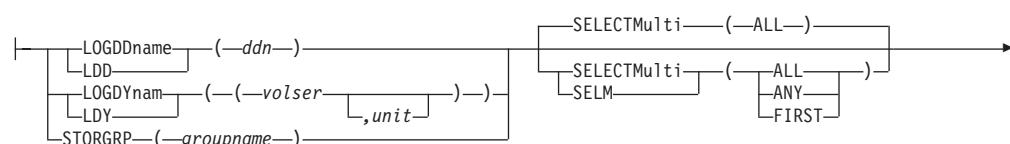
A: Additional Keywords for Physical or Logical Processing:

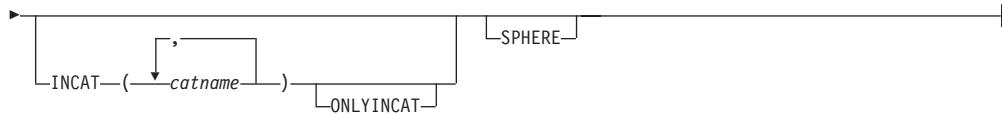


B: Optional Keywords for Physical or Logical Processing:



C: Optional Keywords with RELEASE for Logical Processing:

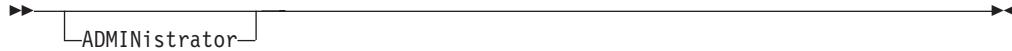




Explanation of RELEASE Command Keywords

This section describes the keywords for the RELEASE command.

ADMINISTRATOR



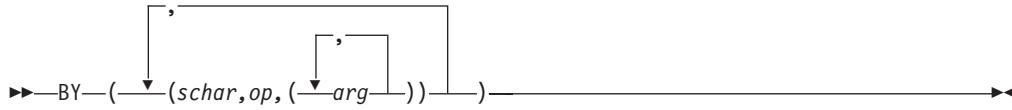
ADMINISTRATOR lets you act as a DFSMSdss-authorized storage administrator for the RELEASE command. If you are not authorized to use the ADMINISTRATOR keyword, the command is ended with an error message. Otherwise, access checking to data sets and catalogs is bypassed.

To use the ADMINISTRATOR keyword all of the following must be true:

- FACILITY class is active.
- Applicable FACILITY-class profile is defined.
- You have READ access to that profile.

Related reading: For additional information about using the ADMINISTRATOR keyword, see “ADMINISTRATOR Keyword” on page 265.

BY



BY specifies that the data sets selected up to this point, by the processing of the INCLUDE and EXCLUDE keywords, are to be further filtered. To select the data set, *all* BY criteria must be met. See the separate discussions of INCLUDE and EXCLUDE for information on how these keywords are specified.

Note: You must use FILTERDD when you have more than 255 entries in INCLUDE, EXCLUDE, or BY list keywords.

Related reading: For additional information about the BY keyword, see “Filtering by Data Set Characteristics” on page 14.

DDNAME



DDNAME requests physical processing.

ddn Specifies the name of the DD statement that identifies a volume whose

RELEASE Command

sequential and partitioned data sets, if selected, are to have their unused space released. To assure correct processing, each of the DD statements corresponding to a DDname (*ddn*) must identify only one volume serial number.

DYNALLOC

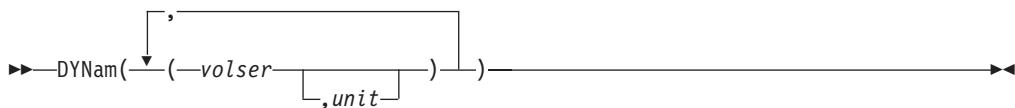


DYNALLOC specifies dynamic allocation, instead of the ENQ macro, to serialize the use of data sets. This allows cross-system serialization in a JES3/MVS environment.

Consider:

- This serialization is of value only when the dynamic allocation/JES3 interface is not disabled.
- Run time increases when you use DYNALLOC to serialize data sets (as opposed to enqueue) because overhead is involved in dynamic allocation and serialization across multiple processors.

DYNAM



DYNAM requests physical processing and specifies that the volume to be processed be dynamically allocated. The volume must be mounted and online. You cannot specify a nonspecific volume serial number using an asterisk (*). Using DYNAM instead of DD statements to allocate DASD volumes will not appreciably increase run time and permits easier coding of JCL and command input.

volser Specifies the volume serial number of a DASD volume to be processed.

unit Specifies the device type of a DASD volume to be processed. This parameter is optional.

EXCLUDE



dsn Specifies the name of a data set to be excluded from the data sets selected by INCLUDE. Either a fully or a partially qualified data set name can be used. See the separate discussions of INCLUDE and BY for information on how these keywords are specified.

Note: You must use FILTERDD when you have more than 255 entries in INCLUDE, EXCLUDE, or BY list keywords.

FILTERDD

```
►─ FILTERDD ─(ddn)─ FDD ─►
```

ddn Specifies the name of the DD statement that identifies the sequential data set or member of a partitioned data set that contains the filtering criteria to use. This is in the form of card-image records, in DFSMSdss command syntax, that contain the INCLUDE, EXCLUDE, and BY keywords that complete the RELEASE command syntax.

Note: You must use FILTERDD when you have more than 255 entries in the INCLUDE, EXCLUDE, or BY list of subkeywords.

FORCECP

```
►─ FORCECP(days) ─►
```

FORCECP specifies that allocated but unused space in checkpointed data sets resident on the SMS volume or volumes can be released. Checkpoint indications are removed from the data set.

days Specifies a one-to-three digit number in the range of zero to 255. It also specifies the number of days that must have elapsed since the last referenced date before the space can be released.

INCAT

```
►─ INCAT( catname ) ONLYINCAT ─►
```

INCAT(*catname*) specifies that DFSMSdss search the user catalogs specified by the INCAT(*catname*) keyword, then follow the standard search order to locate data sets. INCAT(*catname*) allows you to identify specific source catalogs. To specify INCAT, RACF authorization might be required.

catname Specifies a fully qualified catalog name.

ONLYINCAT Specifies that DFSMSdss only searches catalogs that are specified in the INCAT catalog name list.

DFSMSdss does not process an SMS-managed data set that is cataloged outside the standard search order. This is the case even if the data set is cataloged in one of the catalogs that is specified with the INCAT(*catname*) keyword. Ensure that the SMS-managed data sets are cataloged under standard catalog search order.

INCLUDE

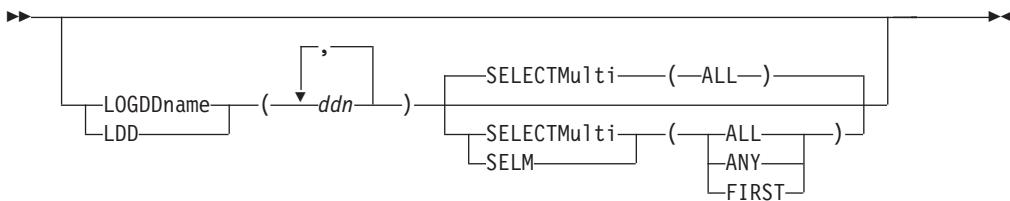
```
►─ INCLUDE( dsn ) ─►
```

RELEASE Command

dsn Specifies the name of a data set whose unused space is eligible to be released. Either a fully or a partially qualified data set name can be used. See “Filtering by Data Set Names” on page 12. If INCLUDE is omitted (but EXCLUDE or BY is specified) or if INCLUDE(**) is specified, *all* data sets are eligible to be selected for releasing. See the separate discussions of EXCLUDE and BY for information on how these keywords are specified.

Note: You must use FILTERDD when you have more than 255 entries in INCLUDE, EXCLUDE, or BY list keywords.

LOGDDNAME

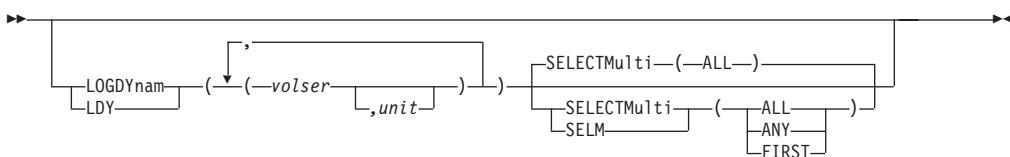


LOGDDname specifies logical processing that is based on a designated volume list.

ddn Specifies the name of the DD statement that identifies a single volume that contains the data sets to be released by logical processing. You can specify up to 255 DDNAME entries with the LOGDDNAME keyword. Each DD statement can correspond to only one volume serial number.

See the **SELECTMULTI** description and the **Notes** under LOGDYNAM.

LOGDYNAM



LOGDYNAM requests logical processing and specifies that the volumes that contain the data sets to be processed are dynamically allocated.

volser Specifies the volume serial number of a DASD volume to be dynamically allocated. The volume must be mounted and online. You cannot specify a nonspecific volume serial number by using an asterisk(*). You can specify up to 511 volumes with the LOGDYNAM keyword.

unit Specifies the device type of a DASD volume to be processed. This parameter is optional.

SELECTMULTI

Specifies how DFSMSdss selects cataloged multivolume data sets. DFSMSdss accepts SELECTMULTI only when you specify logical processing with the LOGDDNAME, LOGDYNAM, or STORGRP keyword. Otherwise, DFSMSdss cannot accept the specification of SELECTMULTI.

ALL Specifies that DFSMSdss *not process* a multivolume data set unless the following criteria is met:

- The volume list created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must list all the volumes that contain a part of the data set.
- The volume list created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must list all the volumes that contain a part of the VSAM cluster.

ALL is the default.

- ANY** Specifies that DFSMSdss process a multivolume data set when the following criteria are met:
- Any volume specified in the volume list that is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must contain a part of the data set.
 - Any volume specified in the volume list that is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword must contain a part of the VSAM cluster.
- FIRST** Specifies that DFSMSdss process a multivolume data set only when the volume list designates the volume that contains the first part of the data set. The volume list is created by the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword. For VSAM data sets, the volume list must include the volume that contains the first extent of the data component for the cluster.

Notes for LOGDDNAME, LOGDYNAM and STORGRP keywords:

1. If the LOGDDNAME, LOGDYNAM, or STORGRP keyword is not specified, DFSMSdss selects from all data sets cataloged in the catalogs accessible through the standard search order.
2. If the LOGDDNAME, LOGDYNAM, or STORGRP keyword is specified, DFSMSdss still uses the standard catalog search order, but it selects data sets only from the specified volumes.
3. A multivolume data set that has extents on volumes is processed when you specify the SELECTMULTI keyword if the following criteria are met:
 - The volumes cannot be designated on the volume list created by the LOGDDNAME, LOGDYNAM, or STORGRP keyword.
 - You must specify the SELECTMULTI keyword option (ANY or FIRST) to release a multivolume data set that has extents on volumes which are not identified with the LOGDDNAME, LOGDYNAM, or STORGRP keyword.

MINSECQTY

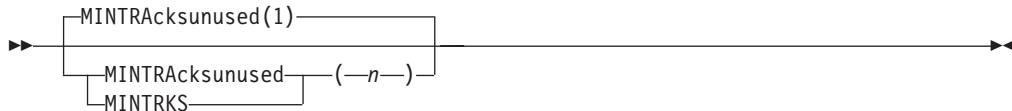


Because DFSMSdss releases all the allocated but unused space when you have not specified a secondary allocation quantity, you will not be able to add records to the data set after the release operation. MINSECQTY solves this problem. The release operation is not performed unless the secondary allocation quantity in the VTOC (VVDS for extended-format VSAM) is equal to or greater than n tracks and the data set has not reached the maximum number of used extents (16 extents for sequential and partitioned data sets, 123 extents for extended-format sequential data sets and PDSEs, or 255 extents for extended-format VSAM data sets). The

RELEASE Command

letter *n* represents a 1-to-8-digit decimal number with a range from zero to 99999999. The default is one track, if you do not specify MINSECQTY (*n*).

MINTRACKSUSED

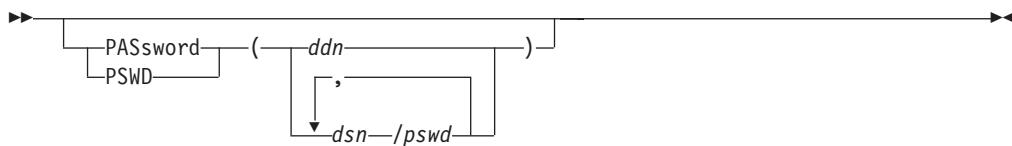


MINTRACKSUSED specifies that the release operation is to be performed only if the number of unused tracks for a selected data set is equal to or greater than *n* (*n* is a 1-to-8-digit decimal number with a range from 0 to 99999999). When you do not specify MINTRKS, DFSMSdss uses a default value of 1. All unused tracks will be released if you perform a release operation.

ONLYINCAT

See "INCAT" on page 73.

PASSWORD



PASSWORD specifies the passwords DFSMSdss uses for selected password-protected sequential and partitioned data sets. (Password checking is bypassed for RACF-protected data sets.) This keyword is required only if:

- You do not have the required volume-level RACF DASDVOL or RACF DATASET access.
- The installation authorization exit does not bypass the checks.

Note: You should specify the passwords for all data sets that do not have RACF protection but do have password protection. During processing, a utility invoked by DFSMSdss may have to prompt the operator for a password. You can control authorization checking by using the installation authorization exit.

ddn Specifies the name of the DD statement that identifies the sequential data set or member of a partitioned data set that contains data set names and their passwords. This data set must contain card-image records in DFSMSdss command syntax format.

dsn/pswd *dsn* is a fully qualified data set name. *pswd* is its password. If no password follows the slash (/), *dsn* is treated as though it were *ddn*.

Printing of actual data set passwords specified in your input command stream is suppressed in the SYSPRINT output.

Related reading: For additional information about the installation authorization exit, see *z/OS DFSMS Installation Exits*.

PROCESS



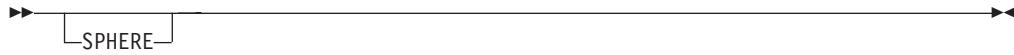
PROCESS specifies that the release operation is to be performed for data sets with a high-level qualifier of SYS1. To specify PROCESS(SYS1), RACF authorization may be required.

Related reading: For additional information about RACF authorization, see the *z/OS DFSMSdss Storage Administration Guide*.

SELECTMULTI

See the LOGDDNAME, LOGDYNAM, or STORGRP keyword.

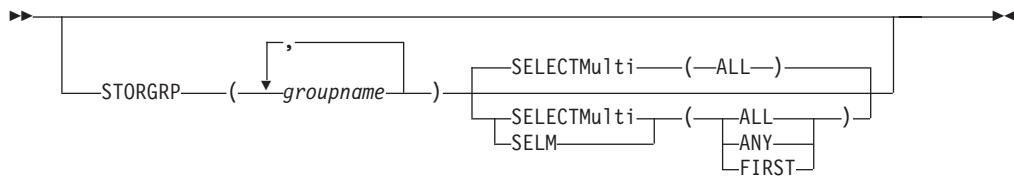
SPHERE



SPHERE specifies that DFSMSdss also select all associated alternate index clusters whenever you select a VSAM base cluster. You do not need to specify individual names of sphere components, only the base cluster name. If you specify a volume list (with LOGDDNAME or LOGDYNAM), you do not need to specify the volumes on which the AIX clusters reside.

To select an entire sphere, specify the base cluster name by using fully or partially qualified data set names. If you specify SPHERE but not the base cluster name, DFSMSdss processes only those data components of the sphere whose names are specified.

STORGRP



STORGRP specifies that all of the online volumes in the storage group be dynamically allocated. If a volume in the storage group is not online, that volume is not used for processing. You can specify up to 255 storage group names. Specifying STORGRP with a storage group name is equivalent to specifying LOGDYNAM or LOGDDNAME with all the online volumes in the storage group included in the list.

You can specify the STORGRP keyword with the SELECTMULTI keyword, but STORGRP cannot be specified at the same time with the DDNAME, DYNAM, LOGDDNAME, or LOGDYNAM keywords.

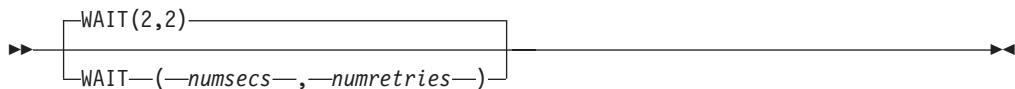
RELEASE Command

Notes for LOGDDNAME, LOGDYNAM, and STORGRP keywords:

1. DFSMSdss selects from all data sets that are cataloged in the catalogs that are accessible through the standard search order if you do not specify the LOGINDDNAME, LOGINDYNAM, or STORGRP keyword.
2. When you specify the LOGDDNAME, LOGDYNAM, or STORGRP keyword, DFSMSdss still uses the standard catalog search order. However, DFSMSdss selects data sets only from the specified volumes.
3. You can process a multivolume data set that has extents on volumes when you specify the SELECTMULTI keyword if the following criteria are present:
 - The volumes cannot be designated on the volume list that is created by the LOGDDNAME, LOGDYNAM, or STORGRP keyword.
 - You must specify the SELECTMULTI keyword option (ANY or FIRST) to release a multivolume data set that has extents on volumes which are not identified with the LOGDDNAME, LOGDYNAM, or STORGRP keyword.

Related reading: For more information about the SELECTMULTI keyword, see “LOGDYNAM” on page 172.

WAIT



WAIT specifies to DFSMSdss the length of a wait in seconds and the number of retries to obtain control of a data set.

numsecs Specifies a decimal number from 0 to 255 that designates the interval, in seconds, between retries.

numretries Specifies a decimal number from 0 to 99 that designates the number of retries to gain control of a data set.

The default for *numsecs*,*numretries* is (2,2), which specifies two retries at 2-second intervals. If you do not want to wait for a resource, specify 0 for either *numsecs* or *numretries*.

Note: The WAIT keyword does not control wait/retry attempts for system resources (such as the VTOC and the VVDS). For system resources, the default wait time is 3 seconds and the default retry count is 30. This results in a total wait time of 90 seconds.

Related reading: For information about controlling the wait/retry attempts for system resources, see the *z/OS DFSMSdss Storage Administration Guide*.

Example of a Release Operation

The following is an example of a release operation on selected sequential and partitioned data sets.

```
//JOB1      JOB  accounting information,REGION=nnnnK
//STEP1      EXEC PGM=ADRDSU
//SYSPRINT DD   SYSOUT=A
//SYSIN      DD   *
  RELEASE INCLUDE(**)  -
    DYNAM(338000)    /* DYNAM ALLOC VOL 338000          */ -
    MINTRKS(10)       /* THERE ARE 10 OR MORE UNUSED TRKS */ -
                      /* MINSEC NOT SPEC. IT DEFAULTS TO 1 */
/*

```

Unused tracks of sequential and partitioned data sets on volume 338000 are to be released if both:

- The number of unused tracks in the data set is greater than or equal to 10.
- The data set can be extended later if required (MINSEC(1)). This need not be specified, because it is the default.

The above example can be modified as follows to release unused tracks of all sequential and partitioned data sets, other than system data sets, that have any unused tracks and that can be extended:

```
//SYSIN      DD   *
  RELEASE INCLUDE(**)  -
    DYNAM(338000)    /* DYNAM ALLOC VOL 338000 */
/*

```

The following is an example of the commands used to release unused space from extended-format VSAM data sets residing wholly or in part on a specific volume, and to release unused space from the data sets' alternate indexes that reside on any volumes:

```
//SYSIN      DD   *
  RELEASE INCLUDE(**)  -
    BY(DSORG EQ VSAM)  /* RELEASE ONLY VSAM      */-
    LOGDYNAM(339000)   /* DYN ALLOC VOL 339000 */-
    SELECTMULTI(ANY)   /* EVEN IF MULTIVOL   */-
    SPHERE             /* RELEASE AIXES     */
/*

```

To release unused space from all eligible data sets that are cataloged in a particular user catalog, without specifying any volume:

```
//SYSIN      DD   *
  RELEASE INCLUDE(**)  -
    INCAT(CATALOGA) ONLYINCAT /* ONLY FROM CATALOG */
/*

```

RESTORE Command

With the RESTORE command, you can restore data to DASD volumes from DFSMSdss-produced dump volumes. You can restore data sets, an entire volume, or ranges of tracks. You can restore to unlike devices from a logical dump tape.

The FULL keyword with the RESTORE command restores an entire DASD volume. The TRACKS keyword for the RESTORE command restores a range of tracks.

RESTORE Command

DFSMSdss offers two ways to process RESTORE commands:

- *Logical processing* is data set-oriented, which means it operates against data sets independently of physical device format.
- *Physical processing* can operate against data sets, volumes, and tracks, but is oriented toward moving data at the track-image level.

The processing method is determined by what type of dump tape is used as input and by the keywords specified on the command.

Target data set allocation differs between a physical data set and logical data set restore of non-VSAM data sets. Logical data set restore allocates target data sets according to the amount of used space in the source data set, thereby freeing unused space. Physical data set restore preserves the original size of the source data set. To force unused space to be kept during logical data set restore, the ALLDATA or ALLEXCP keyword must be specified during a dump. However, the action that restore takes with respect to these keywords depends on data set characteristics and device characteristics. If you require that all of the unused space is restored, then you should be sure that the data set is restored to a like device type and not reblocked or compressed. Compress is the default for PDS data sets on restore unless you use the NOPACKING keyword.

DFSMSdss logical restore processing cannot be used to process partitioned data sets containing location-dependent information that does not reside in note lists or in the directory.

Extended-physical-sequential data sets, VSAM extended-format data sets, and SAM compressed extended function data sets cannot be restored to non-SMS-managed target volumes during a physical or logical data set restore. Data sets with DFM attributes (created by DFM/MVS) can be restored to non-SMS-managed target volumes but the DFM attributes will be lost and a warning message will be issued.

For more information on using the RESTORE command, refer to *z/OS DFSMSdss Storage Administration Guide*.

Special Considerations for RESTORE

The following special considerations apply when you perform a restore operation:

- When restoring a partitioned data set that has a secondary allocation of zero, the amount of unused space allocated to the target data set may be different from that of the source data set if the ALLDATA and ALLEXCP keywords were not specified for the dump.
- When restoring a VSAM data set with no secondary allocation, a secondary allocation may be added as follows:
 - When the primary allocation is less than one cylinder, a secondary allocation equal to the primary is created.
 - When the primary allocation is greater than or equal to one cylinder, a secondary allocation is created by computing one percent of the primary and rounding up to the next cylinder (in tracks).

The index component may also have secondary space added if it has no secondary allocation.

- When performing a logical or physical data set restore operation of a VSAM extended-format data set, the target data set allocation must be consistent with

the source data set allocation as follows:

- If the source is an extended-format VSAM, then the target must be an extended-format VSAM.
- If the source is a compressed VSAM KSDS, then the target must be a compressed VSAM KSDS.
- If the source is an alternate index for an extended-format KSDS, then the target must be an alternate index for an extended-format KSDS.
- The target control interval size must be equal to the source.
- The actions DFSMSdss performs for RENAME, RENAMEUNCONDITIONAL, REPLACE, or REPLACEUNCONDITIONAL depend on the keywords you specify and the configurations of data sets on volumes. This section contains figures that describe specific environments for DFSMSdss restore operations.

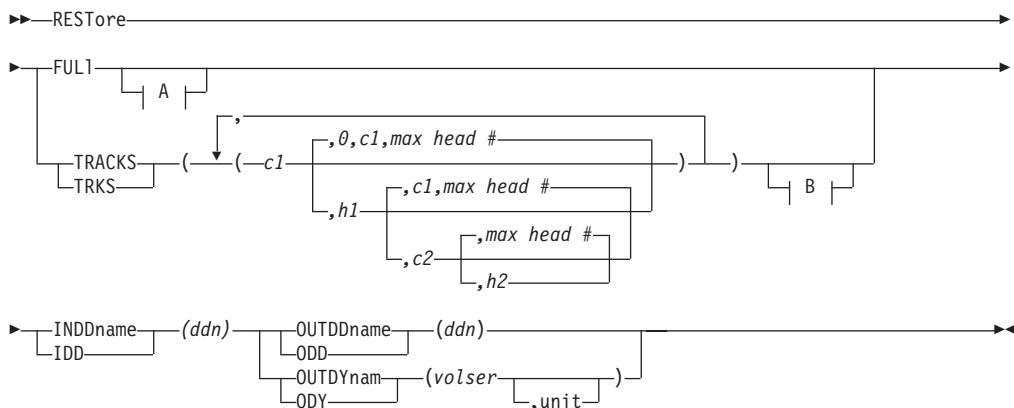
Data Integrity Considerations for Full or Tracks Restore Operations

For a full or tracks restore operation, DFSMSdss serializes the VTOC to preclude DADSM functions such as ALLOCATE, EXTEND, RENAME, and SCRATCH from changing the contents of the VTOC on the volume during the restore operation. Data sets are *not* serialized on these full or tracks operations. Therefore, some data sets might be opened by other jobs during the RESTORE. The result might be that partially updated data sets are restored. Full data integrity can always be guaranteed by performing restore operations by data set only when TOLERATE(ENQFAILURE) or SHARE is not specified.

During full or tracks restore operations, and Stand-Alone restore, it is possible to duplicate a volume serial number in the sysplex. If there is data in the coupling facility for the duplicated volume serial number, data integrity can be compromised. The recommended procedure before restoring a suspected volume is as follows:

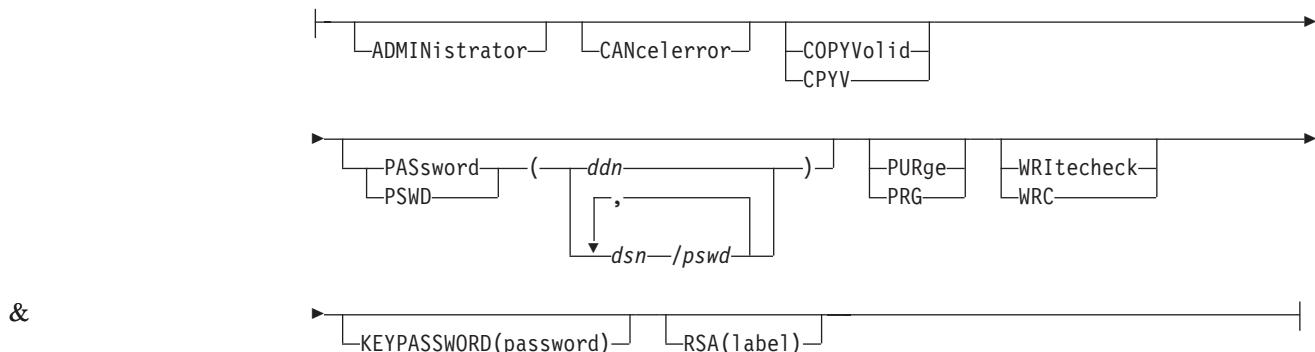
1. Use the D SMS,CFVOL(valid) command to determine if there is any data in the coupling facility caches for the volume serial number to be restored.
2. Determine the disposition of any data that is in the caches.
3. Do not restore the suspected volume until the duplicated volume can be varied offline to the sysplex, thus ensuring that there is no data in any coupling facility cache for the duplicated volume.

RESTORE FULL and RESTORE TRACKS Command Syntax

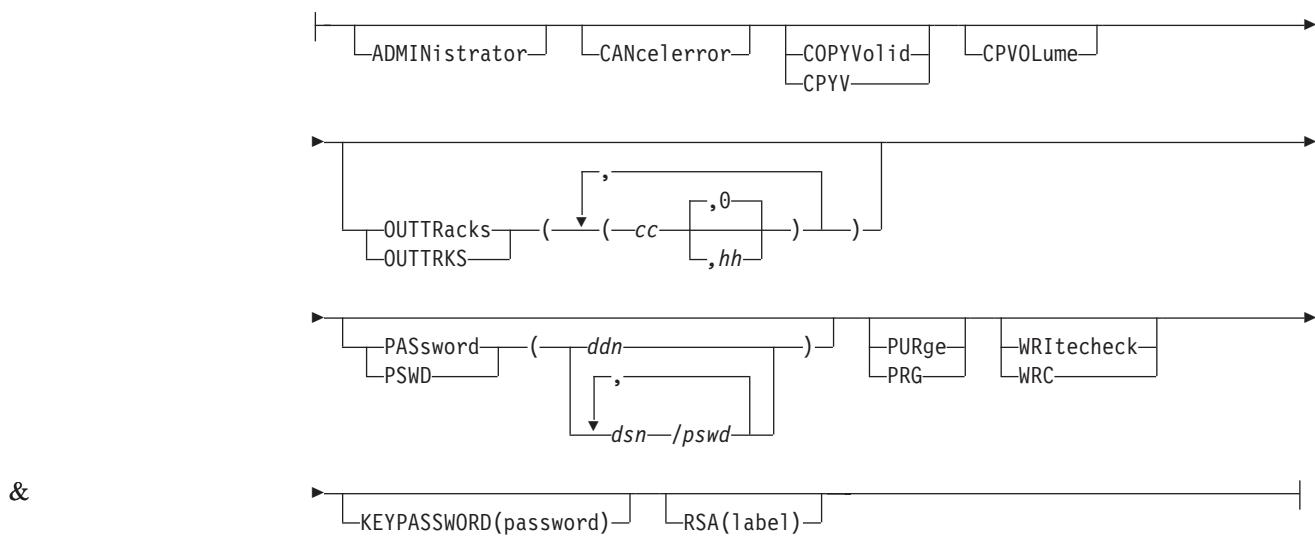


RESTORE Command

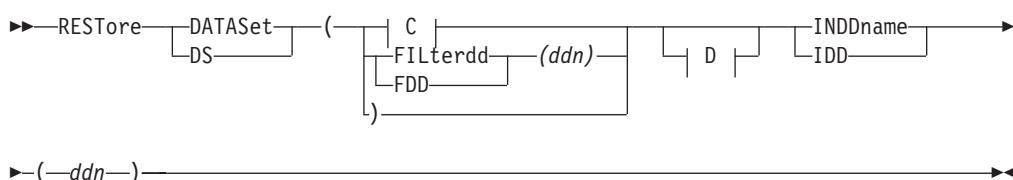
A: RESTORE FULL Command Optional Keywords:



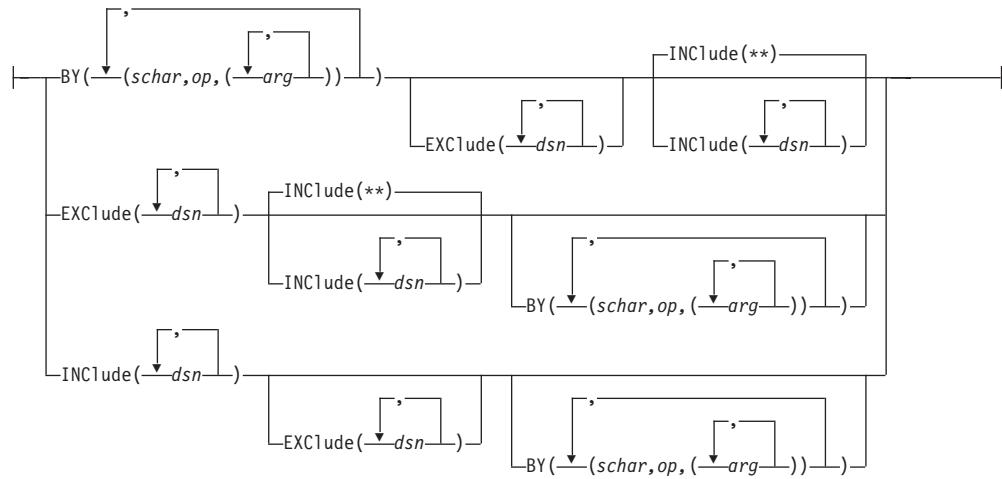
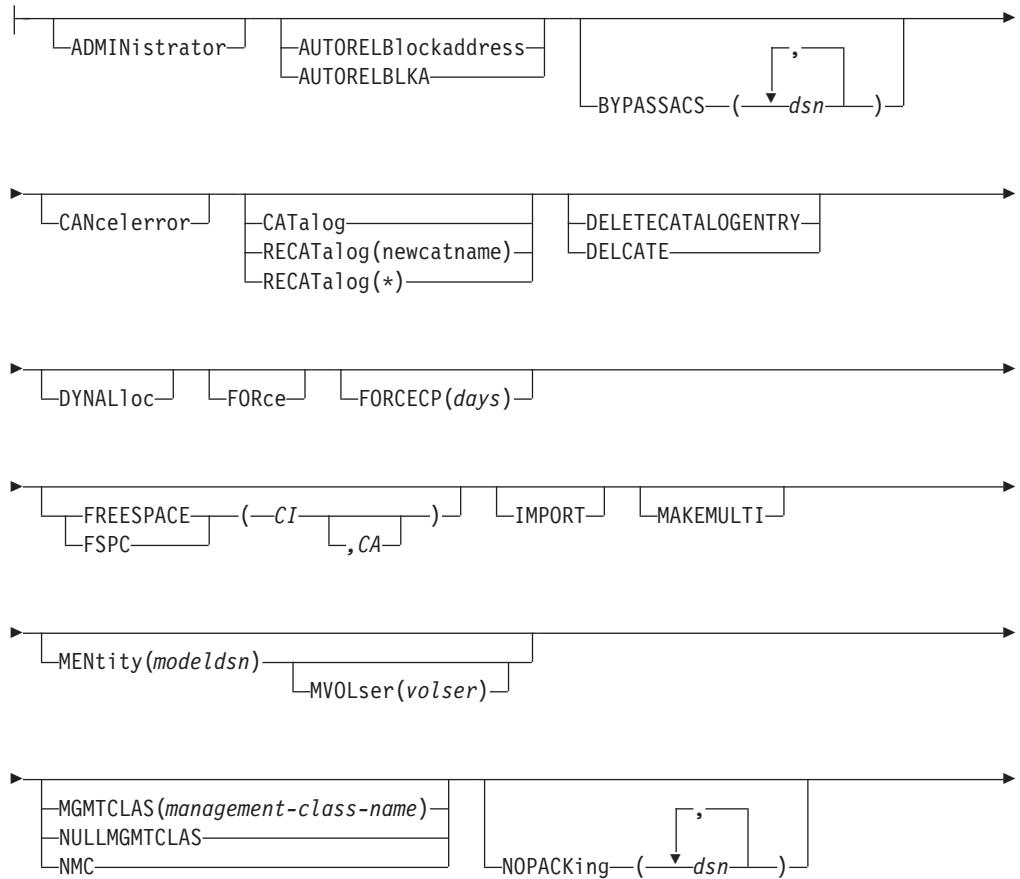
B: RESTORE TRACKS Command Optional Keywords:



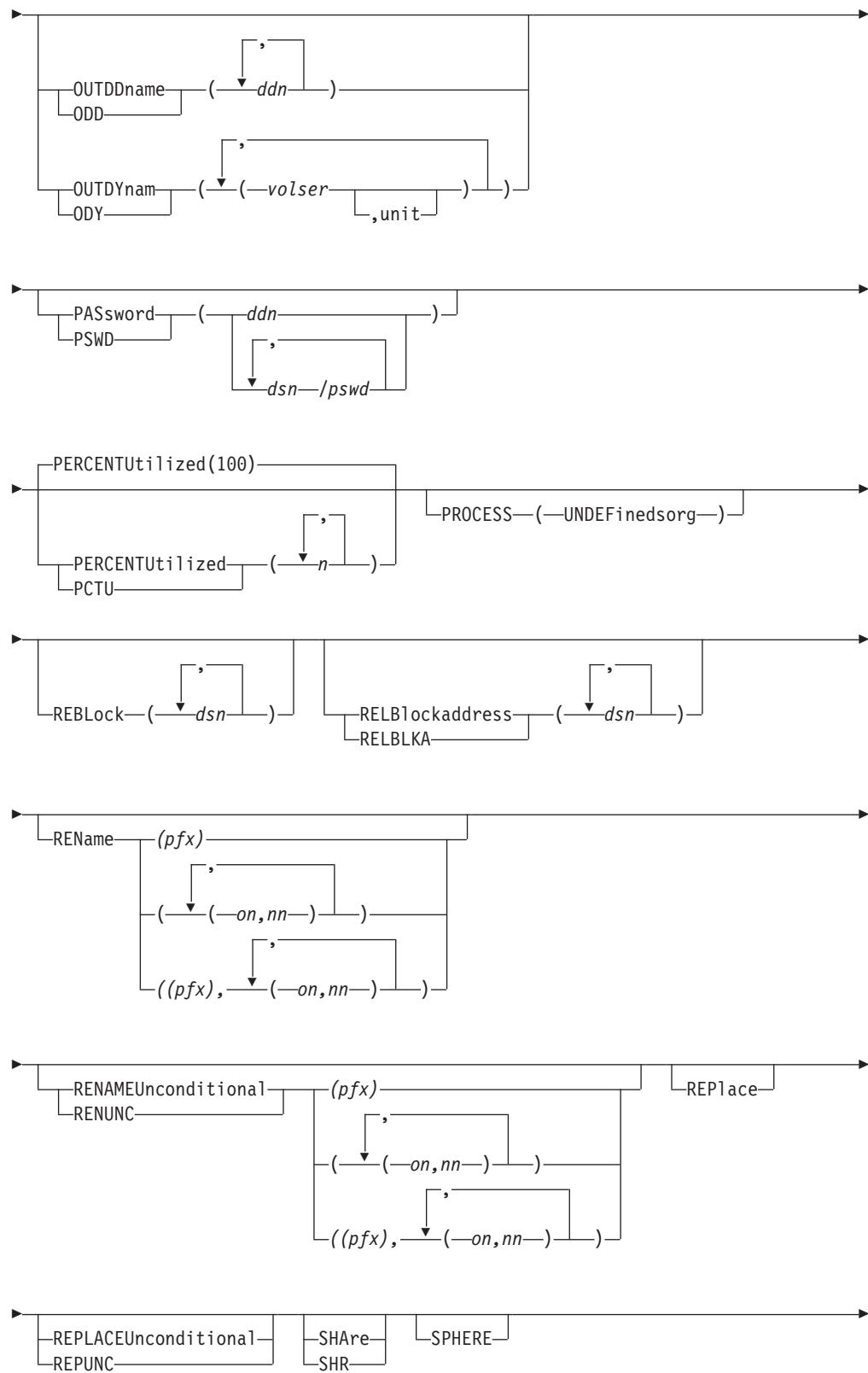
RESTORE DATASET Command Syntax for Logical Data Set



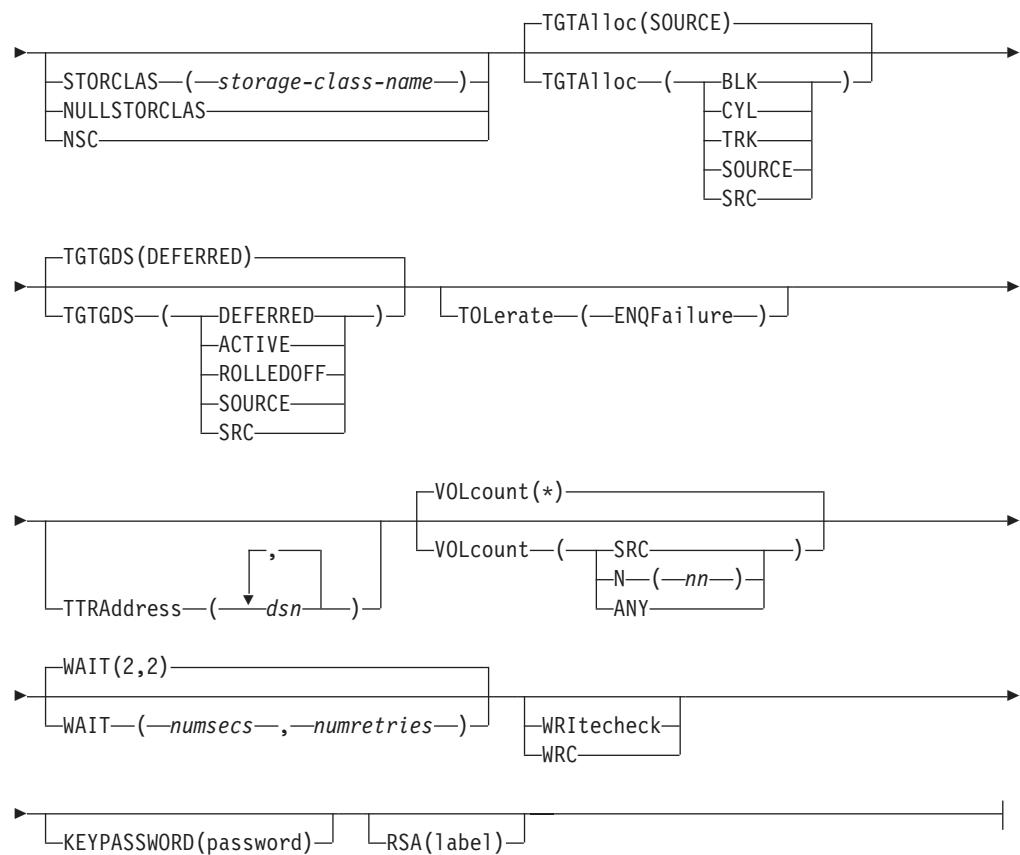
C: Additional Keywords Used for Logical Data Sets:

**D: Optional Keywords Used for Logical Data Sets:**

RESTORE Command

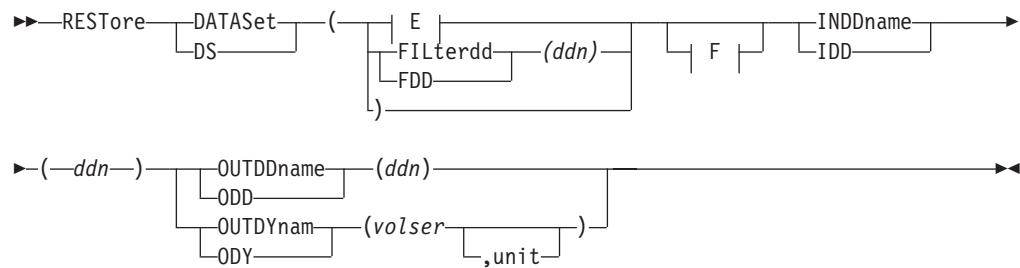


RESTORE Command

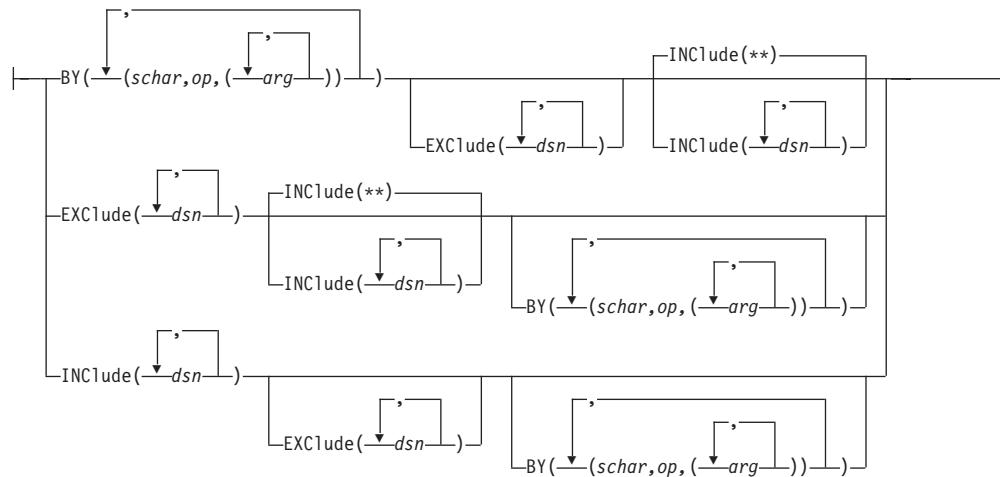


RESTORE Command

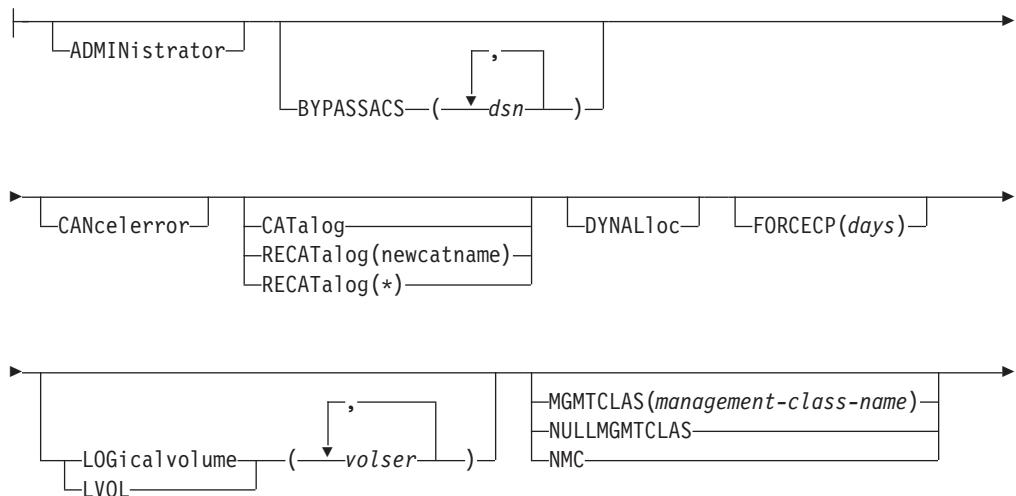
RESTORE DATASET Command Syntax for Physical Data Set



E: Additional Keywords for Physical Data Sets:



F: Optional Keywords Used for Physical Data Sets:





Explanation of RESTORE Command Keywords

This section describes the keywords for the RESTORE command.

ADMINISTRATOR

ADMINistrator

RESTORE Command

ADMINISTRATOR lets you act as a DFSMSdss-authorized storage administrator for the RESTORE command. If you are not authorized to use the ADMINISTRATOR keyword, the command is ended with an error message. Otherwise, access checking to data sets and catalogs is bypassed.

To use the ADMINISTRATOR keyword all of the following must be true:

- FACILITY class is active.
- The applicable FACILITY-class profile is defined.
- You have READ access to that profile.

Requirement: You must also specify the ADMINISTRATOR keyword with CPVOLUME because DFSMSdss cannot check access authorization for VM data.

Related reading: For additional information about the ADMINISTRATOR keyword, see “ADMINISTRATOR Keyword” on page 265.

AUTORELBLOCKADDRESS



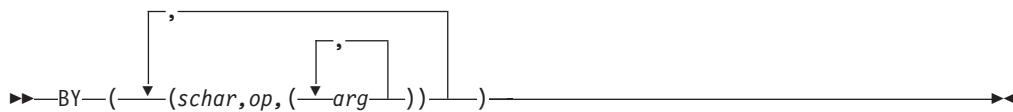
AUTORELBLOCKADDRESS specifies that direct access data sets are to be automatically processed as being organized by relative block address provided that they were accessed with an OPTCD setting indicating relative block addressing. These direct access data sets are to be processed by relative block address rather than by TTR, and without maintaining the relative track and record number for each record. The blocks must fit on the tracks of the target volume.

Notes:

1. If any such data set is actually organized by TTR, the data set may become unusable.
2. The TTRADDRESS keyword takes precedence over the AUTORELBLOCKADDRESS keyword. Refer to the RELBLOCKADDRESS and TTRADDRESS keywords for more information.
3. AUTORELBLOCKADDRESS is ignored for direct access data sets with variable records formats or with standard user labels.

Related reading: For additional information about OPTCD, see *z/OS DFSMS Macro Instructions for Data Sets*.

BY



BY specifies that the data sets selected up to this point, by the processing of the INCLUDE and EXCLUDE keywords, are to be further filtered. To select the data set, *all* BY criteria must be met. Also, separate discussions of INCLUDE and EXCLUDE provide information on how these keywords are specified.

Related reading: For additional information about the BY keyword, see “Filtering by Data Set Characteristics” on page 14.

BYPASSACS



BYPASSACS specifies that Automatic Class Selection (ACS) routines are not to be invoked to determine the target data set's storage class or management class names. To specify BYPASSACS, RACF authorization may be required.

dsn Specifies a fully or partially qualified data set name.

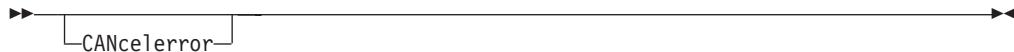
If a data set is being renamed, the old name must be specified.

Note: If BYPASSACS(dsn) is specified, all data sets that pass the BYPASSACS selection criteria are guaranteed the specified storage class. The combination of NULLSTORCLAS and BYPASSACS(dsn) forces the selected data sets to be non-SMS managed.

Related reading:

- For additional information about RACF authorization, see the *z/OS DFSMSdss Storage Administration Guide*.
- For additional information about data set names, see “Filtering by Data Set Names” on page 12.

CANCELERROR



CANCELERROR specifies that the restore function be ended for a permanent read error, or that the restore of a data set be ended for a write error.

- Permanent read error (permanent I/O error):

If CANCELERROR is specified, the restore task is ended. If this keyword is not specified, DFSMSdss attempts to recover from the input errors, but the results can be unpredictable as a result of the difficulty in repositioning and assembling the data.

- Write error, such as an invalid track format:

For data set restore, processing of the data set ends and the target data set is deleted. The restore operation continues with the next data set. For full volume and tracks restore, processing for the volume ends. Subsequent tracks are not processed.

DFSMSdss allows you to change this default operation. A patch byte is provided to allow you to change the default handling of invalid tracks created during RESTORE processing.

CANCELERROR has no effect on the following types of errors on a DASD volume:

- Equipment check
- Command reject

RESTORE Command

- Intervention required
- Busout parity

Related reading: For additional information about the patch byte, see the *z/OS DFSMSdss Storage Administration Guide*.

CATALOG or RECATALOG



For a *logical* restore operation, CATALOG instructs DFSMSdss to catalog data sets that it allocates. For a *physical* restore operation, CATALOG is used for non-VSAM single volume data sets; RECATALOG is ignored. DFSMSdss does not catalog VSAM data sets during physical restore. If the CATALOG keyword is specified, it is ignored when processing VSAM data sets. You must use IDCAMS DEFINE RECATALOG to catalog the data sets after the physical restore.

CATALOG

catalogs the target data set in a catalog as determined by the standard catalog search order. This is the default for VSAM data sets, multivolume data sets, and SMS-managed data sets (during logical data set restore). It is also the default for single-volume, non-VSAM, SMS-managed data sets (in physical data set restore).

RECATALOG(*newcatname*)

catalogs the target data set in the *newcatname* catalog.

RECATALOG(*)

catalogs the target data set in the same catalog that points to the source data set. If the source data set was not cataloged, the new data set is not cataloged either.

Notes:

1. CATALOG or RECATALOG fails if the target data set is already cataloged in the same catalog and RENAME is not specified.
2. CATALOG and RECATALOG are ignored when the target data set is preallocated.
3. RECATALOG is ignored for SMS-managed data sets.
4. Be careful when using RECATALOG(*newcatname*) because the target data set may already be cataloged outside of the standard order of search.
5. If RECATALOG is specified for a physical restore it is ignored. DFSMSdss does not attempt to catalog the data set. If the data set is already cataloged on another volume:
 - The catalog entry is not updated, and
 - The data set is restored, but it is not cataloged.
6. If CATALOG is specified for a physical restore of a single-volume non-VSAM data set, DFSMSdss attempts to catalog the data set. If the data set is already cataloged on another volume:
 - Message ADR385E, reason code 08, is issued,
 - The catalog entry will not be updated, and
 - The data set will be restored, but it will not be cataloged.

7. DFSMSDss physical data set restore does not create catalog entries for VSAM, multivolume non-VSAM, or preallocated data sets. The user must create all catalog entries.
 8. For multivolume non-VSAM data sets, use IDCAMS DEFINE NONVSAM.

COPYVOLID



COPYVOLID specifies that the volume ID from the dumped DASD volume is to be copied to the output DASD volume. This applies to a full restore operation; it applies to a tracks restore only if track 0 (zero) is to be restored.

When the volume serial number on a DASD volume is changed, the operator is notified. The operating system then initiates either:

- A demount if there is another volume with the same serial number
 - A mount to get the volume with the new serial number mounted

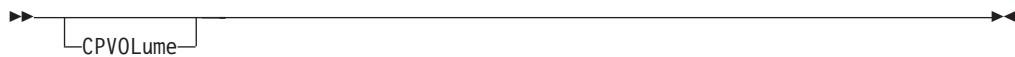
This might change the mount attributes of the volume. You should exercise operating precautions if there are two or more processors sharing the same DASD volume.

When the volume serial number is changed by using a COPYVOLID keyword or when both the dumped volume and the restored volume have different serial numbers, profiles are not built for the RACF-protected data sets on the restored volume or for the RACF DASDVOL for RACF-protected DASD volumes.

Notes:

1. If you are doing a full restore operation and the input has VSAM data sets, the VOLID must be copied.
 2. The COPYVOLID keyword is required for a full-volume restore operation of SMS-managed source volumes.
 3. Exercise caution using the COPYVOLID keyword in a multiple task job step when two or more of the tasks are using the same output volume. If the output volume is made unavailable by the first task, all succeeding tasks that use the same output volume will fail.

CPVOLUME



CPVOLUME specifies that the output volume is a VM-format volume and that the OS-compatible VTOCs must begin on track zero, record five. You must specify the track range to be copied with the **TRACKS** keyword, as the OS-compatible VTOCs do not describe the extents of any data on the volume. You must also specify the **ADMINISTRATOR** keyword with **CPVOLUME** because DFSMSdss cannot check access authorization for VM data.

RESTORE Command

DATASET



DATASET specifies a data set restore operation, using filtering.

Note: Either the FILTERDD, INCLUDE, EXCLUDE, or BY keyword must be used when DATASET is selected.

Related reading: For additional information about filtering, see Chapter 2, “Filtering—Choosing the Data Sets You Want Processed,” on page 11.

DELETECATALOGENTRY



DELETECATALOGENTRY specifies that the user wants to perform a disaster recovery and that the existing catalog entries for the target data sets may no longer be valid. If a target data set is cataloged but not available, then DFSMSdss performs a DELETE NOSCRATCH operation for the data set.

Specifying the DELETECATALOGENTRY command requires proper RACF facility class authorization.

Related reading:

- For additional information about RACF, see the *z/OS DFSMSdss Storage Administration Guide*.
- For additional information about protecting keywords with RACF, see *z/OS Security Server RACF Security Administrator's Guide*.

Caution: Use extreme care with the DELETECATALOGENTRY keyword. If any of the following conditions exist, a DELETE NOSCRATCH operation is performed:

- Condition 1** The target data set is cataloged but it is not found on the volume indicated by the catalog.
- Condition 2** The target data set is cataloged but the volume indicated by the catalog does not exist on the restoring system and the restoring system is not sharing catalogs with another system.
- Condition 3** The target data set is cataloged but the volume indicated by the catalog is offline on the restoring system and the restoring system is not sharing catalogs with another system.
- Condition 4** The target data set is cataloged but the volume indicated by the catalog does not exist on the restoring system and the restoring system is sharing catalogs with another system.
- Condition 5** The target data set is cataloged but the volume indicated by the catalog is offline on the restoring system and the restoring system is sharing catalogs with another system.
- Condition 6** The target data set is a multivolume data set and some, but not all of the data, is no longer available. For this purpose, a multivolume

data set is one in which any part of the data set resides on more than one volume. Examples include the following:

- A VSAM KSDS with the data- and index-components on one volume and an alternate index (AIX) on another volume.
- A VSAM KSDS with the index-component on a different volume than the data-component.
- A key-range VSAM data set with the key-ranges residing on more than one volume.

For conditions 1 and 2, it is appropriate to use `DELETECATALOGENTRY` to have DFSMSdss perform a `DELETE NOSCRATCH` operation for the catalog entries.

Condition 1 typically occurs during a disaster recovery when DFSMSdss restores or imports the catalogs before recovering the data sets. The target data set does not exist on the volumes indicated by the catalog. The restore will not be successful unless you or DFSMSdss delete the data set entries in the catalogs.

Condition 2 typically occurs during a disaster recovery when DFSMSdss restores the data sets to different volumes on a different operating system. DFSMSdss catalogs the target data sets, but points to volumes that do not exist on the target system. The restore will not be successful unless you or DFSMSdss delete the data set entries in the catalogs.

If you specify `DELETECATALOGENTRY` for conditions 3 through 6, the following damage is most likely to occur:

- Condition 3 typically occurs during a disaster recovery when DFSMSdss restores the data sets to different volumes on a different operating system. DFSMSdss cataloged the target data sets, but points to offline volumes on the target system. The restore will not be successful unless you or DFSMSdss delete the data set entries in the catalogs.
- Condition 3 can also occur in any environment where volumes are varied on and offline.

If you specify `DELETECATALOGENTRY` and then vary the volume back online, you may find that there are *two* copies of a data set: the original data set on the volume that was varied offline and the restored data set. DFSMSdss will no longer catalog the original data set.

- Conditions 4 and 5 typically occur in a shared system environment with a nonsymmetric system configuration. The following examples apply:
 - There are shared catalogs between two systems.
 - A data set is dumped from system A, but restored into system B.
 - The data set is cataloged in system A in a catalog that is shared by system B.
 - The volume containing the data set and, if applicable, its VVDS is on system A and is unavailable to system B.

Restriction: If you specify `DELETECATALOGENTRY` and then vary the volume back online, you might find that there are *two* copies of a data set: the original data set on the volume that was varied offline and the restored data set. If this happens, DFSMSdss can no longer catalog the original data set.

- DFSMSdss attempts to detect Condition 6 and issues an error message instead of performing the `DELETE NOSCRATCH`. However, it will not always be possible to do so. In that event, the `DELETE NOSCRATCH` is performed and DFSMSdss attempts to restore the data set with one of the following results:

RESTORE Command

- DFSMSdss successfully restores the data set, and there is no residual data from the original data set. In this case, there is nothing further for you to do.
- DFSMSdss successfully restores the data set, but there is residual data from the original data set. Although DFSMSdss restored the data set, you may find, for example, that there is an uncataloged AIX from the original data set. In this case, you will have to perform other corrective actions. These are the same actions that you would have had to perform even if you had not used the DELETECATALOGENTRY keyword.
- An error occurs during the restore that prevents the data set from being restored. The error results from the fact that the data set was only partially missing. For example, residual data and the VSAM Volume Record (VVR) may exist on a second volume. When the data set that is being restored is extended to that volume, a duplicate entry condition is created. Again, if this occurs, you will have to perform other corrective actions before you can successfully restore the data set.

Notes:

1. DELETECATALOGENTRY is only supported for logical data set restore.
2. DELETECATALOGENTRY should not be used to restore partially damaged volumes or data sets.
3. DELETECATALOGENTRY will not delete any catalog entry for user catalogs.
4. DELETECATALOGENTRY will not delete any catalog entry for migrated data sets (VOLSER=MIGRAT).
5. DELETECATALOGENTRY will not delete any catalog entry for the SYSRES volume (VOLSER=*****).
6. DELETECATALOGENTRY will not delete any catalog entry for a data set where the volser(s) in the catalog do not match either the source volser(s) from the dump tape or the target volser(s) specified with OUTDD/OUTDYNAM.
7. If you are performing a disaster recovery and the original volumes are not available on the restoring system, you should also specify the IMPORT parameter. If you do not specify the IMPORT keyword, the system may prompt you to mount the volumes on which the target data resides (as indicated by the catalog). Even if you reply 'CANCEL', DFSMSdss will attempt to perform a DELETE NOSCRATCH on the data set because it is unable to recognize the 'CANCEL' request. Even if the DELETE NOSCRATCH is successful, DFSMSdss might cause the following results:
 - DFSMSdss might fail to allocate the volume
 - DFSMSdss might issue the message ADR405E
 - DFSMSdss might not restore the data set
8. DELETECATALOGENTRY will not delete any catalog entry when the phantom entry indicates a different type of data set than the source data set.

DYNALLOC



DYNALLOC specifies that DFSMSdss is to use dynamic allocation, instead of enqueue, to serialize the use of data sets. This allows cross-system serialization in a JES3/MVS environment. The following conditions apply to DYNALLOC:

- The serialization is of value only when the dynamic allocation/JES3 interface is *not* disabled.

- Run time increases when you use DYNALLOC to serialize data sets (as opposed to enqueue). This is because overhead is involved in a dynamic allocation and serialization across multiple processors.

EXCLUDE



dsn Specifies the name of a data set to be excluded from the data sets selected by INCLUDE. Either a fully or a partially qualified data set name can be used. See the separate discussion of INCLUDE and BY for information on how these keywords are specified.

Guideline: When data sets have location-dependent data, list their names in the EXCLUDE list. This prevents DFSMSdss from restoring them even if you use the FORCE keyword.

FILTERDD



ddn Specifies the name of the DD statement that identifies the sequential data set or member of a partitioned data set containing the filtering criteria to use. This is in the form of card-image records in DFSMSdss command syntax. It contains the keywords INCLUDE, EXCLUDE, and BY, completing the RESTORE command syntax.

Note: You must use FILTERDD when you have more than 255 entries in the INCLUDE, EXCLUDE, or BY list of subkeywords.

FORCE



For a logical data set restore operation, FORCE specifies that DFSMSdss allows an unmovable data set or a data set allocated by absolute track allocation to be moved.

Guidelines:

- Use the EXCLUDE keyword with the FORCE keyword when you are restoring data sets that have location-dependent data. Naming these data sets in the EXCLUDE list prevents DFSMSdss from restoring them even when you specify the FORCE keyword.
- Specify the FORCE keyword with the REPLACE or REPLACEUNCONDITIONAL keyword if you want to restore the data from unmovable data sets whose extents do not match.

RESTORE Command

FORCECP

```
>>> [FORCECP(days)] <<<
```

FORCECP specifies that any checkpointed data sets resident on the SMS volume or volumes can be logically restored or that MVS checkpointed data sets can be physically restored. Checkpoint indications are removed from the data sets.

days Specifies a one-to-three digit number in the range of zero to 255. It also specifies the number of days that must have elapsed since the last referenced date before the data set can be restored.

Note: For IMS GSAM checkpointed data sets that are physically restored, FORCECP is not required, and checkpoint indications are not removed from the data sets regardless of whether or not FORCECP is specified.

FREESPACE

```
>>> [FREESPACE(FSPC) (CI [,CA])] <<<
```

FREESPACE specifies free space values for DFSMSdss-allocated target VSAM data sets. If this keyword is omitted, the control interval and control area free space are the same as the source data set.

CI Specifies the percentage of freespace to be kept in each control interval during allocation of the data set.

CA Specifies the percentage of freespace to be kept in each control area during allocation of the data set. When omitted, the control area free space is the same as the source data set.

FULL

```
>>> FUL1 <<<
```

FULL specifies that an entire DASD volume is to be restored. This is the default for the RESTORE command.

Notes:

1. You cannot specify SHARE or TOL(ENQF) for FULL operations.
2. It is possible to duplicate a volume serial number during a restore FULL operation.

Related reading: For additional information about how to maintain data integrity when restoring a volume with FULL restore, see “Data Integrity Considerations for Full or Tracks Restore Operations” on page 179.

IMPORT

```
>>> [IMPORT] <<<
```

IMPORT specifies that the data sets that are being restored were dumped from a different system and they should be considered new data sets.

Because the restored data sets are new to the system, DFSMSdss modifies certain source data set processing. The following examples apply:

- DFSMSdss bypasses checking to see if you are authorized to read a data set with the same name as the one that was dumped.

If you are authorized to read the input dump data set that contains the data sets that are being restored, DFSMSdss considers that you have the authority to read any data set that is being restored. DFSMSdss continues to check to see that you are authorized to create a new target data set or replace an existing target data set.

If you are restoring the data set without renaming it, the restore may be unsuccessful. There are several common reasons for such a failure:

- You do not have sufficient authority to create a target data set with the same name as the source data set. You must obtain the required access authority to do so before you can restore the data set.
- A data set already exists with the same name as the source data set and you did not specify the REPLACE or REPLACEUNCONDITIONAL keyword. You must also specify the REPLACE or REPLACEUNCONDITIONAL keyword if you want the restore to replace an existing data set.
- A data set with the same name as the source data set is already cataloged, but is not available to the restoring system. You must make the volumes that contain the data set, and, as applicable, its VVDS, available to the restoring system before you can restore the data set.
- A catalog contains a phantom entry for a data set with the same name as the source data set. In this case, the data set does not exist on any volume. You may perform a separate DELETE NOSCRATCH operation for that data set name. Or you can specify the DELETECATALOGENTRY parameter to request that DFSMSdss perform a DELETE NOSCRATCH operation for you.

Attention: Do not use the DELETECATALOGENTRY keyword if the restoring system is sharing catalogs, but not the data set volumes, with another system.

- DFSMSdss can try to access a VVDS for a source data set in order to obtain information such as the resource owner. If you specify IMPORT, DFSMSdss suppresses a VVDS not available error.

To specify IMPORT, you must have the proper RACF facility class authorization.

Notes:

1. IMPORT should be RACF-protected.
2. IMPORT is only supported for logical data set restore.
3. DELETECATALOGENTRY may have to be specified to successfully restore a data set with the old data set name.
4. When IMPORT is specified, DFSMSdss does not create a discrete data set profile unless the source data set is RACF-protected when it is dumped and you specify the MENTITY keyword.

Related reading:

- For additional information about RACF authorization, see the *z/OS Security Server RACF Security Administrator's Guide*.
- For additional information about using the IMPORT keyword, see "Protecting Usage of DFSMSdss" on page 257.

RESTORE Command

- For additional information about using the DELETECATALOGENTRY keyword to successfully restore a data set with the old data set name, see the *z/OS DFSMSdss Storage Administration Guide*.

INCLUDE

```
INCLUDE( dsn )
```

dsn Specifies the name of a data set eligible to be restored. Either a fully or a partially qualified data set name can be used. See “Filtering by Data Set Names” on page 12. If INCLUDE is omitted (but EXCLUDE or BY is specified) or if INCLUDE(**) is specified, *all* data sets are eligible to be selected for restoring. See the separate discussions of EXCLUDE and BY for information about how these keywords are specified.

INDDNAME

```
INDDname(ddn)
```

ddn Specifies the name of the DD statement that identifies the (input) dump data set. This data set can be on a tape or DASD volume.

Note: Concatenating multiple DFSMSdss dump data sets for the input for RESTORE is not supported and the results are unpredictable. Data sets on any dump data set after the first one in the concatenation will not be restored.

&

KEYPASSWORD

```
KEYPASSWORD(password)
```

&

KEYPASSWORD

&
&
&
&
&
&

The KEYPASSWORD keyword allows you to specify an 8 to 32 EBCDIC character password that is used to generate a clear TDES triple-length key or a clear 128-bit AES key. When using the RESTORE command, you must specify the same KEYPASSWORD password that was specified on the DUMP command.

&
&
&

DFSMSdss removed leading and trailing blanks. Acceptable characters are upper and lower-case letters A through Z, numerals 0-9, and the following characters: !@#\$%&*_-=:<>? | {}

&
&

Imbedded spaces, commas (,), backslash (/), parentheses (()), and semi-colons are unacceptable characters.

&
&
&
&

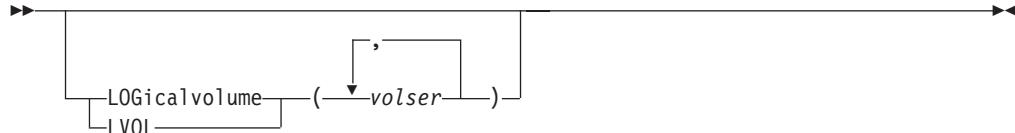
You can specify this keyword on systems that do not have secure cryptographic hardware (for example, z890 or z990 processors that only use CPACF) or on systems that use the clear key options for encryption/decryption (for example, System z9 109 with CPACF).

&

Note:

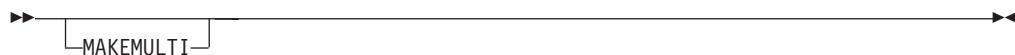
- &
 &
 &
1. If the KEYPASSWORD keyword was specified on the DUMP command, then you must also specify it on the RESTORE command when that dump is being restored.

LOGICALVOLUME



LOGICALVOLUME specifies, for a *physical* data set restore operation, the volume serial numbers of the source DASD volumes that are to be processed. For example, if you have taken a data set dump from volumes 111111, 222222, and so forth, but you want to restore only some data sets from source volume 222222, specify LOGICALVOLUME (222222). LOGICALVOLUME is useful for restoring multivolume data sets.

MAKEMULTI



MAKEMULTI allows DFSMSdss to convert single volume data sets into multivolume data sets. The default is not to convert single volume data sets into multivolume data sets.

This keyword applies only to SMS-managed target data sets. Only single volume, non-VSAM data sets are eligible to be changed into multivolume data sets.

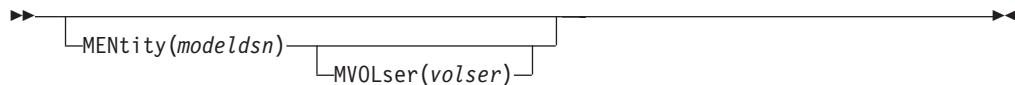
SMS-managed target data sets are given a volume count (VOLCOUNT) that is either:

- The number of SMS output volumes specified in the RESTORE command, if output volumes are specified through OUTDDNAME or OUTDYNAM.
- The number of volumes in the target storage group or 59, whichever is less.

A data set's volume count is the maximum number of volumes to which the data set may extend. At any one time, there may be a mixture of primary volumes (volumes on which space is allocated for the data set) and candidate volumes (volumes on which space may be allocated at a future time). The total sum of the primary volumes and candidate volumes is the data set's volume count.

Note: When MAKEMULTI is specified and VOLCOUNT is also specified with an option other than VOLCOUNT(*), the VOLCOUNT option overrides MAKEMULTI.

MENTITY

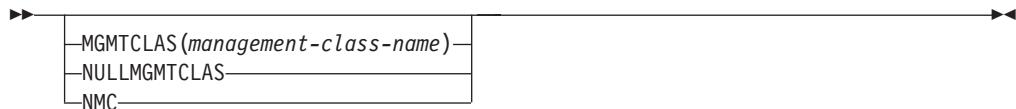


RESTORE Command

MENTITY specifies a model entity and, optionally, the serial number of the volume containing that entity (*volser*) to be used when DFSMSdss defines discrete profiles. These keywords are used to define the data sets to RACF. Specification of MVOLSER is optional when the model entity (MENTITY) is either (1) cataloged in an integrated catalog facility catalog or (2) a non-VSAM data set cataloged in the standard catalog search order. When MVOLSER is specified for a VSAM model entity, the *volser* specified must be the volume serial number of the catalog in which the model entity is cataloged. If these keywords are not specified, DFSMSdss defines the data set to RACF without using a model.

Restriction: You cannot specify the MVOLSER(*volser*) keyword by itself. It can only be specified with the MENTITY(*modeldsn*) keyword.

MGMTCLAS



MGMTCLAS specifies the user-desired management class that is to replace the source management class as input to the ACS routines. You must have the proper RACF authority for the management class specified. The keyword itself does not require RACF authorization.

NULLMGMTCLAS/NMC specifies that the input to the ACS routines is to be a null management class rather than the source data set's management class.

MGMTCLAS and NULLMGMTCLAS are mutually exclusive.

Note: All SMS-managed data sets specified in the BYPASSACS keyword will be assigned the specified management class because the ACS routines will not be invoked. Non-SMS-managed data sets do not have a management class.

NOPACKING



NOPACKING specifies that DFSMSdss is to allocate the target partitioned data set only to devices that are the same or like device type as the source, and that DFSMSdss is to use track level I/O to perform data movement. This results in an exact track-for-track image of the source data set on the target volume.

dsn Specifies the fully or partially qualified names of a PDS to be processed.

NOPACKing is only valid with a PDS. If specified, REBLOCK is ignored for the data set.

A PDS restored using NOPACKing is not compressed during data movement. NOPACKing can be used for a damaged PDS that is currently usable by an application but would be made unusable by compression or other rearrangement of the physical layout of the data.

Note: NOPACKing only applies to logical restore operations. Physical restore uses only track-level I/O. Therefore, no compression will take place against the PDS.

NULLMGMTCLAS

See “MGMTCLAS” on page 78.

NULLSTORCLAS

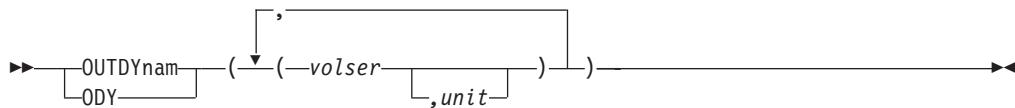
See “STORCLAS” on page 88.

OUTDDNAME



ddn Specifies the name of the DD statement that identifies a volume to be restored to. To assure correct processing, each of the DD statements corresponding to a DDNAME (*ddn*) must identify only one volser. For a logical data set restore or when you specify spill files, you can specify multiple names, separated by commas. For any other type of restore, you can specify only one name.

See the **Notes** under OUTDYNAM for additional information.

OUTDYNAM

OUTDYNAM specifies that the volume to be restored to is to be dynamically allocated. The volume must be mounted and online. You cannot specify a nonspecific volume serial number using an asterisk (*). For a logical data set restore or when you specify spill files, one or more volumes are allowed. For any other type of restore, only one volume is allowed.

volser Specifies the volume serial number of a DASD volume to be restored.

unit Specifies the device type of a DASD volume to be restored. This parameter is optional.

Notes for OUTDDNAME and OUTDYNAM Keywords

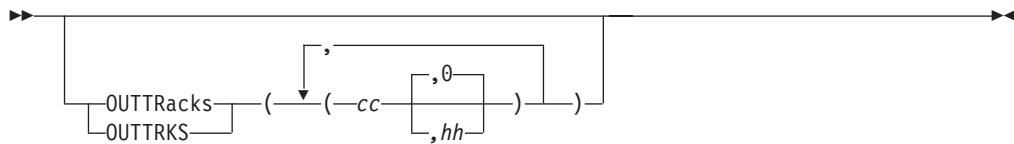
- OUTDDNAME or OUTDYNAM is required for a physical restore, even for SMS-managed data. They are optional for a logical restore operation, except:
 - For multivolume data sets that are preallocated on volumes that are different from the original source volumes; or
 - When the original source volume is not available, and the restored data set is not going to be SMS-managed.
- DFSMSdss now distinguishes between non-SMS and SMS volumes specified in the OUTDDNAME or OUTDYNAM keywords. For non-SMS allocations, only the volumes that are non-SMS are considered for allocation. Similarly, only SMS volumes are considered for SMS allocations.

This distinction is also used when determining the volume count for a multivolume allocation. Where volume count is determined from the number of specified volumes, only those volumes eligible for the type of allocation being done are counted. If there are no volumes that match the type of allocation (SMS volumes for SMS allocation, or non-SMS volumes for non-SMS allocation), processing proceeds with a null volume list.

- If a non-VSAM data set is migrated by DFSMShsm after a logical dump operation is performed on the data set, it should be recalled before a restore is attempted for the data set. If it is not recalled and if either OUTDDNAME or OUTDYNAM is specified indicating an output volume with REPLACE or REPLACEUNCONDITIONAL and RECATALOG(*), DFSMSdss issues a message indicating that an error occurred while trying to catalog the restored data set because it was already cataloged as a migrated data set. The data set will be restored on the volume specified by the OUTDDNAME or the OUTDYNAM, but the data set will not be cataloged. If neither OUTDDNAME nor OUTDYNAM is specified, DFSMSdss issues a message indicating that data sets with a volume serial of MIGRAT cannot be restored.

If the DD statement corresponding to the *ddn* of the OUTDDNAME contains the data set name and disposition but not the *volser*, DFSMShsm recalls the data set automatically and DFSMSdss restores the data set.

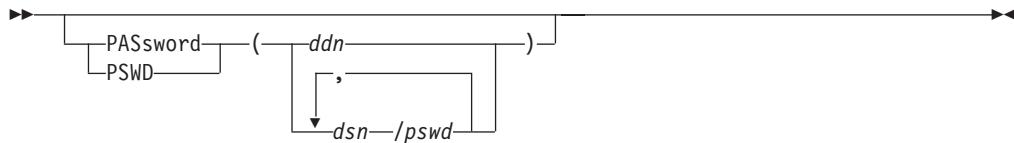
OUTTRACKS



OUTTRACKS specifies, for a track restore operation, the beginning location of the cylinder (*cc*) and head (*hh*) number of the target volume to which the track is to be restored. The number of (*cc, hh*) combinations specified in the OUTTRACKS keyword must be the same as the number of (*c1,h1,c2,h2*) combinations specified in the TRACKS keyword.

If OUTTRKS is not specified, the track is restored to its original cylinder and head number.

PASSWORD



PASSWORD specifies the passwords DFSMSdss is to use for password-protected data sets for all restore operations. (Password checking is bypassed for RACF-protected data sets.) The PASSWORD keyword is required only if:

- You do not have the required volume-level RACF DASDVOL or RACF DATASET access.
- The installation authorization exit does not bypass the checks.
- You do not want to be prompted for the password for VSAM data sets.

Note: Specify the passwords for all data sets that do not have RACF protection but do have password protection. During processing, a utility invoked by DFSMSdss may prompt the operator for a password. You can control authorization checking by using the installation authorization exit.

Catalog passwords are not supported to facilitate disaster recovery operations, application data transfers, and data set migration. Catalog protection via an access control facility, such as RACF, is the preferred method of protection.

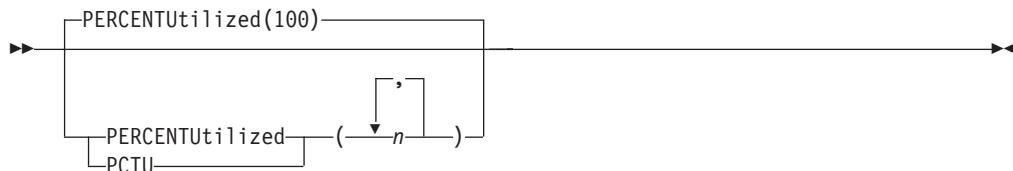
ddn Specifies the name of the DD statement that identifies the sequential data set, or member of a partitioned data set, that contains data set names and their passwords in the format *dsn/pswd[...]*. This data set must contain card-image records in DFSMSdss command syntax format.

dsn/pswd *dsn* is a fully or partially qualified data set name. *pswd* is its password. If no password follows the slash (/), *dsn* is treated as though it were *ddn*.

Note: Printing of actual data set passwords specified in your input command stream is suppressed in the SYSPRINT output.

RESTORE Command

PERCENTUTILIZED



PERCENTUTILIZED specifies that DFSMSdss must stop allocating data sets to the target volumes when the allocated space reaches *n* percent of the total space on the target volume. The default is 100. Specify more than one *n* if you have more than one target volume (for instance, a volume for overflow). If there are more target volumes than you have values in this keyword, the last value will be used for the remaining target volumes.

Notes:

1. PERCENTUTILIZED is ignored when the target data set is preallocated.
2. PERCENTUTILIZED is not supported in an SMS environment. This keyword is valid only for logical data set restore operations.
3. PERCENTUTILIZED is ignored if no output volume is specified.

PROCESS



UNDEFinedsorg

specifies that the logical data set restore operation is to be allowed for data sets with undefined data set organization going to an unlike target device of a larger capacity and that DFSMSdss is to use track level I/O to perform data movement. This results in an exact track-for-track image of the source data set on target volume. To specify PROCESS, RACF authorization may be required.

Note: Even though the data is being copied to a device with a larger track capacity, the data may not fit on the output device. For example, if the source device is a 3380, and the output device is a 3390 and the data set's block size is less than 277 bytes, a track on the target cannot contain as much data as a track on the source and the message ADR366W (Invalid Track Format) is issued.

Related reading: For additional information about RACF authorization, see the *z/OS DFSMSdss Storage Administration Guide*.

PURGE



PURGE prevents a FULL (or TRACK) restore operation from ending while unexpired data sets remain on the target volume. For more information, refer to the REPLACE keyword for data set restore discussion.

Note: You must specify PURGE for a FULL volume RESTORE operation if the VVDS name on the target volume does not match the target volume serial number. This type of volume can be created using full volume COPY in conjunction with one of the following conditions:

- DUMPCONDITONING is specified
- Neither COPYVOLID nor DUMPCONDITONING are specified

REBLOCK



REBLOCK specifies that DFSMSdss is to reblock one or more of the selected sequential or partitioned data sets.

dsn Specifies the fully or partially qualified names of a sequential or partitioned data set to be restored and reblocked.

The REBLOCK keyword is ignored for:

- Unmovable data sets
- Data sets with record format of U (except for partitioned load modules)
- Data sets with a record format of V, VS, VBS, or F
- Partitioned data sets with note lists (except for partitioned load modules)
- partitioned data sets that are also specified in the NOPACKING keyword

Additionally, both the installation options exit and the installation reblock exit can override the specification of the REBLOCK keyword. The installation options exit can specify that no data set is to be reblocked. The installation reblock exit can specify whether a given data set is to be reblocked.

Some sequential and partitioned data sets have an attribute of being ‘reblockable’. These data sets may be automatically reblocked by DFSMSdss independent of the REBLOCK keyword.

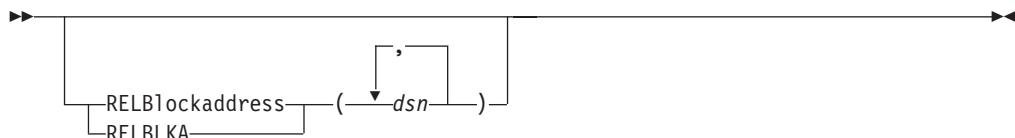
DFSMSdss uses the DASDCALC macro to determine the optimal block size for the target. The reblocking method used, DFSMSdss or DASDCALC, is presented to the installation reblock exit.

Related reading: For additional information about DFSMSdss processing of reblockable data sets, see the *z/OS DFSMSdss Storage Administration Guide*.

RECATALOG

See “CATALOG” on page 52.

RELBLOCKADDRESS

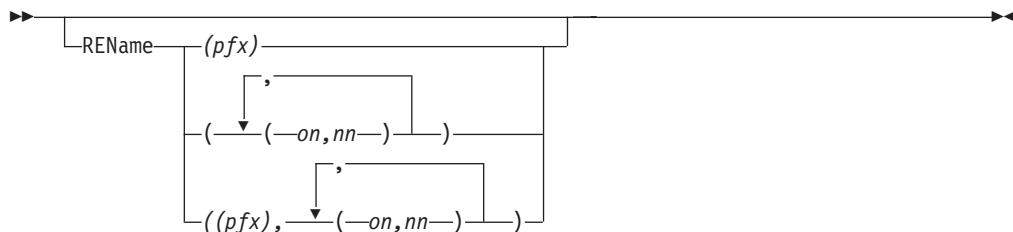


RESTORE Command

REBLOCKADDRESS identifies the direct access data sets whose names match the fully or partially qualified names specified (*dsn*). These direct access data sets are organized by relative block address instead of TTR and are to be restored block by block. DFSMSdss updates the block reference count (the relative position of the physical record as stored on its track) of dummy records. This keyword applies only to direct access data sets with fixed record formats and without standard user labels.

Note: If the data set is actually organized by TTR, the data set may become unusable.

RENAME



RENAME specifies that, if a data set with the old name exists on the output DASD volume, DFSMSdss is to allocate a new data set with the new name and restore the data set. If the data set with the old name does not exist on the volume, the data set is restored with the old name. For a VSAM data set that already exists on another DASD volume and is cataloged, the VSAM data set is restored with the new name unless the new name also exists and is catalogued.

VSAM data sets cannot be renamed during a physical data set restore. If a data set is preallocated with the new name, it is not restored. If a data set is not preallocated, it is restored using the old name. This keyword only applies to *movable* data sets; therefore, *unmovable* data sets will not be renamed. RENAME and RENAMEUNCONDITIONAL are mutually exclusive; you cannot specify these keywords together.

Note: If the RENAME keyword is specified in conjunction with the REPLACE keyword, only one of the keywords take effect for any particular data set. The RENAME keyword takes precedence over the REPLACE keyword. If a source data set name matches the RENAME criteria, then rename processing is performed and replace processing is not performed. If a preallocated target data set exists with the new name as chosen by the rename criteria, then the restore fails even if the REPLACE keyword was specified. If you want to replace a preallocated target with the new name, specify the REPLACEUNCONDITIONAL keyword. If a source data set name does not match the rename criteria, and a preallocated target data set with the source name exists, the preallocated target data set is replaced.

- pfx* Specifies the prefix to be used to replace the first-level qualifier of the data set name. It is optional, but if specified, must be the first parameter in the list of subkeywords. The prefix is used only if the *(on,nn)* parameters are not specified or the old name filters do not match the data set name.
- on* Specifies the old name to be used as filtering criteria for matching data set names on the target volume.
- nn* Specifies the new name to be used to derive the new data set name if the

data set name selected by the corresponding old name filtering criteria matches the name of a data set that already exists on the target volume.

If none of the old name filters match the data set name and the prefix is specified, the prefix is used to derive the new name. If old name filters do not match and the prefix is not specified, the data set is not renamed. If the old name filter matches and there is an error in the new name filter, the data set is not renamed.

The syntax for the **prefix** is as follows:

- Single-level, fully qualified, unquoted DSNAME.
- 8 characters or less.
- The first character must be alphabetic or national.
- The remaining characters can be alphanumeric or national.

The syntax for the **old name** filter is exactly like that of the INCLUDE filter, and their rules match. For more information, see the INCLUDE keyword for the RESTORE command.

Examples of valid syntax for the **new name** filter are:

**	Restore the data set with the old name. This provides a powerful tool whereby some data sets can be restored with the old name and others can be restored with the new name.
*	If DSNAME has one level, then restore with old name.
A.**	First level of DSNAME replaced by A.
A.B.**	First two levels of DSNAME replaced by "A.B".
*.A.**	Second level of DSNAME replaced by A.
**.BCD	Last level of DSNAME replaced by BCD.
DATE.**.LIST	First and last levels are replaced by DATE and LIST.
Q.*	If DSNAME has two levels, replace the first by Q.
Q.*.B	If DSNAME has three levels, replace the first and last by Q and B.
..SYSLIST	If DSNAME has three levels, replace the last by SYSLIST.
ABC.DEF	No asterisk in substring; replace the entire name with "ABC.DEF".

Examples of invalid syntax for the **new name** filter are:

.DATA.	Invalid (level to be replaced is ambiguous).
SYS	Invalid (a qualifier is not completely replaced).
SYS*	Invalid (a qualifier is not completely replaced).
*SYS	Invalid (a qualifier is not completely replaced).
SYS*TEM	Invalid (a qualifier is not completely replaced).
SYS.DAT%	Invalid (a qualifier is not completely replaced).

Restriction: The use of the wildcard character (%) is not supported for the new name filter of the RENAME, RENAMEU, or RENUNC keywords for the COPY or RESTORE operations.

You cannot change the number of qualifiers unless you use fully-qualified names, for example, RENUNC((A.B.C,A.B.C.D)).

RESTORE Command

If the new name filter has errors, the data set is not restored. The new name that is derived is truncated to fit 44 characters. If it ends with a period, that period is also truncated.

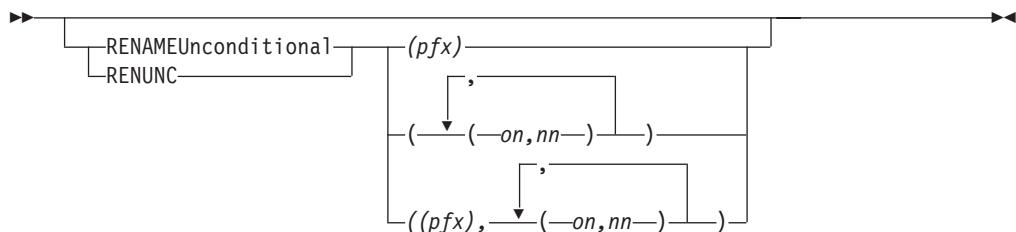
If the new name is not fully qualified, then it must contain the same number of qualifiers as the old name. For example, given the old name filter DATE.** and the new name filter DATE.*.*.LIST, DATE.MARCH.TODAY.OLDLIST would be renamed, but DATE.MARCH.OLDLIST would not.

GDG relative generation filtering cannot be used for old or new names.

Related reading:

- For additional information about the use of the RENAME keyword, see “Special Considerations for RESTORE” on page 178.
- For additional information about filtering, see “Filtering by Data Set Names” on page 12.

RENAMEUNCONDITIONAL



RENAMEUNCONDITIONAL specifies that the data set should be restored with the new name, whether or not the data set exists on DASD with the old name. If the data set exists on the volume with the new name, it is not restored.

RENAMEUNCONDITIONAL is not supported for VSAM data sets during a physical data set restore. This keyword only applies to *movable* data sets; therefore, *unmovable* data sets will not be renamed. If the old name filter matches and there is an error in the new name filter, the data set is not restored.

Notes:

1. RENAME and RENAMEUNCONDITIONAL are mutually exclusive; you cannot specify these keywords together. RENAMEUNCONDITIONAL is not supported for physical restore of VSAM data sets, and, if specified, the data sets will not be restored.
2. RENAMEUNCONDITIONAL specifies that the data set must be restored with the new name, regardless of whether the data set exists on DASD with the old name. If the data set exists on the target volume with the new name and REPLACEUNCONDITIONAL is not specified, an error message is issued and the data set is not restored.
3. If the RENAMEU keyword is specified in conjunction with the REPLACE keyword, only one of the keywords take effect for any particular data set. The RENAMEU keyword takes precedence over the REPLACE keyword. If a source data set name matches the RENAMEU criteria, then rename processing is performed and replace processing is not performed. If a preallocated target data set exists with the new name as chosen by the rename criteria, then the restore fails even if the REPLACE keyword was specified. If you want to replace a preallocated target with the new name, specify the REPLACEUNCONDITIONAL keyword. If a source data set name does not

match the rename criteria, and a preallocated target data set with the source name exists, the preallocated target data set is replaced.

- pfx* Specifies the prefix used to replace the first-level qualifier of the data set name. It is optional but, if specified, must be the first parameter in the list of subkeywords. The prefix is used only if the (*on,nn*) parameters are not specified or the old name filters do not match the data set name.
- on* Specifies the old name to be used as a filtering criterion to check if it matches the data set name.
- nn* Specifies the new name to be used to derive the new data set name if the data set name matches the corresponding old name filtering criterion.

Related reading:

- For additional information about the use of the RENAME keyword, see “Special Considerations for RESTORE” on page 178.
- For syntax rules, see the discussion of *pfx*, *on*, and *nn* under “RENAME” on page 204.

REPLACE



REPLACE specifies that DFSMSdss is to search the target volumes for usable preallocated data sets. If a usable preallocated data set is found, it will be replaced with the data set from the source volume. If no preallocated target is found, DFSMSdss attempts to allocate a data set.

DFSMSdss searches for preallocated data sets as follows:

- For SMS-managed data sets, DFSMSdss first searches in the standard order of search for a catalog entry for the data set.
- For VSAM data sets that are not SMS-managed, DFSMSdss searches the output volumes for preallocated data sets. If no output volumes are specified, DFSMSdss searches for a catalog entry for the data set. If no entry is found, DFSMSdss then searches the volumes on which the data set resided at dump time.
- For Non-VSAM data sets that are not SMS-managed, DFSMSdss searches the output volumes for preallocated data sets. If no output volumes are specified, DFSMSdss searches for a catalog entry for the data set. If no entry is found, DFSMSdss then searches the volumes on which the data set resided at dump time.
- If no preallocated target is found, DFSMSdss attempts to allocate a data set. DFSMSdss invokes the ACS routines to determine if the data set should be SMS managed. If it should be SMS managed, then allocation is done according to the SMS constructs. If the data set should not be SMS managed, the output volumes specified are used. If no output volumes were specified, the volume the data set resided on at dump time is used. If allocation is successful, the data set is restored.

PURGE is accepted and is treated the same as REPLACE. REPLACE only applies if the data set is not being renamed. REPLACEUNCONDITIONAL should be used when renaming a data set and a preallocated target data set should be replaced.

RESTORE Command

- The REPLACE and REPLACEUNCONDITIONAL keywords can not be specified together.

Notes:

1. REPLACE or REPLACEUnconditional must be specified to restore to preallocated data sets.
2. If the source data set is an extended-addressable VSAM data set, then the target must also be an extended-addressable VSAM data set.
3. If REPLACE is specified with the RESTORE command, the SMS constructs already associated with the preallocated data set remain the same.
4. CATALOG and RECATALOG are ignored for preallocated data sets.
5. The target data set name must match the source data set name.
REPLACEUnconditional must be specified to replace a target data set that has a name matching the rename criteria.
6. If you delete and reallocate a striped VSAM data set that is being replaced, DFSMSdss uses the same rules that apply to a new allocation. In general, it is best to not specify output volumes, or the VOLCOUNT keyword, when you are replacing striped VSAM data sets.
7. If the target data set is smaller than the source data set, DFSMSdss scratches the target data set and reallocates it with the size of the source data set.
8. Specify the FORCE keyword with the REPLACE keyword if you want to restore the data from unmovable data sets whose extents do not match.
9. If the RENAME or RENAMEU keywords are specified in conjunction with the REPLACE keyword, only one of the keywords take effect for any particular data set. The RENAME or RENAMEU keyword takes precedence over the REPLACE keyword. If a source data set name matches the RENAME or RENAMEU criteria, then rename processing will be performed and replace processing will not be performed. If a preallocated target data set exists with the new name as chosen by the rename criteria, then the restore fails even if the REPLACE keyword was specified. If you want to replace a preallocated target with the new name, specify the REPLACEUNCONDITIONAL keyword. If a source data set name does not match the rename criteria and a preallocated target data set with the source name exists, the preallocated target data set is replaced.
10. If the data set being restored is a large format sequential data set, but the preallocated target is not a large format sequential data set, the preallocated target will be used and turned into a large format sequential data set as long as it has enough allocated space for the data set being restored. If the preallocated new name target does not have enough allocated space, it will be scratched and reallocated as a large format sequential data set with enough space for the data set being restored.

Related reading: For additional information about the REPLACE keyword, see “Special Considerations for RESTORE” on page 178.

REPLACEUNCONDITIONAL



REPLACEUNCONDITIONAL specifies that DFSMSdss is to search the target volumes for usable preallocated data sets. If a usable preallocated data set is found,

it IS replaced with the data set from the source volume. When used with the RENAME or RENAMEUnconditional keywords, usable preallocated data sets with the new name are replaced. When used without the RENAME or RENAMEUnconditional keywords, usable preallocated data sets with the same name as the source data set are replaced. If no preallocated target is found, DFSMSdss attempts to allocate a data set. The REPLACE and REPLACEUnconditional keywords can not be specified together.

Notes:

1. REPLACEUnconditional must be specified to restore to preallocated data sets that match the rename criteria specified by the RENAME or RENAMEUnconditional keywords.
2. If the source data set is an extended-addressable VSAM data set, then the target must also be an extended-addressable VSAM data set.
3. If REPLACEUnconditional is specified with the RESTORE command, the SMS constructs already associated with the preallocated data set remain the same.
4. CATALOG and RECATALOG are ignored for preallocated data sets.
5. If DFSMSdss deletes and reallocates a striped VSAM data set that is being replaced, the same rules that apply to a new allocation are used to reallocate the data set. In general, it is best to not specify output volumes, or the VOLCOUNT keyword, when you are replacing striped VSAM data sets.
6. If the target data set is smaller than the source data set, DFSMSdss scratches the target data set and reallocates it with the size of the source data set.
7. If the data set being restored is a large format sequential data set, but the preallocated target is not a large format sequential data set, the preallocated target will be used and turned into a large format sequential data set as long as it has enough allocated space for the data set being restored. If the preallocated new name target does not have enough allocated space, it will be scratched and reallocated as a large format sequential data set with enough space for the data set being restored.

SHARE



SHARE specifies that DFSMSdss is to share, for read access with other programs, the data sets that are to be restored. The resetting of the data set change indicator is bypassed if SHARE is specified on a data set restore operation.

SHARE and FULL are mutually exclusive; you cannot specify these keywords together.

The SHARE keyword is not honored for VSAM data sets. Exclusive control is obtained for VSAM data sets even if the SHARE keyword is specified. Neither read access nor write access by other programs is allowed while the VSAM data set is being restored, regardless of the share options defined for the data set.

SPHERE



RESTORE Command

SPHERE specifies that for any VSAM cluster dumped with the SPHERE keyword, DFSMSdss must also restore all associated AIX clusters and paths. Individual sphere component names need not be specified; only the base cluster name is required.

Notes:

1. If an AIX is dumped without the SPHERE keyword, during a restore DFSMSdss treats the AIX as a normal VSAM KSDS.
2. The base cluster name must be specified to process the entire sphere. If the SPHERE keyword is specified but the base cluster name is not, none of the associations will be processed.

&

RSA

&



&

RSA

&

&

&

&

The RSA keyword allows you to specify a different label of an existing RSA private key that is present in the ICSF PKDS than what was specified on the RSA keyword during the DUMP command. The RSA public key is used to encrypt a randomly generated data key, so that the encrypted data key can be stored on the output medium.

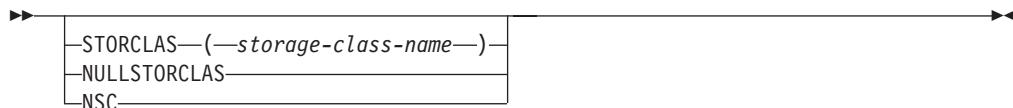
&

Note:

1. If the RSA keyword was specified during the DUMP command, it doesn't need to be specified on the RESTORE command when the same label for the same RSA private key exists in the ICSF PKDS.

&

STORCLAS

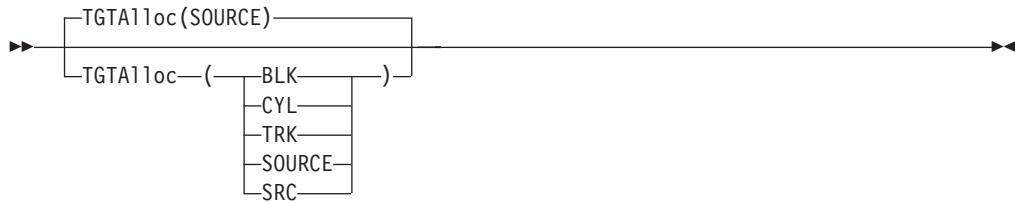


STORCLAS specifies the user-desired storage class that is to replace the source storage class as input to the ACS routines. The user must have the proper RACF authorization for the storage class specified. The keyword itself does not require authorization.

NULLSTORCLAS/NSC specifies that the input to the ACS routines is to be a null storage class rather than the source data set's storage class.

STORCLAS and NULLSTORCLAS are mutually exclusive.

Note: If BYPASSACS(dsn) is specified, all data sets that pass the BYPASSACS selection criteria are guaranteed the specified storage class. The combination of NULLSTORCLAS and BYPASSACS(dsn) forces the selected data sets to be non-SMS managed.

TGTALLOC

TGTALLOC specifies, during a logical data set restore function, how DFSMSdss will allocate the target data set.

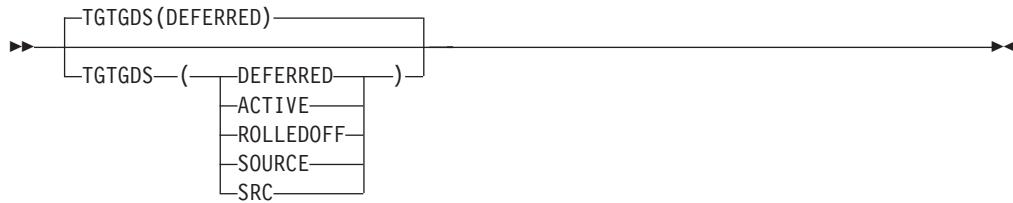
BLK	Specifies to allocate by blocks.
CYL	Specifies to allocate by cylinders.
TRK	Specifies to allocate by tracks.
SOURCE/SRC	Specifies to allocate with the same space allocation type as that of the source data set.

Notes:

1. If the TGTALLOC keyword is omitted, the target allocation defaults to source.
2. If SRC is specified and if the source data set is allocated by track or if TRK is specified, then the final VSAM allocation might be different from the requested one because of VSAM allocation rules.
3. If BLK is specified for VSAM data sets, TRK is used instead. The final VSAM allocation can be different from the requested one because of VSAM allocation rules.

RESTORE Command

TGTGDS



TGTGDS specifies in what status, during a data set operation, that DFSMSdss is to place nonpreallocated SMS-managed GDG data sets.

DEFERRED

Specifies that the target data set is to be assigned the DEFERRED status.

ACTIVE

Specifies that the target data set is to be assigned the ACTIVE status, for example, rolled into the GDG base.

ROLLEDOFF

Specifies that the target data set is to be assigned the rolled-off status.

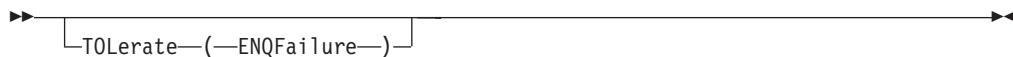
SOURCE/SRC

Specifies that the target data set is to be assigned the same status as that of the source data set.

Notes:

1. If the TGTGDS keyword is omitted, the target data set is assigned the DEFERRED status.
2. The requested target status of generation data sets must not violate generation data group rules.

TOLERATE

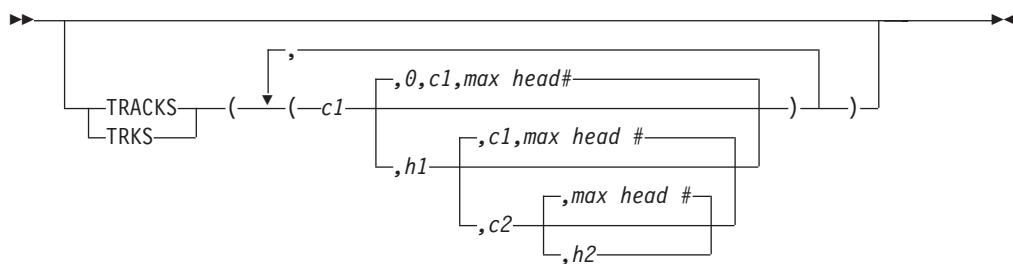


ENQFailure

Specifies that target data sets are to be processed even though shared or exclusive access fails. TOL(ENQF) and FULL or TRACKS are mutually exclusive; you cannot specify these keywords together.

For more information on TOL(ENQF), see Appendix B, "Data Integrity—Serialization," on page 287.

TRACKS



TRACKS specifies ranges of tracks to be restored (that is, a tracks restore operation). If any of the requested tracks are not in the input file, the restore operation is stopped.

- c1,h1* Specifies the cylinder and head number of the beginning of the range. Specify hexadecimal numbers as X'c1' or X'h1'.
- c2,h2* Specifies the cylinder and head number of the end of the range. Specify hexadecimal numbers as X'c2' or X'h2'.

Notes:

1. The *c2* must be greater than or equal to *c1*.
2. If *c2* equals *c1*, *h2* must be greater than or equal to *h1*.

DFSMSSdss verifies that the range is within the limits of the device. If you do not specify all four values for a range, DFSMSSdss provides the missing values unless the omitted value causes a syntax error. No intervening values can be omitted. For example:

Specified	Results
None	Syntax error
<i>c1</i>	<i>c1,0,c1</i> , maximum head number
<i>c1,h1</i>	<i>c1,h1,c1</i> , maximum head number
<i>c1,h1,c2</i>	<i>c1,h1,c2</i> , maximum head number
<i>c1,c2</i>	Syntax error
<i>,h1</i>	Syntax error

You cannot specify TOL(ENQF) with the TRACKS keyword.

Data Integrity Note: It is possible to duplicate a volume serial number during a restore TRACKS operation. Follow the procedure in “Data Integrity Considerations for Full or Tracks Restore Operations” on page 179 to maintain data integrity when restoring a volume with TRACKS restore.

Related reading: For additional information about using the TRACKS keyword during physical processing, see the *z/OS DFSMSSdss Storage Administration Guide*.

TTRADDRESS



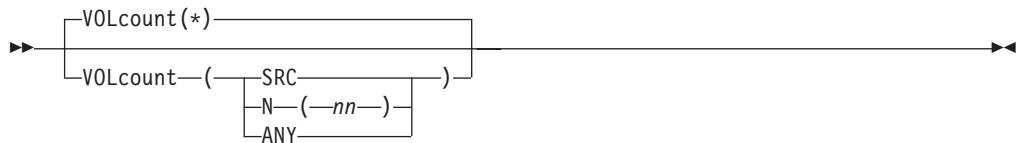
TTRADDRESS identifies the direct access data sets whose names match the fully or partially qualified names specified (dsn). These direct access data sets are organized by TTR rather than by relative block addressing and are to be processed track by track. The target device track capacity must be equal to or greater than the source.

RESTORE Command

Notes:

1. If a direct access data set is specified by both the RELBLOCKADDRESS and the TTRADDRESS keywords, the data set is not processed. Refer to the RELBLOCKADDRESS keyword for more information.
2. The TTRADDRESS keyword takes precedence over the AUTORELBLOCKADDRESS keyword processing for the specified data sets (dsn).

VOLCOUNT



VOLCOUNT specifies the method DFSMSdss uses to determine the number of volumes (volume count) for allocating the SMS target data set for a logical data set restore of VSAM or non-VSAM data sets.

* (asterisk)

Specifies that DFSMSdss determine the volume count for allocation according to the following conditions:

- If the source data set is a single-volume data set, one volume is allocated.
- If the source data set is a multivolume data set and either a list of volumes is not specified to DFSMSdss through OUTDDNAME or OUTDYNAM or there are no SMS volumes in the list, DFSMSdss allocates the same number of volumes that were in the multivolume source data set.
- If the source data set is a multivolume data set and a volume list is specified through OUTDDNAME or OUTDYNAM, the volume count is the number of SMS volumes in the list.

DFSMSdss does not adjust the final number of candidate volumes after the allocation is complete.

The * (asterisk) is the default for this keyword.

SRC Specifies that DFSMSdss *rely on the source volume count* to determine the number of volumes to allocate for the target data set as follows:

- If no output volume list is specified, DFSMSdss allocates the same number of volumes that the source data set had.
- If a volume list is specified through OUTDDNAME or OUTDYNAM, the volumes in the list that are SMS-managed must be in the same storage group, and the allocation must be directed to that storage group.

DFSMSdss does not adjust the final number of candidate volumes after the allocation is complete.

N(nn) *nn* represents the number of volumes to be used for SMS data set allocation. Any value between 0 and 59 may be specified with the following conditions:

- If *nn* is not zero and a volume list is specified through OUTDDNAME or OUTDYNAM, DFSMSdss allocates either the number of SMS volumes in the volume list or *nn*, whichever is less.

- If *nn* is zero and a volume list is specified through OUTDDNAME or OUTDYNAM, DFSMSdss allocates either the number of SMS volumes in the volume list or the number of volumes that were allocated for the source data set, whichever is less.
- If a volume list is specified through OUTDDNAME or OUTDYNAM and there are no SMS volumes in the list, or there is no volume list, DFSMSdss allocates either the number of volumes used by the source data set or *nn*, whichever is *more*.

DFSMSdss does not adjust the final number of candidate volumes after the allocation is complete.

- ANY** Specifies that DFSMSdss *use a maximum volume count* to allocate the SMS target data set as follows:
- DFSMSdss initially sets a volume count of 59 for the allocation.
 - If the data set is allocated on more volumes than were used to allocate the source data set, DFSMSdss reduces the number of volumes used to the number of primary volumes needed to satisfy the allocation.
 - If the data set is allocated on the same number or fewer volumes than were used to allocate the source data set, DFSMSdss reduces the number of volumes used to the number of volumes used for allocation of the source data set.

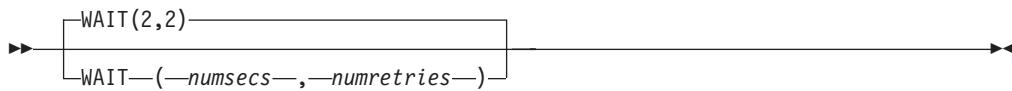
Notes:

1. VOLCOUNT does not convert any of the following data sets to multivolume: PDS or PDSE data sets, single-volume data sets whose organization is undefined, or empty non-VSAM, single-volume data sets.
2. VOLCOUNT does not change the number of volumes for key-range KSDS data sets.
3. Guaranteed space is not honored when VOLCOUNT(ANY) is used.
4. VOLCOUNT(ANY) does not support keyed VSAM data sets that have an imbedded index. If VOLCOUNT(ANY) is specified and a data set has an imbedded index, the data set is processed as if VOLCOUNT(*) were specified.
5. VOLCOUNT(ANY) does not support any type of striped data set (physical, sequential, extended, or VSAM). If VOLCOUNT(ANY) is specified and a data set is striped, the data set is processed as if VOLCOUNT(*) were specified.
6. When you specify VOLCOUNT(ANY), the &ANYVOL and &ALLVOL read-only variables are not available to the storage group ACS routine.
7. For nonguaranteed-space, striped VSAM data sets: The minimum number of volumes that DFSMSdss allocates is determined by the number of stripes, which is based on the STORCLAS sustained data rate (SDR). DFSMSdss does not consider the number of volumes in the output volume list, or any of the VOLCOUNT specifications. If there are not enough enabled volumes in the STORGRP to support the SDR, DFSMSdss reduces the number of stripes. If there are excess volumes specified, those volumes become nonspecific (*) candidates.
8. For guaranteed-space, striped VSAM data sets: DFSMSdss allocates the number of volumes that are specified in the output list, regardless of the SDR. (To be striped, the SDR must be greater than zero.) The VOLCOUNT rules described above apply.

You can override VOLCOUNT keyword settings with the options installation exit routine, as described in *z/OS DFSMS Installation Exits*.

RESTORE Command

WAIT



WAIT specifies to DFSMSdss the length of a wait in seconds and the number of retries to obtain control of a data set.

numsecs Specifies a decimal number from 0 to 255 that designates the time interval, in seconds, between retries.

numretries Specifies a decimal number from 0 to 99 that designates the number of retries to gain control of a data set.

The default for *numsecs,numretries* is (2,2), which specifies two retries at 2-second intervals. If you do not want to wait for a data set, specify 0 for either numsecs or numretries.

Note: The WAIT keyword does not control wait/retry attempts for system resources (such as the VTOC and the VVDS). For system resources, the default wait time is 3 seconds and the default retry count is 30. This results in a total wait time of 90 seconds.

Related reading: For information about controlling the wait/retry attempts for system resources, see the *z/OS DFSMSdss Storage Administration Guide*.

WRITECHECK



WRITECHECK specifies that the data restored is to be verified for successful completion. This keyword increases the overall elapsed time. The default is no WRITECHECK.

Note: The WRITECHECK keyword is not supported for extended-format sequential data sets.

DFSMSdss RESTORE Process

Table 4 describes, in decision table format, DFSMSdss restore actions for physical processing of SMS-managed data sets. The specified RESTORE command keywords and the existence of the data set are shown in the upper half. The actions taken are shown in the lower half.

Table 4. Physical Data Set Restore Actions on SMS-Managed Data Sets

	Non-VSAM Physical Restore														VSAM Physical Restore					
RENAME	N	N	Y	Y	Y	Y	-	-	-	N	N	Y	-	Y	Y	-	-	N	N	N
RENUNC	N	N	-	-	-	-	Y	Y	Y	N	N	-	Y	-	-	Y	Y	N	N	N
REPLACE	Y	Y	N	N	Y	Y	N	N	Y	N	N	N	X	-	-	-	-	Y	N	-
OLD DATA SET EXISTS	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	X	Y	N	Y	N	Y	Y
NEW NAME ON VOLUME	-	-	N	X	N	-	-	-	-	-	-	Y	Y	-	-	-	-	-	-	-
Overlay old data set	T	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	T	-	-
ALLOC with new name on USERVOL and restore	-	-	T	-	T	-	T	T	T	-	-	-	-	-	-	-	-	-	-	-
ALLOC with old name	-	T	-	T	-	T	-	-	-	-	T	-	-	-	T	-	-	-	-	T
Do not restore	-	-	-	-	-	-	-	-	-	T	-	T	T	T	-	T	T	-	T	-
Legend:																				
	Y = Yes T = Action taken N = No X = Doesn't matter - = Not Applicable																			

Figure 3 on page 218 describes *general* DFSMSdss actions for both physical and logical restore on non-VSAM data sets. It is *not* a program flowchart. Use it to clarify the restore actions on non-VSAM data sets under varying conditions.

Note: When following the path that includes REPLACEU, the flow diagram applies only to a logical restore operation.

RESTORE Command

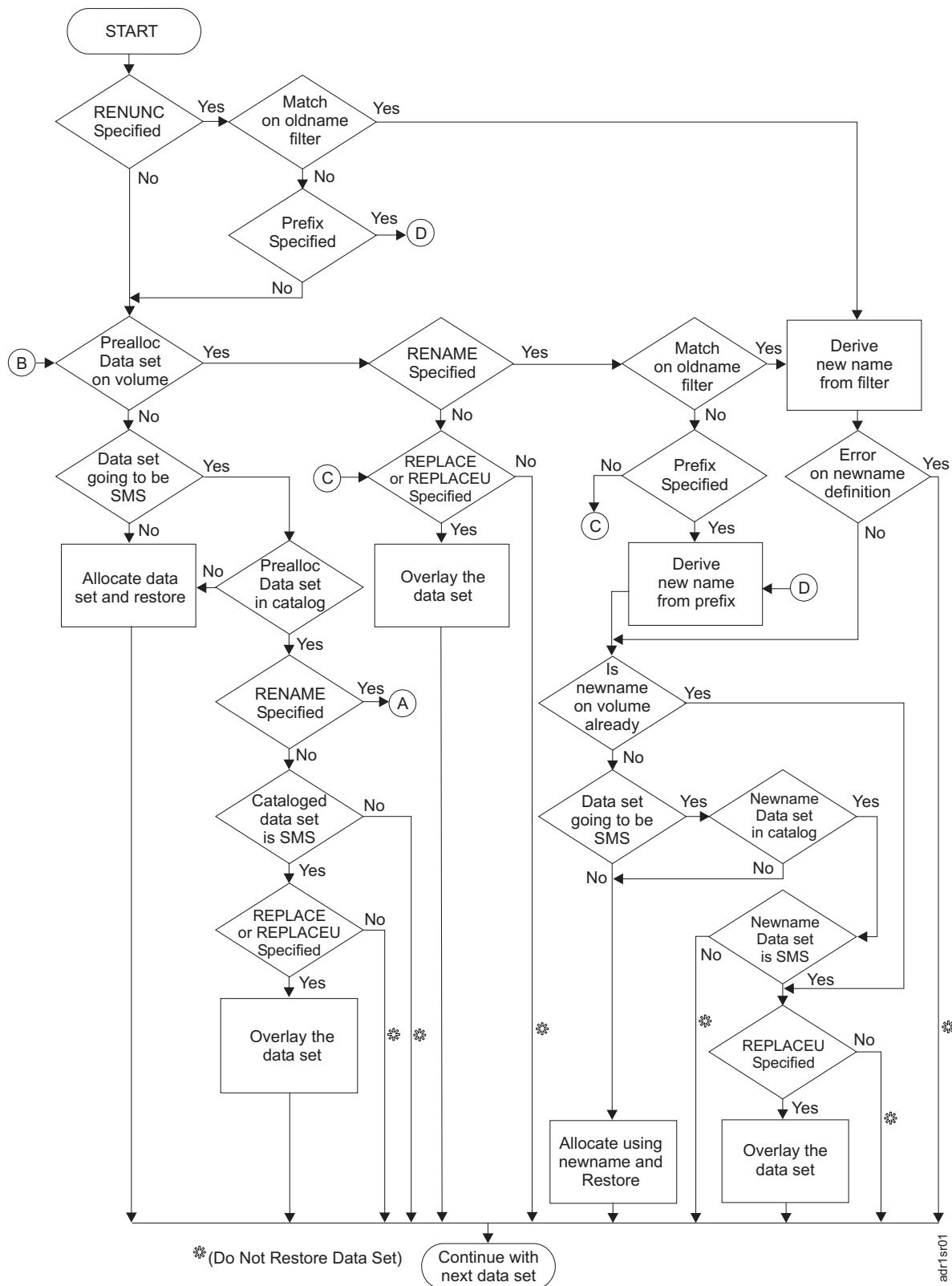


Figure 3. Restore Actions on Non-VSAM Data Sets. These actions apply to a data set RESTORE command with RENAME, RENAMEUNCONDITIONAL, REPLACE and REPLACEUNCONDITIONAL keywords.

Assignment of Class Names by Using the RESTORE and COPY Commands

In an SMS environment, you can use STORCLAS, MGMTCLAS, NULLSTORCLAS, NULLMGMTCLAS, and BYPASSACS keywords with the RESTORE and COPY commands to influence the class names assigned to a data set. Figure 4 shows how these keywords can influence the storage and management class names of the target data set in a restore or copy operation. However, Figure 4 only addresses how the *management and storage class* names are assigned, not how the *storage group* name is assigned or how volumes are selected.

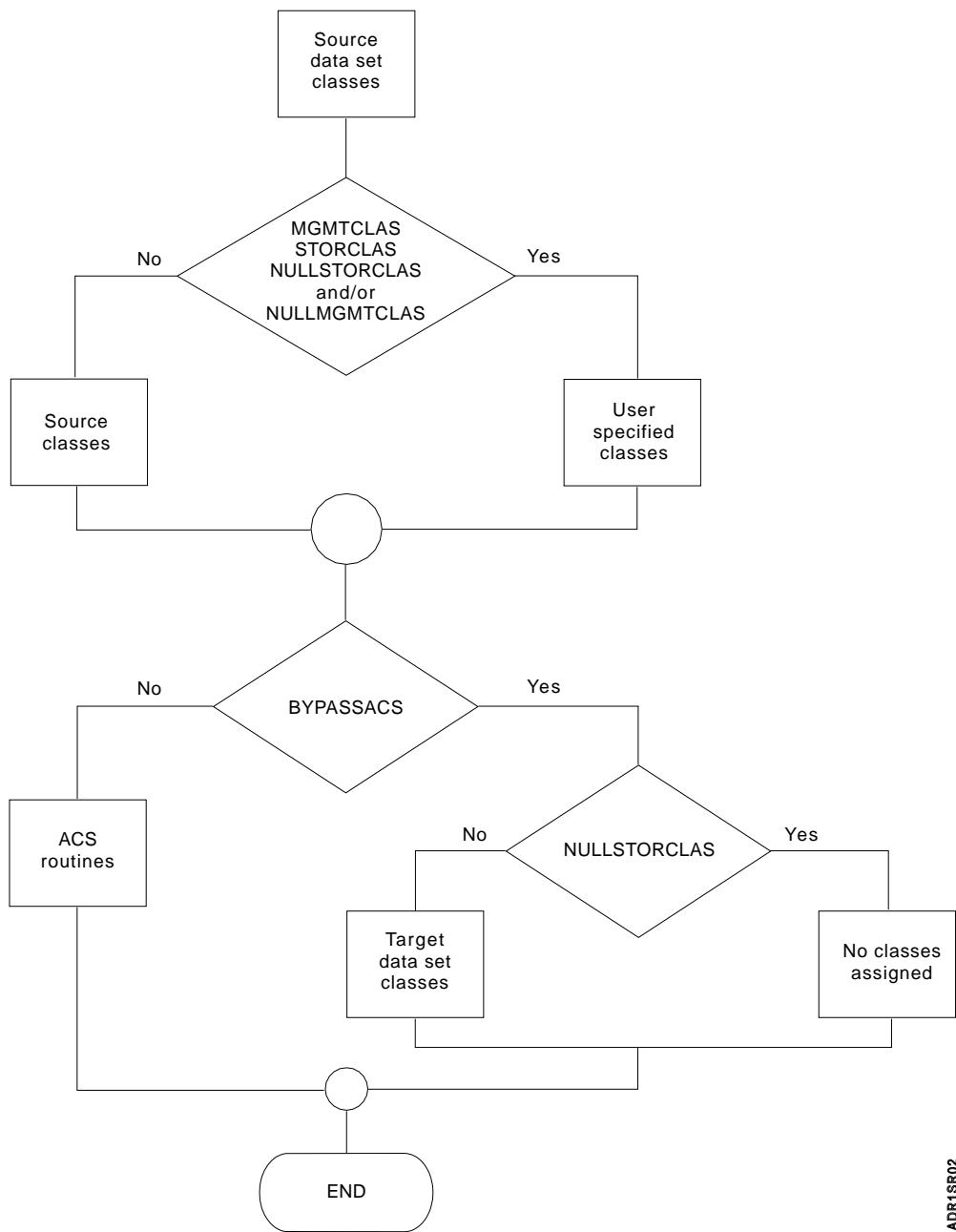


Figure 4. DFSMSdss Target Class Selection

RESTORE Command

Notes to Figure 4 on page 219:

- If you specify BYPASSACS, the source data set's class names or the class names specified with STORCLAS and MGMTCLAS are assigned to the target data set. If you do not specify BYPASSACS, ACS uses the source data set's class names, or the class names specified with STORCLAS and MGMTCLAS as input to assign the target data set's class names.
- If you specify NULLSTORCLAS, DFSMSdss passes a null storage class to ACS, which selects a storage class for the data set. If you specify NULLSTORCLAS and BYPASSACS together, the data set becomes non-SMS managed.
- NULLMGMTCLAS can only be used with SMS-managed data sets. Specifying NULLMGMTCLAS and BYPASSACS together causes the removal of the original management class of the data set.

Related reading: For information about how target volumes are selected during restore and copy functions, see the *z/OS DFSMSdss Storage Administration Guide*.

Examples of Full and Tracks Restore Operations

Example 1 shows that DASD volume numbered 111111 will be restored from the first data set of standard label tape volumes called TAPE01 and TAPE02.

The command input to be substituted for a full and tracks restore operation are shown below in Example 1A and 1B respectively. To dump the same volume, refer to Examples 1, 1A, and 1B of the DUMP command.

Example 1: DASD Volume-Restore Operation

```
//JOB1    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC PGM=ADRDSU
//SYSPRINT DD  SYSOUT=A
//TAPE     DD  UNIT=3480,VOL=SER=(TAPE01,TAPE02),
//          // LABEL=(1,SL),DISP=(OLD,KEEP),DSNAME=USER2.BACKUP
//DASD     DD  UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//SYSIN   DD  *
           command input (see Examples 1A and 1B below)
/*
```

Example 1A: Full Restore Operation

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD) PURGE
```

Example 1B: Tracks Restore Operation

```
RESTORE TRACKS(1,0,1,5) INDDNAME(TAPE) -
          OUTDDNAME(DASD) PURGE
```

Example 1C: Tracks RESTORE—Restore to Different Tracks

```
//JOBTRKS JOB accounting information,REGION=nnnnK
//STEP1 EXEC PGM=ADRDSU
//TAPE DD UNIT=3480,VOL=SER=TAPE04,
// LABEL=(1,SL),DISP=(OLD,KEEP),DSN=USER2.BACKUP
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
      RESTORE INDD(TAPE) OUTDYNAM(338000) -
          TRKS(200,10,200,10) /* INPUT TRK 200,10 ON SOURCE VOL*/ -
          OUTTRKS(100,0)      /*RESTORE TO 100,0 ON TARGET VOL */ -
          PURGE                /* OK TO OVERLAY THE TRK           */ -
          /* EVEN IF UNEXPIRED DATA SET AT THE LOCATION */ -
          PSWD(ABC/WRTPSWD)   /* PASSWORD FOR THE DATA SET      */
/*

```

A track of dump data is restored to a cylinder and head number other than that from which it was dumped. The dump tape (which might have resulted from a full, tracks, or data set dump operation) contains a track dumped from cylinder 200 head 10 that is restored to cylinder 100 head 0.

Examples of Physical Data Set Restore Operations

Example 2 specifies that data sets on standard label tape volume TAPE02 are to be restored to DASD volume numbered 111111.

Examples 2A through 2D below complement Examples 2A through 2G in “Examples of Physical Data Set Dump Operations” on page 152 in any combination. For example, the dump tape produced in Example 2C in “DUMP Command” can be used as the input tape for Example 2A below.

Example 2: Label Tape Volume RESTORED to DASD

```
//JOB2    JOB accounting information,REGION=nnnnK
//STEP1  EXEC PGM=ADRDSU
//SYSPRINT DD SYSOUT=A
//TAPE    DD UNIT=3480,VOL=SER=TAPE02,
// LABEL=(1,SL),DISP=(OLD,KEEP),DSNAME=USER2.BACKUP
//DASD1   DD UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//SYSIN   DD *
      command input (see Examples 2A, 2B, ... below)
/*

```

Example 2A: Using the INCLUDE Subkeyword to Restore All Data Sets on a Dump Tape

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD1) -
        DATASET(INCLUDE(**))
```

Example 2B: Using the INCLUDE and EXCLUDE Subkeywords

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD1) -
        DATASET(INCLUDE(**) -
                  EXCLUDE(**.LIST))
```

All data sets are restored, except those ending with a qualifier of LIST.

RESTORE Command

Example 2C: Using the INCLUDE, EXCLUDE, and BY Subkeywords

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD1) -  
        DATASET(INCLUDE(**) -  
                EXCLUDE(**.LIST) -  
                BY((EXPDT,LE,80045)))
```

All data sets that satisfy the BY subkeyword except those specified in the EXCLUDE subkeyword are restored.

Example 2D: With Filtering Data in a Data Set

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD1) -  
        DATASET(FILTERDD(A1))
```

Note to Example 2D: The following DD statement must be added to the JCL shown in Example 2:

```
//A1 DD DSNAME=USER2.FILTER,DISP=SHR
```

This cataloged data set (USER2.FILTER) contains three card-image records. The information shown below is positioned in columns 2 through 72 of each record:

```
INCLUDE(**) -  
EXCLUDE(**.LIST) -  
BY((DSCHA,EQ,1))
```

Example 2E: Using the LOGICALVOLUME and REPLACE Keywords

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD1) -  
        DATASET(INCLUDE(**)) LOGICALVOLUME(111111) -  
        REPLACE
```

Although the dump tape contains data sets from source volumes 111111 and 222222, only the data sets from source volume 111111 are restored. If preallocated data sets exist on the volume, DFSMSdss replaces them. Unmovable data sets that are not preallocated are not restored.

Example 2F: Using the REPLACE and RENAME Keywords

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD1) -  
        DATASET(INCLUDE(**)) LOGICALVOLUME(111111) -  
        REPLACE -  
        RENAME((USER2),(USER4.**,USER3.**))
```

In the above example, renaming takes place only for data sets that exist on DASD with the old name. Data sets with a first-level qualifier of USER4 are renamed to a

first-level qualifier of USER3. The first-level qualifiers of all other data sets are replaced by USER2. Unmovable data sets are not renamed.

Example 2G: Using the REPLACE and RENAMEUNCONDITIONAL Keywords

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD1) -
DATASET(INCLUDE(**)) LOGICALVOLUME(111111) -
REPLACE -
RENAMEUNCONDITIONAL((USER2),(*.PEAR.**,*.PLUM.**), -
(MY.SPECIFIC.DS,YOUR.ANY))
```

RENAMEUNCONDITIONAL is used for movable data sets; REPLACE is used for unmovable data sets. With the RENUNC keyword, movable data sets are renamed whether or not they exist on DASD with the old name. In the example, data sets with a second-level qualifier of PEAR are renamed by using a second-level qualifier of PLUM. MY.SPECIFIC.DS is renamed as YOUR.ANY. The first-level qualifier of all other movable data sets is changed to USER2. Unmovable data sets are not renamed.

Example 2H: Restore All Data Sets

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD1) -
WAIT (1,99) DATASET(INCLUDE(**))
```

Example 2H shows you what to do if the data sets are in use for a short time interval during a restore operation. DFSMSdss waits for a second at a time and retries as many as 99 times if the data set is in use by another job.

Example 2I: Restore Data Sets

```
RESTORE INDDNAME(TAPE) OUTDDNAME(DASD1) -
DATASET(INCLUDE(**)) TOL(ENQF) WAIT(0,0)
```

In Example 2I, DFSMSdss tries to serialize (ENQ) each data set. If the ENQ fails, DFSMSdss does not wait (WAIT(0,0)), and the data set is processed without serialization or enqueueing (TOL(ENQF)).

Examples of Logical Data Set Restore Operations

The following are examples of logical data set RESTORE operations.

RESTORE Command

Example 1: Logical Data Set RESTORE—Output Volumes Not Specified

```
//JOB5      JOB    accounting information,REGION=nnnnK
//STEP1     EXEC   PGM=ADRDSU
//TAPE      DD     UNIT=3480,VOL=SER=TAPE04,
//          LABEL=(1,SL),DISP=(OLD,KEEP),DSN=USER3.BACKUP
//SYSPRINT DD     SYSOUT=A
//SYSIN     DD     *
               RESTORE INDD(TAPE)      -
               DS(INCL(USER1.MULTVOL)) -
               REPLACE
/*
```

In Example 1, data set USER1.MULTVOL is restored. The location to which it is to be restored is not given. This RESTORE statement also applies when the data set has been scratched inadvertently after the dump operation. A multivolume data set is restored to the volumes from which it was dumped, provided the data set is preallocated on the output volumes. If it is not preallocated, the data set is restored to the first volume from which it was dumped that has adequate space to restore the data set as a single-volume data set.

The RESTORE statement can be modified as follows to support multiple restores of both single and multivolume data sets from dump tapes:

```
//SYSIN     DD     *
               RESTORE INDD(TAPE)      /* RESTORE           */
               DS(INCL(USER1.CNTL.**)) /* USER'S CONTROL DATA SETS */ -
               REPLACE                 /* OVERLAY DATA SETS IF THEY EXIST */
/*
```

Example 2: Logical Restore of an Unmovable Data Set

```
//JOB6      JOB    accounting information,REGION=nnnnK
//STEP1     EXEC   PGM=ADRDSU
//TAPE      DD     UNIT=3480,VOL=SER=TAPE04,
//          LABEL=(1,SL),DISP=(OLD,KEEP),DSN=USER4.BACKUP
//SYSPRINT DD     SYSOUT=A
//SYSIN     DD     *
               RESTORE INDD(TAPE) OUTDYNAM(338000) -
               DS(INCL(HIGH.PERF)) -
               FORCE                  /* TO FORCE RESTORE OF UNMOVABLE DATA SET */
/*
```

In Example 2, the unmovable data set HIGH.PERF does not currently exist on volume 338000. However, this data set did exist on volume 338000 at the time of the dump. You want to restore it to volume 338000 and would prefer it be restored to the location it originally occupied. However, the location where this data set would normally be restored is occupied by other data sets. So, you restore the data set to another place on the volume because you specify the FORCE keyword. The data set is marked as unmovable on volume 338000 because of one of the following situations:

- You do not want DFSMShsm to move it to an unlike device type.
- You do not want the data set to be relocated by a DEFrag operation for performance reasons.
- It was allocated as an ABSTR data set for performance reasons.

Example 3: Logical Data Set Dump, Followed by a Restore to an Unlike Device

```

//JOB1      JOB accounting information,REGION=nnnnK
//STEP1      EXEC PGM=ADRDSSU
//TAPE       DD UNIT=3480,VOL=TAPE01,
// LABEL=(1,SL),DISP=(NEW,CATLG),DSNAME=USER2.BACKUP
//SYSPRINT   DD SYSOUT=A
//SYSIN      DD *
      DUMP DATASET(INCL(USER2.OLDDDS)) -
                  OUTDD(TAPE)
/*
/*
//JOB2      JOB accounting information,REGION=nnnnK
//STEP1      EXEC PGM=ADRDSSU
//SYSPRINT   DD SYSOUT=A
//TAPE       DD UNIT=3480,VOL=TAPE01,
// LABEL=(1,SL),DISP=(OLD,KEEP),DSNAME=USER2.BACKUP
//DASD      DD UNIT=3380,VOL=(PRIVATE,SER=222222),DISP=OLD
//SYSIN      DD *
      RESTORE DATASET(
          INCLUDE(USER2.OLDDDS) ) -
          INDDNAME(TAPE) -
          OUTDDNAME(DASD) -
          RENAME(*.OLDDDS,*.NEWDS)
/*

```

In the first part of example 3, DFSMSdss dumps a cataloged data set (USER2.OLDDDS) from the source volume to an IBM standard label dump tape (TAPE01). Next, DFSMSdss restores USER2.OLDDDS from TAPE01 to a 3380 target volume (DASD volume 222222). The RENAME keyword is used to change the name of the data set to USER2.NEWDS.

Example 4: Dump and Restore for Storage Management Subsystem (SMS) Conversion

```

//JOB1      JOB accounting information,REGION=nnnnK
//STEP1      EXEC PGM=ADRDSSU
//SYSPRINT   DD SYSOUT=*
//DTAPE01    DD DISP=(,CATLG),DSN=V338001.USER3.BACKUP,
// LABEL=(1,SL),UNIT=3480,VOL=SER=TAPE01
//SYSIN      DD *
      DUMP -
          DS(INC(**)) -
          LOGINDYNA ( -
              (338001) -
          ) -
          DELETE PURGE COMPRESS -
          OUTDDNAME (DTAPE01)
/*

```

This initial part of “Example 4: Dump and Restore for Storage Management Subsystem (SMS) Conversion” dumps all the single-volume data sets on the non-SMS volume 338001 to TAPE01 with the DELETE option. The DELETE and PURGE keywords are required to avoid duplications in the restore operation.

RESTORE Command

```
//SYSPRINT DD SYSOUT=*
//DTAPE01 DD DISP=(OLD,KEEP),DSN=V338001.USER3.BACKUP,
//   LABEL=(1,SL),UNIT=3480,VOL=SER=TAPE01
//SYSIN DD *
  RESTORE DS(INC(**)) -
    STORCLAS(SC01MJA1) -
      INDDNAME (DTAPE01)
/*
```

This second part of “Example 4: Dump and Restore for Storage Management Subsystem (SMS) Conversion” on page 225 restores all of the data sets that were dumped in the first half of this example. Because no output volume is specified, most of the data sets will be allocated on SMS volumes throughout the system. The STORCLAS keyword indicates that the storage administrator wants the data sets to have a storage class of SC01MJA1. The ACS routines might or might not assign the target data sets that the storage class specified. All data sets that are not converted to SMS (ACS STORCLAS routine returns a null storage class) will be restored to the original volume.

Example 5: Using the RENAME Keyword to Restore a VSAM Data Set

```
//JOB3   JOB accounting information,REGION=nnnnK
//STEP1  EXEC PGM=ADRDSU
//TAPE   DD UNIT=3480,VOL=SER=TAPE04,
//   LABEL=(1,SL),DISP=(OLD,KEEP),DSN=USER3.BACKUP
//SYSPRINT DD SYSOUT=A
//SYSIN  DD *
  RESTORE INDD(TAPE) OUTDYNAM(338000) -
    DS(INCL(PARTS.VSAM1)) -
    RENAME(*.VSAM1,*.VSAM2) -
    CATALOG
/*
```

A VSAM key-sequenced data set, PARTS.VSAM1, is restored from a *logical* dump tape in this example. It is renamed as PARTS.VSAM2 and cataloged in the standard order of search. The cluster’s components, PARTS.VSAM1.DATA and PARTS.VSAM1.INDEX, are also renamed.

Example 6: Using the RECATALOG Keyword

```
//JOB4   JOB accounting information,REGION=nnnnK
//STEP1  EXEC PGM=ADRDSU
//TAPE   DD UNIT=3480,VOL=SER=TAPE04,
//   LABEL=(1,SL),DISP=(OLD,KEEP),DSN=USER3.BACKUP
//SYSPRINT DD SYSOUT=A
//SYSIN  DD *
  RESTORE INDD(TAPE) OUTDYNAM((338001),(338002)) -
    DS(INC(PARTS.**)) /* OR DS(INC(**)) */ -
    PCTU(80) -
    RECATALOG(USERCAT2) -
    TGTALLOC(SOURCE)
/*
```

In “Example 6: Using the RECATALOG Keyword,” data sets with a first-level qualifier of *PARTS* were dumped logically. All are restored to volume 338001 and cataloged in catalog USERCAT2. These data sets were on volume 338000 at the time of dump. If the data sets do not fit on volume 338001, a spill volume 338002

RESTORE Command

is specified. To ensure that the data sets on volume 338001 can be extended, 20% of the total space on volume 338001 is to be left as free space (PCTU(80)). TGTALLOC(SOURCE) specifies that the data sets are to be restored with the same allocation type they had when they were dumped.

Chapter 4. Syntax—Auxiliary Commands

This chapter describes the following **auxiliary** commands that you can use to further refine DFSMSdss processing:

- Writing to the Operator:
 - write-to-operator (WTO) command
 - write-to-operator with reply (WTOR) command
- Scheduling Tasks:
 - SERIAL command
 - PARALLEL command
- Controlling Task Processing:
 - SET command
 - IF-THEN-ELSE command sequence
 - EOJ command

Writing to the Operator

Use either the WTO or the WTOR command to direct a message to the system console. DFSMSdss prefixes the WTO and WTOR messages with a message ID of ADR111I and ADR112A, respectively.

When DFSMSdss encounters either a WTO or WTOR command, it waits until the last-requested function command completes before issuing the WTO or WTOR command.

WTO Command

The WTO command lets you write an ADR111I message to the system console that does not request a reply from the operator. The message cannot be greater than 247 characters. You must enclose the message within quotation marks. The command syntax is:

►►—WTO— '*message*' —►►

DFSMSdss assigns the following routing codes to the WTO message:

- | | |
|----|----------------------------|
| 2 | Master console information |
| 11 | Programmer information |

DFSMSdss assigns the following descriptor code to the WTO message:

- | | |
|---|------------|
| 6 | Job status |
|---|------------|

WTOR Command

The WTOR command lets you write an ADR112A message to the system console. The ADR112A message requests that the operator perform some action, and then issue a reply. You can use WTOR, for example, to request that the operator mount a required volume or quiesce a data base before your DFSMSdss job continues to process. The WTOR message cannot be greater than 114 characters. You must enclose the message within quotation marks. The command syntax is:

Syntax—Auxiliary Commands

►►WTOR— 'message' —►►

DFSMSdss assigns the following routing code to the WTOR message:

- 1 Master console action

DFSMSdss assigns the following descriptor code to the WTOR message:

- 2 Action that is required

Scheduling Tasks

You can use the SERIAL or PARALLEL commands to schedule tasks. The SERIAL or PARALLEL command must precede the commands to be executed in the SERIAL or PARALLEL mode.

SERIAL Command

The SERIAL command lets you re-initiate serial task scheduling (only one task at a time) after you have used parallel task scheduling. Tasks are processed in the order in which they appear in the input stream.

If neither the SERIAL nor PARALLEL command has been issued, SERIAL is the default. The command syntax is:

►►SERial—►►

PARALLEL Command

The PARALLEL command initiates parallel task scheduling wherein multiple tasks are processed concurrently. When you use parallel processing, commands may not process in the order in which they appear in the input stream. The PARALLEL command is effective only when the required system resources (virtual storage, DASD, or tape volumes) are available. If there are resource conflicts (multiple tasks that use the same DASD or the same tape volume) or if there is not enough virtual storage that is available for all of the tasks to run concurrently, some tasks can be delayed until the resources are available (other tasks end).

DFSMSdss attempts to delay the initiation of additional tasks if there are insufficient available resources to initiate the tasks. DFSMSdss is not able to accurately assess which additional resources that a task might later require. If you attempt to run too many tasks in parallel for the available resources, it can result in unpredictable failures. In such cases, the user must establish a safe upper boundary on the number of tasks that can be supported as parallel tasks.

When you switch between SERIAL and PARALLEL modes, DFSMSdss waits for the completion of all previously scheduled tasks before switching. If DFSMSdss is in PARALLEL mode, an IF statement ensures that all prior commands are processed. The command syntax is:

►►PARallel—►►

Controlling Task Processing

With the SET, IF-THEN-ELSE, and EOJ commands, you can direct DFSMSdss through a logical path in your command sequence, based on the condition (return) codes of previously completed operations. In addition, you can temporarily set patch bytes with the PATCH parameter of the SET command.

SET Command—Setting Condition Codes and Patch Bytes

As an alternative to setting flags in ADRPATCH (a module in the DFSMSdss load module ADRDSSU), you can customize certain DFSMSdss functions by temporarily setting patch bytes during DFSMSdss processing through a SET PATCH command.

The SET command also allows you to set the LASTCC and MAXCC variables to any value from 0 to 16, inclusive. By doing so, you can influence the logical path that DFSMSdss takes in the command sequence following the SET command.

You can set the following condition codes with the SET command. Use an IF-THEN-ELSE command sequence to test for these condition codes.

- 0** The function was processed as expected. Informational messages may have been issued.
- 4** A problem occurred, but processing continued. The result may not be exactly what you wanted, but no permanent harm was done. A warning message was issued.
- 8** A function did not process, began processing but ended prematurely, or the job ran without processing all requested functions. An error message is issued. If an abend occurs in any of the DFSMSdss subtasks, the return code is set to 8.
- 12** The job did not process. No functions were processed.
- 16** A function processed and left at least one volume or data set in an unusable condition. For example, a full-volume dump operation ended prematurely, leaving the output tape in an unusable condition.

If you are running a batch job, the condition codes that are tested in the IF-THEN-ELSE command sequence, and that can be set by the SET command, cannot be passed from one job step to the next. However, the final maximum condition code is passed to the MVS system when DFSMSdss returns control to the system at the completion of step processing. **Do not use SET LASTCC before the first function command.**

You can determine the condition code of the last-requested operation (LASTCC) and the maximum code of all completed operations (MAXCC) with an IF command. When an IF MAXCC or SET MAXCC command is encountered, DFSMSdss waits for all previously requested function commands to complete before the highest return code is determined. Also, when an IF LASTCC or SET LASTCC command is encountered, DFSMSdss waits for the last-requested function command to complete before the return code is determined. After the return code is determined and if the condition code is tested and satisfied, DFSMSdss ends the command or commands following the THEN keyword. If the tested condition is not satisfied, DFSMSdss bypasses the command or commands following the THEN keyword.

SET Command

The syntax of the SET command is:



The SET command allows you to customize certain DFSMSdss functions by temporarily setting patch bytes. You can also influence the logical path that DFSMSdss takes in the command sequence that follows the SET command. A SET command that occurs within an unprocessed THEN or ELSE clause is not processed.

PATCH	Specifies that DFSMSdss set the patch byte, at offset <i>offset</i> , to the value specified with <i>value</i> .
	Your installation can limit the use of the SET PATCH command with the RACF FACILITY-class profile STGADMIN.ADR.PATCH. You need READ access authorization to that profile to use the SET PATCH command.
<i>offset</i>	Specifies, in hexadecimal, the value for the patch byte offset. The maximum offset that you can specify is X'0FFF', and the minimum offset that you can specify is X'08'.
<i>value</i>	Specifies, in hexadecimal, the value that DFSMSdss assigns to the patch byte at the specified offset. The value must be in the range X'00' to X'FF'.
MAXCC	Specifies that the MAXCC be set to a new condition-code value. Setting MAXCC does not affect the value of LASTCC.
LASTCC	Specifies that the LASTCC be set to a new condition-code value. If the value assigned to LASTCC is higher than the value of MAXCC, MAXCC is also set to the higher value.
<i>number</i>	Specifies the value assigned to MAXCC or LASTCC. The maximum value that can be assigned is 16; a higher value is reduced to 16.

Examples of the SET Command: The examples that follow show the use of the SET command.

To set the patch byte at offset X'08' to X'FF', specify the following:

```
SET PATCH 8 = FF
```

To set the patch byte at offset X'44' to X'25', specify the following:

```
SET PATCH 44 = 25
```

To set the last condition code established to 12, specify the following:

```
SET LASTCC=12
```

To replace the highest condition code established in processing so far with 8, specify the following:

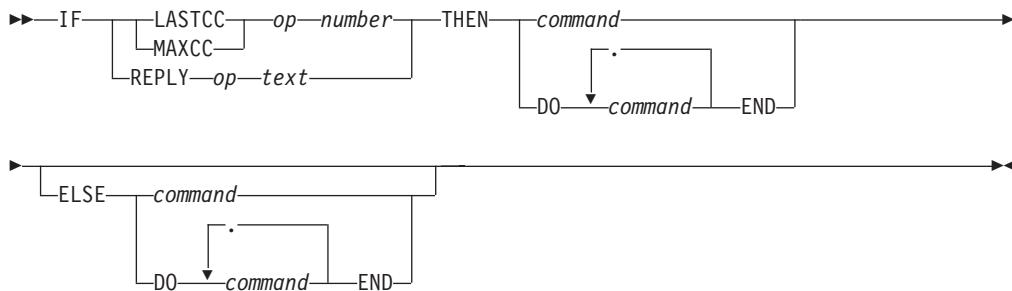
```
SET MAXCC=8
```

IF-THEN-ELSE Command Sequence—Using Condition Codes

Condition codes are used to set up the IF-THEN-ELSE statements. LASTCC specifies a comparison to the last condition code and MAXCC specifies the maximum condition code for comparison.

IF-THEN-ELSE Command Sequence

The syntax of the IF-THEN-ELSE command sequence is:



IF	Specifies that a comparison is to be made. The outcome determines which logical path (in your command sequence) DFSMSdss takes.
LASTCC	Specifies that LASTCC (condition code of the last-requested function, such as COMPRESS, CONVERTV, COPY, COPYDUMP, DEFrag, DUMP, PRINT, RELEASE, RESTORE) be compared to a specified number.
MAXCC	Specifies that MAXCC (maximum condition code of all completed operations) be compared to a specified number. MAXCC is initialized to zero upon entry to DFSMSdss.
REPLY	Specifies that DFSMSdss compare the operator response to the last SYSIN command stream to the text specified with the <i>text</i> variable of the REPLY keyword.
op	Specifies an operator that describes a comparison that DFSMSdss will make. For example, you may have specified a returned condition code with either the LASTCC or MAXCC keyword. You can compare this value to the value of the <i>number</i> variable that you specify after the specific operator. You can also compare the operator's response to a WTO command to the text that you specify with the <i>text</i> variable of the REPLY keyword. The compare operator can perform any one of six possible comparisons:
	EQ or = Equal to
	LE or <= Less than or equal to
	LT or < Less than
	GT or > Greater than
	GE or >= Greater than or equal to
	NE or != Not equal to
number	Specifies the decimal integer that is to be compared with MAXCC or LASTCC. Values greater than 16 are reduced to 16.
text	Specifies the text that is to be compared with the operator's response to the WTO command. You must enclose the text string within quotation marks.

Syntax—Auxiliary Commands

THEN	Specifies that a single command or a group of commands (enclosed by DO and END) is to be processed if the tested condition is satisfied. THEN can be followed by another IF command.
ELSE	Specifies that a single command or a group of commands (enclosed by DO and END) is to be processed if the tested condition is not satisfied. ELSE can be followed by another IF command. The ELSE clause cannot be on the same line as the THEN clause nor on the same line as the continuation of the THEN clause.
DO	Specifies that the group of commands that follows is to be treated as a single unit, that is, to be processed as a result of a single IF command. The set of commands is ended by END. A command following a DO must begin on a new line.
END	Specifies the end of a set of commands initiated by the nearest unended DO. END must be on a line by itself.

Creating a Null Command

If THEN or ELSE is not followed by a continuation character or by a command on the same line, the result is a null command. A semicolon after the THEN or ELSE keyword also results in a null command. A null command specifies that no action is taken if the IF clause is satisfied (a null THEN command) or if the IF clause is not satisfied (a null ELSE command).

To specify a null THEN command, specify:

IF ... THEN or IF ... THEN;
 ELSE ... ELSE ...;

To specify a null ELSE command, specify:

IF ... THEN ... or IF ... THEN ...
 ELSE ELSE; ;

Continuation Rules for IF-THEN-ELSE Command Sequencing

The following continuation rules apply for the IF-THEN-ELSE command sequencing.

1. IF (condition) must be followed by a THEN on the same line or a continuation of that line.

Examples:

IF LASTCC = 0 THEN COPYDUMP ...

or

IF LASTCC = 0 -
 THEN COPYDUMP ...

2. THEN must be followed by a command or DO on the same line or a continuation of that line.

Examples:

IF LASTCC = 0 -
 THEN -
 COPYDUMP ...

or

IF LASTCC = 0 -
 THEN DO
 COPYDUMP ...
 PRINT ...
 END

3. ELSE must be the first word on a line and no continuation character should be used on the preceding line.

Example:

```
IF LASTCC = 0      -
  THEN
    COPYDUMP ...
  ELSE      -
    PRINT ...
```

Nesting IF Commands

An IF command in a THEN or ELSE clause is a nested IF command. The maximum level of nesting is 10, starting with the first time you specify IF.

Within a nest of IF commands, the innermost ELSE clause is associated with the innermost THEN clause, the next innermost ELSE clause with the next innermost THEN clause, and so on. If there is an IF command that does not require an ELSE clause, use a null ELSE clause (see “Creating a Null Command” on page 234), unless the nesting structure does not require one. If a nesting structure does not require a null ELSE clause, the DFSMSdss job stream tells you.

Examples of Controlling Task Processing for the IF-THEN-ELSE Command

The following are examples of controlling task processing for the IF-THEN-ELSE command.

Example 1: Nested IF commands are used to determine whether a COPYDUMP, EOJ, or PRINT command is to be processed:

```
IF LASTCC > 4      -
  THEN IF MAXCC < 12      -
    THEN COPYDUMP ...
    ELSE EOJ
  ELSE IF LASTCC = 4      -
    THEN
    ELSE PRINT...
```

If the value of LASTCC is greater than 4, the value of MAXCC is to be tested. If the value of MAXCC is less than 12, the COPYDUMP command is processed. Otherwise, the EOJ command is processed. If LASTCC is 4, no action is taken. If LASTCC is less than 4, the PRINT command is processed.

Example 2: Nested IF commands are used to determine whether a COPYDUMP or a PRINT command is to be processed:

```
IF LASTCC > 4      -
  THEN IF MAXCC < 12      -
    THEN COPYDUMP ...
    ELSE
  ELSE IF LASTCC = 4      -
    THEN PRINT ...
```

If the first IF clause finds that LASTCC is greater than 4 and the second IF command finds that MAXCC is 12 or greater, no function command is processed. The null ELSE command specifies that the next ELSE corresponds to the first THEN.

Common Continuation Errors

The continuation rules, described in Chapter 1, “Specifying DFSMSdss Commands,” on page 1, must be followed carefully when auxiliary commands,

Syntax—Auxiliary Commands

comments, or blank records appear in your input. You must also be careful not to inadvertently specify a null clause when continuing auxiliary commands.

The following examples show common continuation errors.

Error Example 1:

```
IF LASTCC = 0 -
THEN
PRINT ...
```

A continuation character (hyphen) is missing after *THEN*; consequently, a null *THEN* clause is assumed. The *PRINT* command is unconditionally processed.

Error Example 2:

```
IF LASTCC = 0 -
THEN -
COPYDUMP ...
/* ALTERNATE PATH */
ELSE -
PRINT ...
```

Because no continuation character (hyphen) follows the comment, a null *ELSE* clause is assumed. *ELSE* is not matched with *THEN*, and an error message is issued. The *PRINT* command is ignored. Notice the correct use of the continuation character on the other lines.

Error Example 3:

```
PRINT INDD( - /*COMMENT*/
DDN1)
```

The *DDN1* on the second line is ignored and nothing is printed because characters other than blanks appear after the continuation character (hyphen).

EOJ Command—Ending Your DFSMSdss Step

With the end-of-job (EOJ) command, you can end your DFSMSdss step after the currently processing operation or scheduled tasks are completed. The command syntax is:

►► EOJ —————►►

Chapter 5. DFSMSdss Stand-Alone Services

This section, which describes the Stand-Alone Services function, is intended for the storage administrator, the system programmer, and anyone who runs the Stand-Alone Services program. IBM offers this Stand-Alone data recovery solution to all users of the DFSMSdss software package for DFSMS/MVS® Version 1 Release 4 and above.

Stand-Alone Services can perform either a full-volume or a tracks restore from dump tapes produced by DFSMSdss or DFDSS, and offers the following benefits:

- Provides user-friendly commands to replace the previous control statements
- Supports IBM 3494 and 3495 tape libraries and 3590 tape subsystems
- Supports IPLing from a DASD volume, in addition to tape and card readers
- Allows you to predefined the operator console to be used during Stand-Alone Services processing

Preparing to Run the Stand-Alone Services Program

The Stand-Alone restore function is a single-purpose program designed to allow the system programmer to restore vital system packs during disaster recovery without needing to rely on an MVS environment. Stand-Alone Services runs independently from a system environment either as a “true” stand-alone system or under a VM system.

This section helps you to prepare your environment before you IPL and run the Stand-Alone Services program. Information is provided about running Stand-Alone Services in different processor operating modes, running with a predefined console, setting up tape library and device options, and using command syntax and processing options.

The Stand-Alone Services program operates on an IBM System/370™ (S/370™) processor in either MVS/ESA™ mode, MVS/XA™ mode, or S/370 mode. It also runs on an IBM System/390® processor in either S/390® mode or S/370 mode. The Stand-Alone Services program can run on a processor that is in BASIC or LPAR mode, or you can run the Stand-Alone Services program in a virtual machine under VM.

The Stand-Alone Services program operates in extended control (EC) mode and requires 2MB of real storage.

Virtual Machine (VM) Note: To specify EC mode while the Stand-Alone Services program is running under VM/370, enter:

CP SET ECMODE ON

Running Stand-Alone Services in 370 Mode

The following conditions apply to Stand-Alone Services operations in 370 mode:

- Tape libraries are not supported by Stand-Alone Services in 370 mode.
- The initial program load (IPL) device and the console must be attached to the same processor you IPLed from (when there are two or more processors).

Stand-Alone Services

- For DASD and tape devices:
 - In 370 mode, Stand-Alone Services *does not issue* Assign or Unassign commands for tape devices, or Device Reserve or Device Release commands for DASD devices.
 - The user must ensure that all devices to be used by Stand-Alone Services are not accessed by other systems during IPL and while Stand-Alone Services operations are in process.
- Potential interference from other devices can occur as follows:
 - When the Stand-Alone Services is IPLed and loaded with the operator console not predefined, a Wait PSW is loaded with the rightmost bytes containing X'FFFFFF'. The Stand-Alone Services program waits for the operator to identify the operator console. The first interrupt presented at this time is expected to be a console and is treated as such.
 - When the operator console is predefined and a problem is detected while attempting initial communication with the predefined console, Stand-Alone Services loads a Wait PSW with the rightmost bytes containing X'DDDDDD', giving the operator an opportunity to identify a console (other than the predefined console) to be used as the operator console.

Other devices can generate interrupts that interfere with Stand-Alone Services operations. If this happens, determine which device is causing the interference and follow your installation's procedures to prevent the device from interrupting until the Stand-Alone Services operation is complete.

Running Stand-Alone Services in XA or ESA Mode

The following conditions apply to Stand-Alone Services operations in XA or ESA mode:

- For DASD and tape devices:
 - In XA or ESA mode, Stand-Alone Services issues Assign and Unassign commands for tape devices, and Device Reserve and Device Release commands for DASD devices (if the command is supported by the device).
- **VM Note:** When running Stand-Alone Services under VM, if the Stand-Alone Services program does not run to completion or is unable to free the device, an Assign or Reserve condition may be left outstanding.
- The user must ensure that the devices to be used by Stand-Alone Services are not accessed by other systems while Stand-Alone Services IPL and operations are in progress.
- Potential interference from other devices can occur as follows:
 - When the Stand-Alone Services is IPLed and loaded with the operator console not predefined, a Wait PSW is loaded with the rightmost bytes containing X'FFFFFF'. The Stand-Alone Services program waits for the operator to identify the operator console. The first interrupt presented at this time is expected to be a console and is treated as such.
 - When the operator console is predefined and a problem is detected while attempting initial communication with the predefined console, Stand-Alone Services loads a Wait PSW with the rightmost bytes containing X'DDDDDD', giving the operator an opportunity to identify a console (other than the predefined console) to be used as the operator console.

Other devices can generate interrupts that interfere with Stand-Alone Services operations. If this happens, determine which device is causing the interference. If the interrupting device is not on the same channel path ID (CHPID) as the devices you are using for Stand-Alone Services processing, configure that

CHPID offline, re-IPL the Stand-Alone Services program, and configure the CHPID back online after processing is complete. If the interrupting device is on the same CHPID as the devices you are using for Stand-Alone Services processing, follow your installation's procedures to prevent the device from interrupting until after the operator console has been identified.

Running Stand-Alone Services with a Predefined Console

Use the OPERCNSL keyword of the BUILDSA command to predefine the console when creating the Stand-Alone Services program. The OPERCNSL keyword allows you to specify the address of the device to be used as the operator console, or you can specify OPERCNSL(SERV) to use an ES/9000 service console.

The device must also be a valid device *within the configuration that is running the Stand-Alone Services program*. DFSMSdss performs limited validation of the OPERCNSL keyword during the BUILDSA processing. This is because the Stand-Alone Services program may be run with a system configuration different from the one that is used to build the core image with the BUILDSA command. (The core image is the executable module that is loaded into the processor's storage during IPL).

After it is IPLed, Stand-Alone Services attempts to use the predefined device as the operator console instead of waiting for the first interrupt to identify the operator console. If a problem with the predefined device is detected, the processor enters a wait-state with the rightmost bytes of the PSW containing "DDDDDD". This PSW indicates that Stand-Alone Services is unable to use the predefined device and is waiting for the operator to identify another console to be used as the operator console. If this happens, do the following:

1. Determine the cause of the problem with the predefined device and take steps to correct the problem. Some of the possible reasons for a problem being detected are listed below.
2. Generate an interrupt on a different console that you can use as the operator console.

The following are some of the reasons for a problem being detected with the predefined console:

- An error has occurred during the Stand-Alone Services program's initial communication with the predefined console.
- The console address may have been incorrectly specified when the Stand-Alone Services program IPL-able core image was built. For example, the address that was specified with the OPERCNSL keyword of the BUILDSA command does not exist in the configuration in which the Stand-Alone Services program is being IPLed.
- The device at the address specified with the OPERCNSL keyword cannot be identified as a supported operator console.
- If the console was predefined to be the service console, the necessary features may not exist on the processor for Stand-Alone Services to communicate with the console.

Using a Tape Library

This section explains how to use the IBM 3494 and 3495 tape libraries to IPL Stand-Alone Services and restore your dump data set tapes, and how to use the tape library menu options.

Notes:

1. Stand-Alone Services supports IBM tape libraries in XA or ESA mode only.

Stand-Alone Services

2. Tape drives to be used for Stand-Alone Services must remain offline to other systems.
3. The IPL tape must be mounted and ready prior to performing the IPL.

The Stand-Alone Services RESTORE and TAPECNTL commands are supported by devices within IBM 3494 and 3495 tape libraries. Table 5 shows the options available when you use a tape library to IPL the core image or restore from dump tapes. Stand-Alone Services can use the tape library in different ways depending on the features that exist on the tape library.

IPLing and Restoring from a Tape Library

Use one of the options shown in Table 5 to either IPL the Stand-Alone Services program or to restore data from dump data set tapes.

Table 5. Stand-Alone Services Options when Using an IBM 3494 or 3495 Tape Library. Procedures referred to in this table appear later in this section.

Task ↓	Can You Perform the Task Using an IBM Tape Library with:		
	No “Setup Stand-Alone Device” Feature?	“Setup Stand-Alone Device” Feature Only?	Both “Setup Stand-Alone Device” and “Transient Mount” Features?
IPL the Core image	No.	Only for tapes inside the library. Use Procedure A.	For tapes inside the library, use Procedure A. For tapes outside the library, use Procedure B.
Restore from Dump Tapes	Only for tapes inside the library. Use the TAPEVOLSER keyword of the RESTORE command.	Only for tapes inside the library. Either use Procedure A or use the TAPEVOLSER keyword of the RESTORE command. Only one method can be used for a single invocation of the RESTORE command.	For tapes inside the library, either use Procedure A or use the TAPEVOLSER keyword of the RESTORE command. For tapes outside the library, use Procedure B. Only one method can be used for a single invocation of the RESTORE command.

Note: When the “Setup Stand-Alone Device” feature is used to mount tapes, Stand-Alone Services treats the tape drive as if the drive is not part of a tape library.

Identifying Procedures to Mount and Demount Tapes Using the IBM Tape Library Stand-Alone Device Setup Features

The following procedures to mount and demount the Stand-Alone Services IPL tape and dump data set tapes are referenced within Table 5. Use Procedure A for tapes that reside inside the library. Use Procedure B for tapes that reside outside the library.

Note: When both the IPL tape and the dump data set tapes are mounted from the input station (Transient Mount), the same tape drive must be used for both the IPL tape and the dump data set tapes, rather than using different tape drives.

PROCEDURE A

Use this procedure for tapes residing **inside** the library.

Mounting —

Mount tapes using the Library Manager Console “Setup Stand-Alone Device” window.

1. Select the **Mount a single volume** option.
2. Type the device type.
3. Type the volume serial number.
4. Click **Enter**.
5. Repeat steps 1–3 for each tape that you want to mount.

Requirement: Mount the first dump data set tape prior to IPLing the Stand-Alone Services program when separate tape drives are used for the IPL tape and the dump data set tapes.

Demounting —

Demount and unload tapes using the Library Manager Console “Setup Stand-Alone Device” window or use the TAPECNTL command. (These instructions apply only to tapes that are not unloaded by Stand-Alone Services [for example, the IPL tape]).

1. Select the **Demount a single volume** option.
2. Type the device type.
3. Click **Enter**.

PROCEDURE B

Use this procedure for tapes residing **outside** the library.

Mounting —

Mount tapes using the Library Manager Console “Setup Stand-Alone Device” window.

1. Select the **Mount from Input Station** option.
2. Type the device type.
3. Type the volume serial number.
4. Click **Enter**.
5. Repeat steps 1–3 for each tape that you want to mount.

Requirement: Mount the first dump data set tape prior to IPLing the Stand-Alone Services program when separate tape drives are used for the IPL tape and the dump data set tapes.

Demounting —

When the IPL tape is the only tape that is mounted from the input station, unload and demount the tape after the Stand-Alone Services Restore operation has completed. Do this by canceling the mount from the input station using the Library Manager Console.

For additional information about tape library operations, refer to the appropriate IBM Tape Library operator’s guide.

Using an Automatic Cartridge Loader

A tape drive with an automatic cartridge loader can be set to manual mode or auto mode. When the loader is set to manual mode, you must manually remove the tape and mount subsequent tapes when Stand-Alone Services has unloaded the tape in the drive and is waiting for a subsequent tape to be mounted.

When the loader is set to auto mode, you can premount the tapes in the order that they will be needed. When Stand-Alone Services unloads the tape in the drive, the cartridges continue to feed and load without requiring intervention.

Controlling Command Sequence Processing

You can control Stand-Alone Services command processing by using SET and IF-THEN-ELSE command sequences. For more information about these commands, see “Controlling Task Processing” on page 231.

IPLing and Running the Stand-Alone Services Program

This section contains an overview of the Stand-Alone Services process, the procedure to IPL Stand-Alone Services, and specific information about the RESTORE and TAPECNTL commands.

Figure 5 on page 243 shows an overview of the Stand-Alone Services data restoration process.

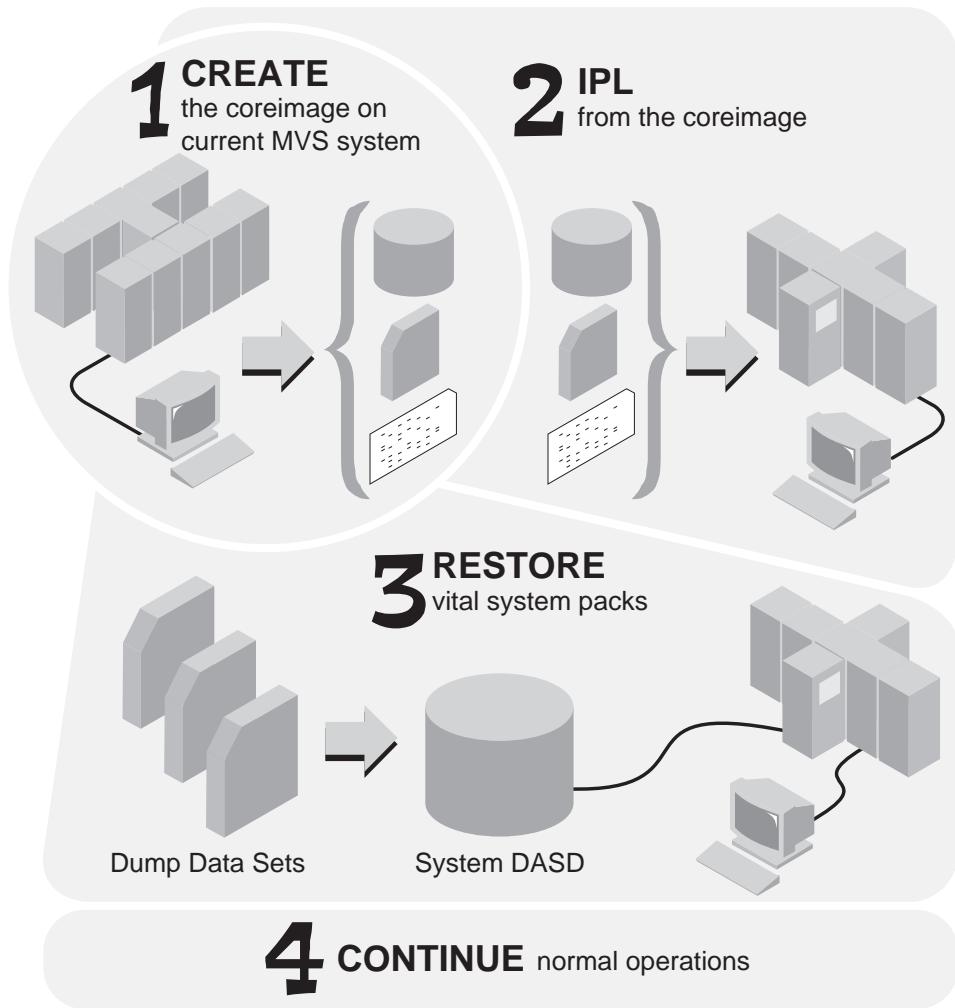


Figure 5. Stand-Alone Services Restore Process Overview

The following is an overview of the steps necessary to implement the Stand-Alone Services program:

1. **Prepare** for Stand-Alone Services by setting up the environment as described in Preparing to Run the Stand-Alone Services Program, and creating a Stand-Alone Services IPL-able core image with the BUILDSSA command. The BUILDSSA function is not part of Stand-Alone Services, yet it is necessary before Stand-Alone Services can be IPLed in a stand-alone environment. The BUILDSSA command is described in “BUILDSSA Command” on page 24.
2. **IPL** the Stand-Alone Services program from your specified tape, DASD, or card reader device.
3. **Restore** your dumped volumes with the RESTORE command. The RESTORE command performs a full-volume or tracks restore from a DFSMSdss-formatted or DFDSS-formatted dump tape.
Use the TAPECNTL command to rewind and unload a tape under Stand-Alone Services control rather than perform this function manually.

IPLing Stand-Alone Services

This section lists the steps to IPL the Stand-Alone Services program. The programming status word (PSW) wait-state codes (encountered during

Stand-Alone Services

Stand-Alone Services processing) are found in “Interpreting Wait-State Codes” on page 245. An example of the Stand-Alone Services system IPL is included on page 245.

To IPL from the Stand-Alone Services core image (created with the BUILDSSA command), proceed as follows:

1. Load the Stand-Alone Services Program.

Load Stand-Alone Services from the processor’s IPL console by specifying the IPL address for the device (card, tape, or DASD) that contains the IPL-able core image and then, by performing a Load Clear operation (also referred to as an IPL Clear).

2. Select the Operator Console.

When the Stand-Alone Services program has finished loading, one of the following conditions exist:

- If the Stand-Alone Services core image was created *without* specifying the OPERCNSL keyword, then the processor enters a wait-state with the rightmost bytes of the PSW containing “FFFFFF”. Press the Enter key on the operator console you are using. The first interrupt presented is expected to be a console and is treated as such.
- If the Stand-Alone Services core image was created *with* the OPERCNSL keyword specified, then Stand-Alone Services attempts to use the device address specified by OPERCNSL as the operator console, rather than waiting for the first interrupt. See “Running Stand-Alone Services with a Predefined Console” on page 239 for more information.

3. Specify the Input Device.

After the operator console is identified, the following message is displayed:

```
ADRY005E DEFINE INPUT DEVICE, REPLY 'dddd,ccuu' or 'CONSOLE'
```

To specify the operator console as the input device, enter CONSOLE or a null line. To specify a different device type, enter *dddd,ccuu*, where *dddd* is either the device type or card, and *ccuu* is the unit address. For example, to select a 3505 card reader at address 502, enter:

```
card,502
```

The input device can be one of the supported console devices or a card reader device.

4. Specify the Message Output Device.

After the input device is identified, the following message is displayed:

```
ADRY006E DEFINE OUTPUT DEVICE, REPLY 'dddd,ccuu' or 'CONSOLE'
```

To specify the operator console as the output device, enter CONSOLE or a null line. To specify a different device type, enter *dddd,ccuu*, where *dddd* is either the device type or prnt, and *ccuu* is the unit address. For example, to select a 3800 print subsystem at address 510, enter:

```
prnt,510
```

The output device can be a supported console device or a supported printer device.

Note: operator messages (such as ADRY003D) will be sent to the operator console, not the specified output device.

5. Specify the correct date and time, if needed.

Stand-Alone Services automatically picks up the time and date from the processor’s time-of-day (TOD) clock, usually in the Greenwich Mean Time (GMT) format. When the TOD clock is incorrect or is not set, the following message is displayed:

ADRY015E SUPPLY TODAY'S DATE, REPLY 'MM/DD/YY'

When you have entered the correct date in the format indicated, the following message is displayed:

ADRY016E SUPPLY TIME OF DAY, REPLY 'HH:MM:SS'

Enter the correct time in the format indicated. If you press Enter without specifying a date or time, the value is set to zero.

At this point the IPL is complete. You can now enter Stand-Alone Services commands from the specified input device. Multiple commands can be entered without requiring a re-IPL of the Stand-Alone Services program.

IPL Example

In the following example, the operator console is defined as both the input device and output device. System messages are highlighted in **bold** followed by the user's response. The example shows how more than one command can be entered without requiring a re-IPL.

ADRY005E DEFINE INPUT DEVICE, REPLY 'DDDD,CCUU' OR 'CONSOLE'
ENTER INPUT/COMMAND:

{The Enter key is pressed}

ADRY006E DEFINE OUTPUT DEVICE, REPLY 'DDDD,CCUU' OR 'CONSOLE'
ENTER INPUT/COMMAND:

{The Enter key is pressed}

SA/XA/ESA 5695-DF1 DFSMSdss STAND-ALONE V1.3.0
TIME: 16:36:23 06/07/95

ENTER INPUT/COMMAND:

restore frmdv(tape) frmadr(faf) toadr(f4a) vfy(tstb04)

RESTORE FRMDV(TAPE) FRMADR(FAF) TOADR(F4A) VFY(TSTB04)
ADRY0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
16:38:05 06/07/95

ENTER INPUT/COMMAND:

restore frmdv(tape) frmadr(faf) toadr(f4a) -

RESTORE FRMDV(TAPE) FRMADR(FAF) TOADR(F4A) -
ENTER INPUT/COMMAND:

vfy(tstb04)

VFY(TSTB04)
ADRY0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
16:39:48 06/07/95

ENTER INPUT/COMMAND:

Interpreting Wait-State Codes

The following programming status word (PSW) wait-state codes can occur during the Stand-Alone Services IPL:

Code	Explanation
------	-------------

Stand-Alone Services

000033	A program check occurred while Stand-Alone Services was being loaded. Contact your software service representative.
000044	The IPL device is not operational. Contact your hardware service representative. If the IPL device is a tape drive, try to IPL from another tape drive.
000055	An I/O error occurred on the IPL device or channel while Stand-Alone Services was being loaded. Contact your hardware service representative to correct the cause of the problem.
000066	The IPL loader is unable to determine if the entire Stand-Alone Services core image has been loaded. This could be due to a software or hardware error. Determine the cause of the problem and contact the appropriate service representative.
000077	The IPL loader is unable to locate the SYS1.ADR.SAIPLD.Vvolser data set on the IPL volume.
000088	The device type being used to IPL is not a supported DASD device for IPLing the Stand-Alone Services.
0000AE	An external interrupt occurred while Stand-Alone Services was being loaded. Contact your software service representative.
0000AF	A supervisor call instruction (SVC) interrupt occurred while Stand-Alone Services was being loaded. Contact your software service representative.
0000E2	A machine check has occurred. Contact your hardware service representative.
00BCBC	An attempt was made to IPL a processor that is in BC mode. IPL on a processor that supports EC mode. When you IPL from a VM user ID, set it to EC mode. This PSW is only loaded if you are IPLing from card or tape.
111111	Stand-Alone Services is waiting for an I/O interrupt. If the processor stops with this code loaded it may be necessary to re-IPL and rerun the command. Contact your hardware service representative if the problem persists.
888888	A temporary wait for a service-signal interrupt.
999999	A temporary wait for a service-signal interrupt.
BBBBBB	Stand-Alone Services is waiting for the operator to enter the input in response to a prompting message on the service console.
DDDDDD	Stand-Alone Services has detected an error related to the predefined console, and is waiting for the operator to identify another console to be used as the operator console. Possible reasons for the error with the predefined console and actions to take are listed in “Running Stand-Alone Services with a Predefined Console” on page 239.
EE4990	Stand-Alone Services cannot find a required module. Refer to message ADRY4990I for more information.
EEEEnn	The processor is in a wait-state. The error is indicated by <i>nn</i> as follows: nn Indicates: 13 An SVC interrupt has occurred. Run the SADMP* service

aid to dump the contents of real storage to tape, and contact your software service representative.

- 14** A program interrupt has occurred. Run the SADMP* service aid to dump the contents of real storage to tape, and contact your software service representative.
- 15** There is insufficient main storage. Stand-Alone Services requires 2MB of storage.
- 16** An I/O error has occurred.
- 17** Stand-Alone Services is unable to open a data set or access a device, possibly because the device type is not supported.
- 18** Stand-Alone Services cannot send an operator message because the console is either not defined or is unavailable.
- 19** An end-of-data routine is missing. Run the SADMP* service aid to dump the contents of real storage to tape, and contact your software service representative.
- 1A** The predefined console is not attached or is not operational. Possible reasons for this error are listed in "Running Stand-Alone Services with a Predefined Console" on page 239.
- 1B** Stand-Alone Services is unable to communicate with the predefined service console. This may be due to an error, or because the necessary features do not exist on the processor to communicate with the predefined console.

Note: Refer to your appropriate MVS Service Aids publication for information on creating a stand-alone dump with the AMDSADMP (SADMP) service aid.

EECC03	A condition code 3 (not operational) has been received when attempting to communicate with the service console. Contact your hardware service representative.
EECCCC	An error occurred while trying to communicate with the service console. This can result from a hardware or a software problem.
F1F1F1	Waiting for an I/O interrupt while Stand-Alone Services is being loaded.
FFFFFF	Stand-Alone Services is waiting for the operator to identify the operator console. Generate an interrupt from the console that is to be used as the operator console.

RESTORE—Restoring a Formatted Dump Tape

Use the RESTORE command to perform either a full-volume or a tracks restore from dump tapes produced by DFSMSdss or DFDSS without the use of a system environment.

The RESTORE command can restore from tape volumes created by a full-volume or tracks dump, or it can restore a track or tracks from the first logical volume of a physical data set dump. It cannot restore from tapes created by a DFSMSdss or DFDSS logical dump, from a DFSMSdss tracks dump using the CPVOLUME keyword, or from dump tapes produced by other utilities.

Stand-Alone Services

With the RESTORE command you specify both the source tape volume (containing the dump data set) and the DASD target volume. Use either IBM standard label or nonlabeled tapes as the source tape volumes (dump tapes) for the Stand-Alone Services restore operation. If the volume serial number of the DASD target volume is different from the volume serial number of the original DASD source volume, the restore operation changes the DASD target volume serial number to that of the DASD source volume.

When IPLing from tape, the data to be restored can be mounted on a tape drive other than the IPL tape drive. Alternatively, a single tape drive can be used to mount the tape to be IPled and the tape to be restored.

The device type of the DASD source volume used for the system dump must match the device type of the receiving volume used in a Stand-Alone Services restore operation. However, dump data from a smaller capacity (less cylinders) device can be restored to a larger capacity (more cylinders) device of the same device type.

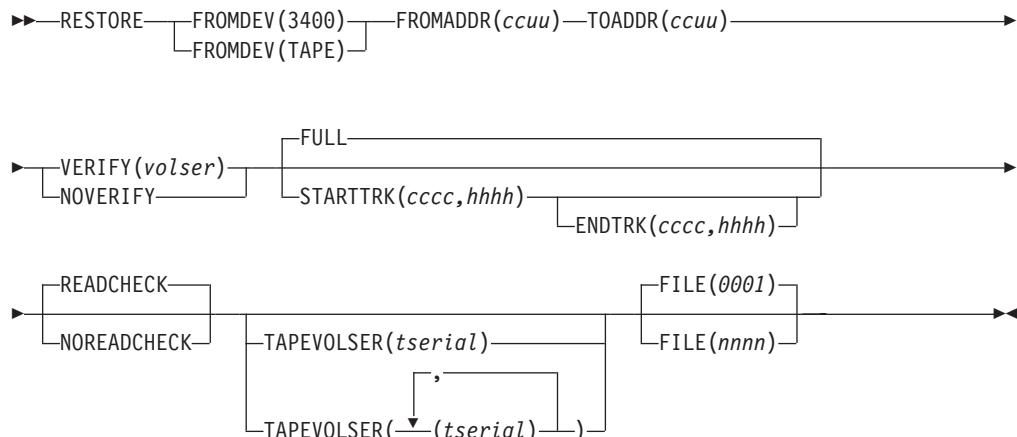
&
&
&
&
&
&

Notes:

1. Stand-Alone Restore does not allow restore of dump tapes created with Encryption or Hardware Assisted compression.
2. When data is restored from a smaller capacity device to a larger capacity device, the free space information becomes invalid. The free space information in the VTOC is rebuilt when the next data set is allocated on the volume.

RESTORE Command Syntax

The syntax of the Stand-Alone RESTORE command is:



See “Command Syntax” on page 1 for command and comment formatting specifics

Required Parameters

FROMDEV	Specifies the device type that the dump data set resides on. Eligible device names are 3400 and TAPE. When either 3400 or TAPE is specified, Stand-Alone Services attempts to determine the device type from the self-description information. If the device self-description information indicates a supported device type, Stand-Alone Services uses the returned device type for processing. When a device does not support self-description, Stand-Alone Services processes the device differently depending on whether 3400 or TAPE is specified. Abbreviations: FRMDEV and FRMDV.
----------------	---

3400	specifies device types 3420, 3422, and 3430. If the device type cannot be determined from the self-description information, Stand-Alone Services processes the device as a 3400-type device.
TAPE	specifies all other supported tape devices. If the device type cannot be determined from the self-description information, Stand-Alone Services processing ends.
FROMADDR	Specifies the address of the device that the dump data set resides on. You can specify a 3-digit or 4-digit address. Abbreviations: FRMADDR and FRMADR.
TOADDR	Specifies the address of the DASD target device to be restored. The device type must be the same as the device type of the volume originally dumped. You can specify a 3-digit or 4-digit address. Abbreviation: TOADR.
VERIFY (volser)	<p>Specifies that the volume serial number that is currently on the DASD target volume must be verified before restoring the data. Specify either VERIFY or NOVERIFY, not both.</p> <p>Initialize DASD volumes with a readable volume label and VTOC before restoration. Abbreviation: VFY.</p>
NOVERIFY	Specifies that no action be taken to either verify the volume serial number, or to verify that any volume serial number exists. When NOVERIFY is specified, a prompting message is issued for the user to reply with permission to continue. Specify either VERIFY or NOVERIFY, not both. Abbreviations: NOVFY and NVFY.

Optional Keywords

FULL	Specifies that the full volume be restored. The dump data set must be a DFSMSdss or DFDSS full-volume physical dump. The default is FULL.
STARTTRK	<p>Specifies that the designated range of tracks be restored. The dump data set can be a full-volume physical dump, a tracks dump, or a physical data set dump. Specify the STARTTRK keyword with a tracks dump or a physical data set dump. STARTTRK is not valid when FULL is specified.</p> <p>Specify the track in the form (cccc,bbbb), where cccc is the cylinder number and bbbb is the head number. You can specify the cylinder and head numbers in either decimal or hexadecimal (for example, X'AC',X'E' for hexadecimal or 172,14 for decimal). Leading zeros are not required. The STARTTRK keyword is processed as follows:</p> <ul style="list-style-type: none"> • When the ENDTRK keyword is not specified, the ending track is set to the last track on the volume. • When the starting track value is higher than the ending track value, an error message is issued and the restore operation ends. • When the starting track value exceeds the volume limits, an error message is issued and the restore operation ends.
ENDTRK	Abbreviations: STRTRK and STRK.

Specifies the ending track to be restored when STARTTRK is specified. This keyword is not valid when FULL is specified.

Stand-Alone Services

Specify the track in the form *(cccc,hhhh)*, where *cccc* is the cylinder number and *hhhh* is the head number. You can specify the cylinder and head numbers in either decimal or hexadecimal (for example, X'AC', X'E' for hexadecimal or 172,14 for decimal). Leading zeros are not required. The ENDTRK keyword is processed as follows:

- When the STARTTRK keyword is specified and the ENDTRK keyword is not specified, the ending track is set to the last track on the volume.
- When the ending cylinder value exceeds the volume limits, the ending cylinder value is set to the last cylinder on the volume, and a warning message is issued.
- When the ending head value exceeds the volume limits (exceeds the last head in a cylinder), the ending head value is set to the last head on the ending cylinder. In addition, a warning message is issued.
- When the starting track value is higher than the ending track value, an error message is issued and the restore operation is ended.

Abbreviation: ETRK.

READCHECK Specifies that a read-back check of the restored data be performed. READCHECK is the default. Abbreviations: READCHK, RDCHECK, RDCHK, and READ.

NOREADCHECK

Specifies that a read-back check of the restored data *not* be performed. Abbreviations: NOREADCHK, NREADCHK, and NREAD.

TAPEVOLSER (*tserial*)

Specifies the tape volume serial numbers of the tapes to be mounted by Stand-Alone Services when the tapes are in an IBM tape library. Volumes are mounted by Stand-Alone Services in the order in which they are specified.

The maximum number of tape volume serial numbers that can be specified is 32. All of the specified tape volumes must be part of the same dump data set that is used for the restore.

The TAPEVOLSER keyword is ignored if the FROMDEV keyword does not specify a valid tape drive in a tape library.

Do not specify the TAPEVOLSER keyword when the tapes are mounted from the Library Manager Console with the Setup Stand-Alone Device Pop-up Window. See “Using a Tape Library” on page 239 for information about tape libraries. Abbreviations: TAPEVOL and TPVOL.

FILE (*nnnn*)

Specifies the relative position, from the beginning of the tape volume, where the dump data set begins. Allowable values are from 1 to 9999. When the FILE keyword is not specified, the default value is 1. Leading zeros are not required. If the specified file number does not exist on the tape, unpredictable results can occur. For example, on 3400 tape devices with tape reels, an incorrectly specified file number can cause the tape to run off the end of the reel.

RESTORE Command Examples

In the following example, device address 2FAF is a 3490 tape drive with a tape mounted that contains the dump data set to be restored. Device address 4791 is a 3380 DASD with volume serial number D3380K. Before writing on the volume, the RESTORE command verifies that the DASD at address 4791 has volume serial number D3380K. The full volume is restored.

```
RESTORE FRMDV(TAPE) FRMADR(2FAF) TOADR(4791) VFY(D3380K)
```

In the following example, device address F01 is a 3420 tape drive with a tape mounted that contains the dump data set to be restored. Device address 9B9 is a 9345 DASD with volume serial number TS9345. Before writing on the volume, the RESTORE command verifies that the DASD at address 9B9 has volume serial number TS9345. The range of tracks to be restored is cylinder 0, head 0 through the end of the volume.

```
RESTORE FRMDV(3400) FRMADR(F01) TOADR(9B9) VFY(TS9345) STRK(0,0)
```

In the following example, device address F77 is a 3480 tape drive with a tape mounted that contains the dump data set to be restored on file 3 of the tape. Device address F4A is a 3390 DASD. The NOVERIFY keyword prompts the operator for permission to write on the device at address F4A.

```
RESTORE FRMDV(TAPE) FRMADR(F77) TOADR(F4A) NVFY FILE(3)
```

In the following two examples, device address F77 is a 3480 tape drive with a tape mounted that contains the dump data set to be restored. Device address 9B9 is a 9345 DASD with volume serial number TS9345. Before writing on the volume, the RESTORE command verifies that the DASD at address 9B9 has volume serial number TS9345. The range of tracks to be restored is from cylinder 200, head 5 through cylinder 205, head 14. The first example specifies the tracks in decimal:

```
RESTORE FRMDV(TAPE) FRMADR(F77) TOADR(9B9) VFY(TS9345) -
STRK(200,5) ETRK(205,14)
```

The second example specifies the tracks in hexadecimal:

```
RESTORE FRMDV(TAPE) FRMADR(F77) TOADR(9B9) VFY(TS9345) -
STRK(X'C8',X'5') ETRK(X'CD',X'E')
```

In the following example, device address FDD is a tape drive in a 3495 Tape Library, and the tape volume with volume serial number BCD103 contains the dump data set to be restored. Device address F4A is a 3390 DASD. The NOVERIFY keyword of the RESTORE command prompts the operator for permission to write on the device at address F4A. Stand-Alone Services mounts the tape volume with volume serial number BCD103 on the tape drive with address FDD. The range of tracks to be restored is cylinder 0, head 0 through cylinder 5, head 5.

Stand-Alone Services

RESTORE FRMDV(TAPE) FRMADR(FDD) TOADR(F4A) NVFY -
TAPEVOL(BCD103) STRK(0,0) ETRK(5,5)

In the following example, device address FDD is a tape drive in a 3495 Tape Library, and the tape volumes with volume serial numbers BCD101 and BCD102 contain the dump data set to be restored. Volume BCD101 is the first volume in the sequence, and BCD102 is the second volume. Device address 791 is a 3380 DASD. The NOVERIFY keyword of the RESTORE command prompts the operator for permission to write on the device at address 791. Stand-Alone Services mounts the tape volumes on the tape drive with address FDD. Volume BCD101 is mounted first, and when the end of the tape is reached volume BCD102 is mounted.

RESTORE FRMDV(TAPE) FRMADR(FDD) TOADR(791) NVFY -
TAPEVOL((BCD101) (BCD102))

TAPECNTL—Rewinding and Unloading a Tape

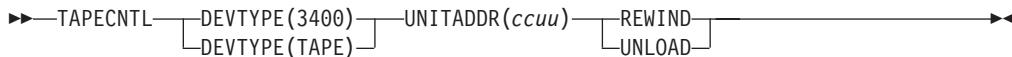
Use the TAPECNTL command to either rewind, or rewind and unload a tape.

Stand-Alone Services gives you the ability (with the REWIND and UNLOAD keywords) to rewind and unload tapes under Stand-Alone Services control rather than doing so manually.

The TAPECNTL command supports devices that are part of IBM 3494 and 3495 Tape Libraries.

TAPECNTL Command Syntax

The syntax of the TAPECNTL command is:



See “Command Syntax” on page 1 for command and comment formatting specifics.

Required Keywords

DEVTYPE	Specifies the tape device type that the TAPECNTL operation is to be performed against. Eligible device names are 3400 and TAPE. When either 3400 or TAPE is specified, Stand-Alone Services attempts to determine the device type from the self-description information. If the device self-description information indicates a supported device type, Stand-Alone Services uses the returned device type for processing. When a device does not support self-description, Stand-Alone Services processes the device differently depending on whether 3400 or TAPE is specified. Abbreviation: DEV.
----------------	---

3400 specifies device types 3420, 3422, and 3430. If the device does not support self-description, Stand-Alone Services may not be able to determine if the device is a tape drive. Stand-Alone Services issues the rewind or unload instruction to the device anyway. When the device is not a tape drive, unpredictable results may occur.

TAPE specifies all other supported tape devices. Stand-Alone Services does not issue rewind or unload instructions to the device when Stand-Alone Services cannot determine the device type from the self-description.

UNITADDR (ccuu)

Specifies the unit address of the device against which the TAPECNTL operation is performed. You can specify a 3-digit or 4-digit address. Abbreviations: UNIT and ADDR.

REWIND

Specifies a rewind operation. Specify either the REWIND keyword or the UNLOAD keyword, not both. Abbreviation: REW.

UNLOAD

Specifies that a rewind *and* unload operation be performed. If the tape drive is in an IBM tape library, the tape is demounted if necessary. Abbreviation: UNL.

TAPECNTL Command Examples

In the following example, the TAPECNTL command rewinds the tape mounted in the 3490 tape drive at address 2FAF:

```
TAPECNTL DEV(TAPE) UNIT(2FAF) REW
```

In the following example, the TAPECNTL command rewinds and unloads the tape mounted in the 3480 tape drive at address F77:

```
TAPECNTL DEV(TAPE) UNIT(F77) UNL
```

In the following example, the TAPECNTL command rewinds and unloads the tape mounted in the 3420 tape drive at address F01:

```
TAPECNTL DEV(3400) UNIT(F01) UNL
```

Building the IPL-able Core Image

This section describes how to use the BUILDSA command to build the Stand-Alone Services IPL-able core image. The core image is the executable module that is loaded into the processor's storage during the IPL and load process.

The BUILDSA Function

The BUILDSA function is not part of Stand-Alone Services, yet it is necessary before Stand-Alone Services can be IPLed in a stand-alone environment. The DFSMSdss BUILDSA command and examples are presented in "BUILDSA Command" on page 24.

Understanding BUILDSA Command Authorization Levels

Your ability to use the BUILDSA command is determined by your level of access authorization to source (input) data sets and target (output) data sets or volumes used by the BUILDSA operation. Storage administrators with any of the following access levels may use the BUILDSA command:

- Without special authorization
- DASDVOL-access authority to a volume

Stand-Alone Services

- DFDSS or DFSMSdss authorization

Using BUILDSA without Special Authorization

To use the BUILDSA command, you must have the following data-set-level authorization to build the IPL-able core image for card or tape:

- READ access to the input data sets used by the SYS1.SADRYLIB target library
- UPDATE access to the output card or tape data sets

Without special authorization, you cannot create an IPL-able core image for DASD. Even though you may have UPDATE or ALTER access to the output SYS1.ADR.SAIPLD.Vvolser data set, you are not authorized to update cylinder 0 head 0 of the DASD volume.

Using BUILDSA with DASDVOL-Access Authorization

To use the BUILDSA command for IPL(DASD), you must have the following access:

- READ access to the input data set (SYS1.SADRYLIB)
- UPDATE access at the DASDVOL level for each DASD volume on which you create an IPL-able core image

Using BUILDSA with the ADMINISTRATOR Keyword

Instead of having DASDVOL update access to volumes for IPL(DASD), you can act as a DFDSS- or DFSMSdss-authorized storage administrator for the BUILDSA command by specifying the ADMINISTRATOR keyword.

The FACILITY-class profile STGADMIN.ADR.STGADMIN.BUILDSA for the BUILDSA command ADMINISTRATOR keyword lets you build the Stand-Alone Services IPL-able core image without having UPDATE access to the output data sets or UPDATE access to a DASD volume when you create a core image for IPLing from DASD. You must still have access at the data set level for the input data set.

See the ADMINISTRATOR keyword in “Explanation of BUILDSA Command Keywords” on page 25 for more information.

Chapter 6. Data Security and Authorization Checking

This section describes the data security protection and access authorization checks done by DFSMSdss. The DFSMSdss functions available to a user depend on the access authorizations as defined by:

User and group profiles

Define the authorized users of a RACF-protected system. The user or group identifier (ID) used for access-authority checking must be defined to RACF.

Data set and general resource profiles

Protect the resources in a RACF-protected system and identify the access levels that users have to those resources.

DFSMSdss supports data-security protection and access-authorization checking through the system authorization facility (SAF), Resource Access Control Facility (RACF), and system services such as catalog management services and allocation. The phrase "RACF-protected" implies that SAF and RACF (or equivalent) are installed and active.

DFSMSdss uses the SAF interface and checks to ensure that RACF at the 1.8.1 level or later is installed and active. If an equivalent to RACF is used, either it must set the same level information that DFSMSdss checks, or your installation must use the DFSMSdss installation options exit to tell DFSMSdss that SAF with a RACF equivalent is installed. The proper level of RACF must be installed for the data-security features to work as described. The primary features and their levels of RACF are listed below:

Generic profile handling:	RACF 1.5 or later
DASDVOL-access authority:	RACF 1.6 or later
Facility class authority:	RACF 1.7 or later
Erase-on-scratch:	RACF 1.7 or later
Group data set creation:	RACF 1.8.1 or later
Storage class and management class:	RACF 1.8.1 or later

Related reading:

- For additional information about the installation options exit, see *z/OS DFSMS Installation Exits*.
- For additional information about data security and RACF, see *z/OS Security Server RACF Security Administrator's Guide*.

Effects of SPECIAL, OPERATIONS, and DASDVOL

The SPECIAL and OPERATIONS attributes and DASDVOL-access authority can affect the results when you use DFSMSdss to access and process data sets.

SPECIAL

When you use DFSMSdss, the SPECIAL attribute does not give you the authority to define or rename discrete profiles during a copy, move, or restore of user data sets that you do not own. This is because DFSMSdss uses the DEFINE function of SAF and RACF and the ALTER function of catalog management services to define and rename discrete profiles on your behalf. Instead of the system-SPECIAL attribute, you need the system-OPERATIONS attribute to define or rename discrete profiles for user data sets that you do not own.

OPERATIONS

If you have the system-OPERATIONS attribute, you have authority to the protected resources in resource classes such as DATASET, DASDVOL, and TAPEVOL. You are limited to the access specified in the access list if either:

- Your current connect group (or any connect group when list-of-groups checking is active) is in the access list of a resource profile.
- Your user ID is in the access list.

As a user with the system-OPERATIONS attribute, you have full control over data sets, and you can do the following:

- Copy, reorganize, catalog, and scratch (delete) data sets
- Perform input and output operations on protected tape volumes
- Define profiles for group data sets to which you are not connected
- Create user data sets and data sets in groups to which you are not connected. However, if your ID is in the access list, you need at least UPDATE access.

You need the system-OPERATIONS attribute to define or rename discrete profiles when you use DFSMSdss, even if you are acting as a DFSMSdss-authorized storage administrator (see “DFSMSdss Storage Administrator” on page 265). DFSMSdss can define or rename discrete profiles during copy or restore operations. However, the OPERATIONS attribute does not necessarily let you perform all DFSMSdss functions:

- If you have group-OPERATIONS, your authority is restricted to the resources within the scope of the group.
- The data set to be processed may be a data set over which you have no authority. For example, your user ID only has READ access to a data set that you want to copy and delete.
- A copy of a group data set may be disallowed because you are connected to the group and thus not authorized to define a discrete profile for the data set.
- Access can be denied due to security-level, security-category, or security-label checking.

DASDVOL

DASDVOL is supported directly for volume-level operations, physical operations for both SMS-managed and non-SMS-managed data sets, and logical operations for non-SMS-managed data sets. With DASDVOL-access authority to a volume, you can perform DFSMSdss operations as described in “Volume Access and DASDVOL” on page 268 and “DASDVOL Limitations” on page 269. When you do so, DFSMSdss bypasses access checking to the data sets and catalogs on that volume.

You have DASDVOL-access authority to one or more DASD volumes if all of the following are true:

- DASDVOL class is active.

- Profiles for the DASD volumes are defined in the DASDVOL class.
- You have the level of access needed for the function you are trying to perform.

Instead of DASDVOL-access authority, your installation can authorize you to act as a storage administrator for one or more DFSMSdss operations. See “DFSMSdss Storage Administrator” on page 265 for details.

Note: The system programmer or storage administrator uses ICKDSF to format tracks, write a volume label, and create a VTOC. The system programmer or storage administrator needs to have RACF DASDVOL authority to do that. No one needs DASDVOL authority to allocate space on volumes. The system controls space on SMS volumes by other means such as ACS routines, storage group definitions, and Interactive Storage Management Facility (ISMF) commands.

For information on customizing the ISMF, see Appendix F, “Customizing ISMF,” on page 363.

General Data Security Information

This section contains information on the following topics:

- “Protecting Resources and Data Sets”
- “Protecting Usage of DFSMSdss”
- “Password Protection” on page 258
- “Protected User and Group Data Sets” on page 259
- “Generic and Discrete Profile Considerations” on page 260
- “Security-Level, Category, and Label Checking” on page 262
- “Protect-All and Always-Call” on page 262
- “Standard Naming Conventions” on page 262
- “DFSMSdss Temporary Data Set Names” on page 262
- “Discretely Protected Multivolume Data Set” on page 264
- “Erase-on-Scratch” on page 264
- “SMS-Managed Data Set Protection” on page 264
- “Logging” on page 265

Protecting Resources and Data Sets

A security administrator or resource owner can control access to resources by creating a discrete profile, which protects one resource, or a generic profile, which protects one or more resources. Each profile has a defined, universal access level and an access list that permit user and group IDs specific levels of access authority. When profiles control resources, DFSMSdss usage can be restricted.

Protecting Usage of DFSMSdss

Your installation can put DFSMSdss in a protected library to restrict access and use. Your installation can also limit use of certain DFSMSdss commands and keywords by defining resource profiles in the FACILITY class and restricting access to those profiles. To use a protected command or keyword, you need READ-access authority to the applicable profile. Table 6 on page 258 provides an overview of the keywords and profiles. For information on the ADMINISTRATOR keyword, see “FACILITY-Class Profiles for the ADMINISTRATOR Keyword” on page 266.

Data Security

&
& *Table 6. DFSMSdss FACILITY-Class Profiles.* This table shows the DFSMSdss commands and keywords and the FACILITY-class profile that can restrict them.

Keyword	Profile Name
BYPASSACS with COPY	STGADMIN.ADR.COPY.BYPASSACS
BYPASSACS with RESTORE	STGADMIN.ADR.RESTORE.BYPASSACS
CGCREATE	STGADMIN.ADR.CGCREATE
CONCURRENT with COPY	STGADMIN.ADR.COPY.CNCURRNT
CONCURRENT with DUMP	STGADMIN.ADR.DUMP.CNCURRNT
CONVERTV	STGADMIN.ADR.CONVERTV
DEFrag	STGADMIN.ADR.DEFRAG
DELETECATALOGENTRY with RESTORE	STGADMIN.ADR.RESTORE.DELCATE
FCCGFREEZE with COPY	STGADMIN.ADR.COPY.FCFREEZE
FCTOPPRCPrimary with COPY	STGADMIN.ADR.COPY.FCTOPPRC
FCTOPPRCPrimary with DEFrag	STGADMIN.ADR.DEFRAG.FCTOPPRC
IMPORT with RESTORE	STGADMIN.ADR.RESTORE.IMPORT
INCAT(catname) with COPY	STGADMIN.ADR.COPY.INCAT
INCAT(catname) with DUMP	STGADMIN.ADR.DUMP.INCAT
INCAT(catname) with RELEASE	STGADMIN.ADR.RELEASE.INCAT
SET PATCH	STGADMIN.ADR.PATCH
PROCESS(SYS1) with COPY	STGADMIN.ADR.COPY.PROCESS.SYS
PROCESS(SYS1) with DUMP	STGADMIN.ADR.DUMP.PROCESS.SYS
PROCESS(SYS1) with RELEASE	STGADMIN.ADR.RELEASE.PROCESS.SYS
TOLERATE(ENQF) with COPY	STGADMIN.ADR.COPY.TOLERATE.ENQF
TOLERATE(ENQF) with DUMP	STGADMIN.ADR.DUMP.TOLERATE.ENQF
TOLERATE(ENQF) with RESTORE	STGADMIN.ADR.RESTORE.TOLERATE.ENQF

Related reading:

For additional information about RACF class profiles, see *z/OS Security Server RACF Security Administrator's Guide*.

Password Protection

DFSMSdss supports password checking at the data set level, but not the catalog level. If a data set is password-protected and RACF-protected, access to the data set is determined only through RACF-authorization checking (password checking is bypassed).

Data set passwords in the DFSMSdss input command do not appear in the SYSPRINT data set listing. However, if your job abnormally ends, those passwords may appear in any resultant dump. To prevent this, you can put the passwords in a data set that is referred to with a DD statement.

Protected User and Group Data Sets

DFSMSdss does authorization checks to ensure that you have the authority to access data sets. Your ability to access and protect a data set is affected by whether the data set is a user data set or a group data set.

User Data Set

The high-level qualifier of the name of a user data set is a RACF-defined user ID. As a RACF-defined user, you can protect your own data sets. However, when you use DFSMSdss to discretely protect a data set for another user, you need the system-OPERATIONS attribute.

Usually, you can use DFSMSdss to create new user data sets if you own them or have ALTER access to them through a data set profile or an entry in the global access checking table. You can also create a new user data set in any of the following situations:

- You are acting as a DFSMSdss-authorized storage administrator through the ADMINISTRATOR keyword.
- You have DASDVOL-access authority to the non-SMS-managed volume that the data set is being created on.
- You have the system-OPERATIONS attribute, and you have not been explicitly denied access to the data set.
- The system has *always-call*, the data set name is protected by a generic profile, and you do not have Automatic Data Set Protection (ADSP).
- The data set is not protected by a generic profile, and you do not have ADSP.

Related reading:

- For additional information about *always-call*, see “Protect-All and Always-Call” on page 262.
- For additional information about ADSP, “Automatic Data Set Protection (ADSP) Attribute” on page 261.

Group Data Set

The high-level qualifier of the name of a group data set is a RACF-defined group ID. As a RACF-defined user, you can RACF-protect a group data set under any of these conditions:

- You have JOIN, CONNECT or CREATE authority in the group.
- You have the group-SPECIAL attribute in the group that owns the user profile.
- You have the system-OPERATIONS attribute, and you are not connected to the group.

You can create new group data sets with the DFSMSdss COPY or RESTORE command in the following situations:

- You are acting as a DFSMSdss-authorized storage administrator through the ADMINISTRATOR keyword.
- You have DASDVOL-access authority to the non-SMS-managed volume that the data set is being created on.
- You have the OPERATIONS attribute, and you have not been explicitly denied access to the data set.
- The system has always-call, the data set name is protected by a generic profile, you have ALTER-access authority to the data set profile, and you do not have ADSP. Instead of ALTER access, you can have CREATE authority in the group and UPDATE access to the data set.

Data Security

- The data set is not protected by a generic profile, and you do not have ADSP.

Generic and Discrete Profile Considerations

A generic or a discrete data set profile can protect a data set. The type of profile affects DFSMSdss functions, especially copy, dump, and restore.

Generic Profiles

Generic profiles can be used to protect existing data sets without turning on the RACF-indicator flag in the data set VTOC entry for a non-VSAM data set or in the catalog entry for a VSAM data set. Generic profiles can be used to permit access to data sets, deny access to data sets, and control who can create data sets. A small number of generic profiles can be used to protect many data sets.

Generic profile checking for the DATASET class must be activated by the RACF SETROPTS GENERIC(DATASET) command. An entry in the global access checking table can let a user access a data set that the generic data set profile does not. See "Global Access Checking Table" for more information.

Discrete Profiles

A discrete profile should be used to protect a data set only if the data set has a unique security requirement from any other resource. You can protect data sets with discrete profiles when you use the DFSMSdss COPY or RESTORE command if you have the authority to define a discrete profile to protect that data set.

When a data set is protected with a discrete profile, an indicator is set in the VTOC entry for a non-VSAM data set, in the catalog entry for a VSAM data set, or in the tape volume profile for the tape volume that contains a tape data set. This condition is called RACF-indicated.

The following are special considerations that apply to discrete profiles:

- If the data set is scratched, RACF deletes the discrete profile.
- If you rename a data set to your high-level qualifier, the data set and the discrete profile are renamed, and the owner information of the profile is changed to your user ID.
- If you rename a data set to a high-level qualifier other than your own, the owner of the high-level qualifier becomes the owner of the data set.

Discrete and Generic Profile Checking

For RACF-indicated data sets, RACF searches first for a discrete profile, and then, if one is not found, for a generic profile. If neither is found, access is denied. If a data set is not RACF-indicated, the data set is RACF-protected only if there is a covering generic profile. The search is done in the following order:

1. Discrete profile if the data set is RACF-indicated.
2. Fully-qualified generic profile.
3. Other generic profiles from the most specific to the least specific profile name.

Global Access Checking Table

Your installation can use entries in the global access checking table to permit, but not deny, access to data sets. Only data set profiles can be used to deny access to data sets. Each entry in the table should have a corresponding generic profile to ensure consistent processing results.

Automatic Data Set Protection (ADSP) Attribute

If you have the ADSP attribute, RACF automatically defines a discrete profile whenever you create a permanent DASD or tape data set that is not already protected. For a tape data set, TAPEDSN and TAPEVOL must be active. If you have the ADSP attribute, you can create and protect data sets:

- Whose names begin with your user ID.
- Whose high-level qualifier belongs to a RACF group in which you have CREATE or higher authority. Besides CREATE in the group, you also need UPDATE access to the data set when you use DFSMSdss to copy or restore a data set.

Security-Level, Category, and Label Checking

DFSMSdss does not perform any explicit security-level, security-category, or security-label (SECLABEL) checking. For example, SECLABELs are not dumped or restored unless they are embedded within the data itself such as zFS data sets. However, DFSMSdss and the system services it uses (such as allocation) call RACF to make authorization checks that can result in access being denied because of a mismatch between the security level, security category, or security label of the resource and that of your user ID.

Protect-All and Always-Call

If protect-all has been activated, you can access a data set only through a data set profile or through an entry in the global access checking table. When always-call is in effect, RACF is called whenever a data set is accessed or DASD space is allocated. When RACF is called because of always-call (and not because of RACF-indication), it only checks generic profiles and the global access checking table. If protect-all is not in effect and RACF cannot find an appropriate data set profile or entry in the global access checking table, RACF accepts the request by default. Be aware of the following conditions:

- Always-call is in effect if the data sets are cataloged.
- Data sets that are not RACF-indicated, but which are protected by generic profiles and always-call, are not protected if they are transferred to another system that does not have RACF, always-call, and appropriate generic profiles.
- VSAM data sets are protected only by the RACF profile for the cluster name. Profiles for the index- and data-component names are ignored, as are the profiles associated with any PATHs or with the cluster's alternate indexes (AIXs).

Standard Naming Conventions

By default, RACF expects a data set name to consist of at least two qualifiers, with the high-level qualifier either a RACF-defined user or group ID. Single-qualifier data set names, especially for DASD data sets, affect your ability to manage or protect your data sets. For example:

- Data set name filtering is less usable for selecting data sets to be processed.
- DFSMSdss definitions of discrete data set profiles fail due to lack of correct prefix information.
- Your installation has trouble protecting the temporary data set names that DFSMSdss uses.

DFSMSdss Temporary Data Set Names

DFSMSdss must allocate temporary data sets to perform certain functions such as copy and restore. The high-level qualifiers of those data set names can be protected, and your installation must ensure that these temporary data sets can be allocated.

Message data set

Allocated by DFSMSdss to store messages. This data set lets DFSMSdss print out messages by task rather than intermixing them. This data set is deleted when DFSMSdss completes the operation. System-generated temporary names are used.

Special DEFrag data set

Allocated by DFSMSdss to contain information about the DASD extents that are being moved. The data set name is in the following format:

SYS1.DFDSS.DEFRAG.xxxxxxx.volser.DUMMY

where xxxxxxxx represents eight bytes of X'FF', and volser is the volume serial number of the volume being defragmented. The data set is deleted when the DEFrag operation completes successfully. If the DEFrag operation is interrupted (for example, when DFSMSdss is cancelled), this data set is left on the volume, and:

- You must run a new DEFrag operation.
- You may have to convert an index VTOC (IXFORMAT) volume to nonindexed (OSFORMAT) before rerunning the DEFrag operation. Otherwise, the volume free-space values may be incorrect.
- The hexadecimal qualifier is used to prevent you from deleting this data.

Temporary copied data sets

Allocated by DFSMSdss when a copy is performed and deleted when the copy is completed.

The format of the temporary name depends on the number of qualifiers of the data set that is being copied:

Number of qualifiers (n)	Temporary name
1	dsn1q.Atidasid.chmmsstt
2	First 2 qualifiers.Atidasid.chmmsstt
> 2	First 3 qualifiers.Atidasid.chmmsstt

The next to last qualifier Atidasid is: a combination of a fixed "A" character, a task id (tid), and an address space id (asid).

The last qualifier is *chmmsstt* where *c* is:

T	Target cluster name
D	Target data component name
I	Target index component name
U	Source cluster name
E	Source data component name
J	Source index component name
P	Source path name
Q	Target path name

and *hmmsstt* is the time stamp information in low-order hours digits (*h*), minutes (*mm*), seconds (*ss*), and hundredths of a second (*tt*).

Note: In the course of copying data sets, DFSMSdss renames the source data set using the above conventions. Whenever DFSMSdss renames a data set that is protected by RACF to a temporary name, a RACF profile must exist for the temporary data set name.

Temporary copied catalogs

Allocated by DFSMSdss when it copies a catalog. When DFSMSdss copies a catalog, two temporary data sets are used.

First, DFSMSdss allocates a temporary data set into which records are temporarily exported with the following name format:

CATHLQ.EXPORT.Thmmsstt
where,

CATHLQ First three high-level qualifiers of the catalog being copied

Data Security

hmmsstt Time-stamp information in low-order hours digits (*h*), minutes (*mm*), seconds (*ss*), and hundredths of a second (*tt*)

Second, DFSMSdss allocates a temporary catalog with the following name format:

CATHLQ.Thmmsstt
where,

CATHLQ First four high-level qualifiers of the catalog that is being copied

hmmsstt Time-stamp information in low-order hours digits (*h*), minutes (*mm*), seconds (*ss*), and hundredths of a second (*tt*)

Dummy data set

Allocated by DFSMSdss when copying or restoring volumes and an indexed VTOC needs to be rebuilt or the volume free-space values need to be recalculated. The data set name is in the following format:

SYS1.VTOCIX.DSS TEMP.volser

where volser is the volume serial number of the volume being restored. Allocation of this data set is never successful because DFSMSdss uses dummy allocation values.

Discretely Protected Multivolume Data Set

To create a discrete profile for a multivolume, non-VSAM, DASD data set, you must define each volume of the data set to RACF. When the data set is extended to another volume or deleted from a volume, that volume's serial number is automatically added to or deleted from the data set profile.

Erase-on-Scratch

When the erase indicator is set in a DASD data set profile, the tracks of any scratched or released data set extents that are part of the protected DASD data set are erased. Erase-on-scratch is supported for the following DFSMSdss commands:

- DUMP with DELETE
- COPY with DELETE
- DEFrag
- RELEASE

When DFSMSdss deletes a data set or moves data extents, the original tracks are erased if the erase indicator is set. This also applies during a copy or restore when preallocated data sets are deleted and reallocated.

SMS-Managed Data Set Protection

A data set profile may contain a DFP segment. The DFP segment contains a RESOWNER field, which may be used to specify the owner of an SMS-managed data set that is protected by the data set profile. If a new SMS-managed data set is allocated, the user or group ID in the RESOWNER field must have at least READ access to any MGMTCLAS and STORCLAS profile used in the allocation when:

- Your installation has activated the RACF general resource classes MGMTCLAS and STORCLAS.
- Profiles have been defined in those classes to protect against unauthorized usage of an SMS management class name or a storage class name.

If RESOWNER is not specified when the data set is allocated, the user or group ID that matches the high-level qualifier is used as the data set owner. Regardless of who owns an SMS-managed data set, you can select different default values for MGMTCLAS and STORCLAS by using those parameters with the COPY or RESTORE command. When you do, RACF checks to ensure that the data set owner, rather than you, is authorized to use the specified MGMTCLAS and STORCLAS. The user or group ID must not be revoked.

Storage class and Management class authorization checking can be bypassed for logical restore and physical data set restore processing by using the ADRPATCH Serviceability Aid in conjunction with the ADMINISTRATOR keyword.

Related reading: For additional information about using the ADRPATCH Serviceability Aid, see *z/OS DFSMSdss Storage Administration Guide*.

Logging

DFSMSdss automatically supports RACF logging as follows:

- Logging of DASDVOL-access checks is allowed for successful authorizations only. Unsuccessful attempts are not logged because the user can still gain access at the data set catalog level.
- Logging of access checking for data sets occurs as specified in the applicable data set profile. However, because catalog management services are used, DFSMSdss does not fully control logging for VSAM data sets.
- Logging of access to the FACILITY-class profiles for DFSMSdss is allowed as defined for those profiles.
- DFSMSdss does not control logging of RACF DEFINE requests. DEFINE requests are made to define discrete profiles and to determine if a user has CREATE authority in a group.
- Logging of access-level checks for a catalog occurs only once.
- Logging cannot be disabled by specifying RACFLOG=NO in the PARM statement of a DFSMSdss job. The only way to turn off logging is through the DFSMSdss installation options exit.

DFSMSdss Storage Administrator

Your installation can use DASDVOL-access authority to designate users who can act as DFSMSdss storage administrators. DASDVOL-access authority lets you perform any DFSMSdss function against the data sets on that volume. However, DASDVOL-access is not supported for logical operations against SMS-managed data sets. Further, DASDVOL-access authority must be granted in every volume for all the data sets for which you are the storage administrator.

DFSMSdss provides an alternative to DASDVOL-access authority. Your installation can define special FACILITY-class profiles to let you act as a DFSMSdss-authorized storage administrator.

ADMINISTRATOR Keyword

To act as a DFSMSdss-authorized storage administrator, specify the ADMINISTRATOR keyword on the appropriate DFSMSdss command. If you are not authorized to use the ADMINISTRATOR keyword, the command is ended with an error message. Otherwise, access checking to data sets and catalogs that are initiated by DFSMSdss are bypassed. Please note, certain authorization checking to RACF for data sets are unavoidable, regardless of whether the ADMINISTRATOR

Data Security

keyword is specified. For example, when it is necessary to determine if the data set has a defined RACF profile or when authorization checks are initiated by callable services or utilities that DFSMSdss invokes such as Catalog and IEBCOPY, cannot be bypassed.

To use the ADMINISTRATOR keyword, all of the following must be true:

- FACILITY class is active.
- Applicable FACILITY-class profile is defined (see below).
- You have READ access to that profile.

As a DFSMSdss-authorized storage administrator, your authority is for both physical and logical operations. For example, a DFSMSdss-authorized storage administrator for the COPY command can copy tracks, a full volume, or data set without having access to the individual data sets or their catalogs.

DFSMSdss tries to define a discrete profile when you copy a discretely-protected data set or when you use the MENTITY keyword. The authority to do so is not given to you by the DFSMSdss ADMINISTRATOR support. For example, to define or rename a discrete profile for a user data set that you do not own, you also need the system-OPERATIONS attribute. The ADMINISTRATOR support also does not give you authority to bypass checking for MGMTCLAS and STORCLAS authority.

Storage class and Management class authorization checking can be bypassed for logical restore and physical data set restore processing by using the ADRPATCH Serviceability Aid in conjunction with the ADMINISTRATOR keyword.

Related reading: For additional information about using the ADRPATCH Serviceability Aid, see *z/OS DFSMSdss Storage Administration Guide*.

FACILITY-Class Profiles for the ADMINISTRATOR Keyword

The following are the names and descriptions of the FACILITY-class profiles for the ADMINISTRATOR keyword.

STGADMIN.ADR.STGADMIN.COMPRESS

Lets you compress data sets without having UPDATE access authority to those data sets.

STGADMIN.ADR.STGADMIN.COPY

Lets you copy data sets without having access authority to the source (READ) or target (UPDATE or ALTER) data sets and their catalogs. This profile does not give you the authority to rename a data set.

STGADMIN.ADR.STGADMIN.COPY.DELETE

Lets you copy or copy and delete data sets without having access authority to the source (ALTER) or target (UPDATE or ALTER) data sets and their catalogs. This profile does not give you the authority to rename a data set.

STGADMIN.ADR.STGADMIN.COPY.RENAME

Lets you copy or copy and rename data sets, through the RENAMEUNCONDITIONAL keyword, without having access authority to the source (READ) or the target (ALTER) data sets and their catalogs. This profile does not give you the authority to delete a data set.

STGADMIN.ADR.STGADMIN.DEFRAG

Lets you perform a DEFrag operation without having READ access to the data sets that are moved. RACF can still be called for DFSMSdss to determine if erase-on-scratch processing for a given data set must be done.

STGADMIN.ADR.STGADMIN.DUMP

Lets you dump data sets without having READ access to the data sets. This profile does not give you the authority to delete a data set.

STGADMIN.ADR.STGADMIN.DUMP.DELETE

Lets you dump or dump and delete data sets without having ALTER access to the data sets and their catalogs.

STGADMIN.ADR.STGADMIN.PRINT

Lets you print data without having READ access to the data sets containing the data to be printed.

STGADMIN.ADR.STGADMIN.RELEASE

Lets you release unused space without having UPDATE access to the data sets.

STGADMIN.ADR.STGADMIN.RESTORE

Lets you restore data without having READ, UPDATE, or ALTER access to source and target data sets and their catalogs. This profile does not give you the authority to rename a data set.

STGADMIN.ADR.STGADMIN.RESTORE.RENAME

Lets you restore or restore and rename data sets, through the RENAME or RENAMEUNCONDITIONAL keyword, without having READ, UPDATE, OR ALTER access to source and target data sets and their catalogs.

DFSMSdss Volume, Data Set and Catalog Access Authority

DFSMSdss checks your access authority to the volume before it performs any data set access-authorization checking. You have the required volume-level authority whenever any of the following is true:

- NOPASS was specified on the PPT statement of the SCHEDxx parmlib member for the ADRDSSU or user program which calls ADRDSSU. ADRDSSU (or its caller) is coded in the PGM parameter of the EXEC statement and should be authorized.
- Bypass authorization checking is requested by the DFSMSdss installation authorization exit. This exit cannot be used to bypass authorization checking initiated by utilities such as IEBCOPY that are invoked by DFSMSdss.
- You have the required DASDVOL-access authority for the function you are trying to perform.
- You are acting as a DFSMSdss-authorized storage administrator.

For SMS-managed data sets, access authorization to create, update, or delete data sets also authorizes you to create, update, or delete entries in user catalogs. However, to add or delete any entry for an SMS-managed data set in a protected master catalog, you also need UPDATE access to the master catalog. To add or delete any non-SMS entry in any protected catalog, you need UPDATE access to the catalog.

Figure 6 on page 268 shows the major decisions that are made to determine if you are authorized to perform a function against a data resource. The figure does not represent the actual program flow in DFSMSdss. Use the diagram to clarify the data security decisions under varying conditions.

Related reading: For additional information about when DFSMSdss invokes utilities, see the *z/OS DFSMSdss Storage Administration Guide*.

Volume Access and DASDVOL

You can use DFSMSdss to perform volume-level operations such as a full-volume dump. Instead of requiring you to have sufficient access authority to each data set on the volume, DFSMSdss supports DASDVOL-access authority.

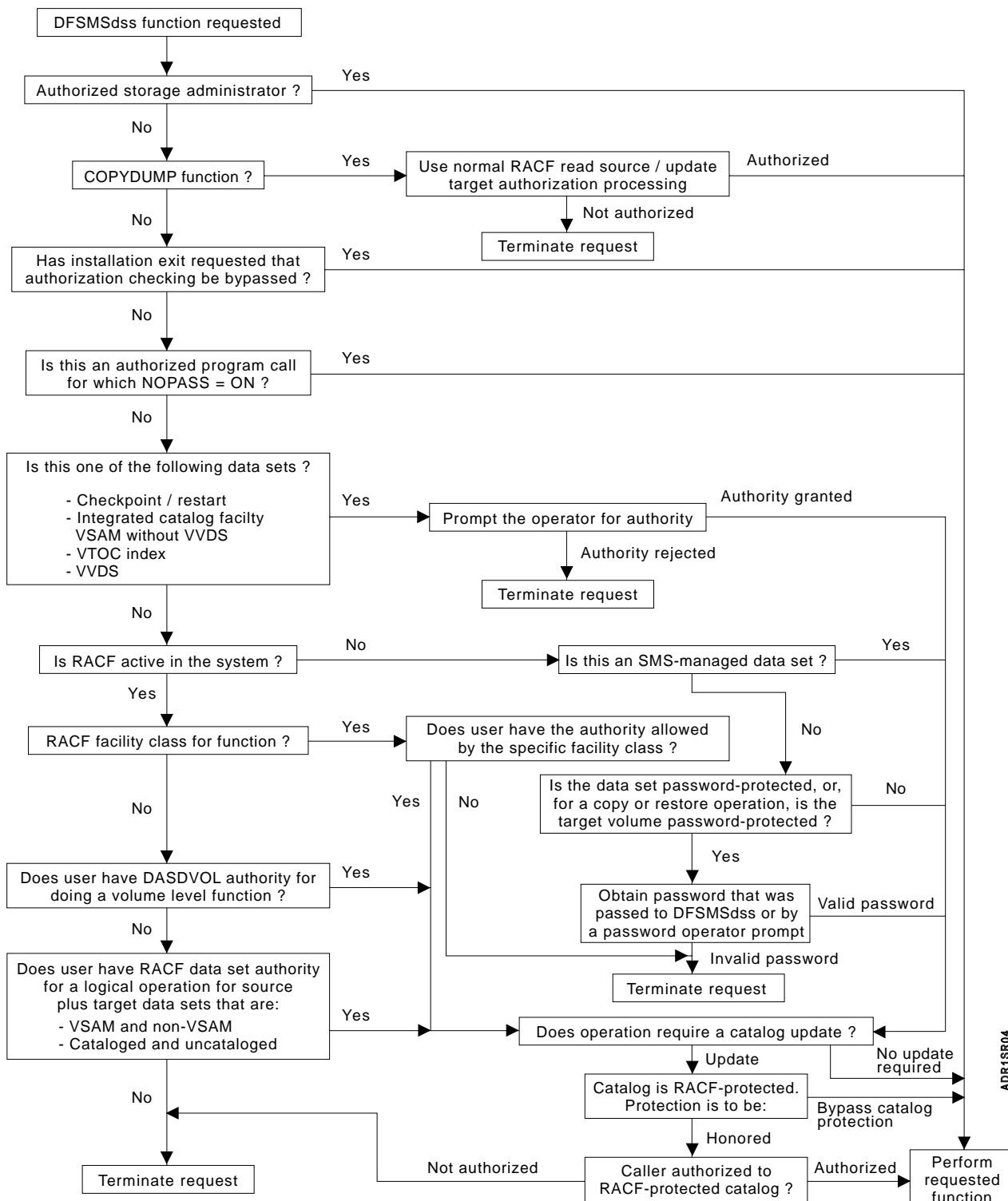


Figure 6. DFSMSdss Data Security Decisions

DASDVOL Access Authority

When you have the required level of DASDVOL authority to a volume, as shown in Table 7, you can perform maintenance operations such as dump, restore, scratch and rename of the data sets on that volume regardless of your access authority to those data sets. DFSMSdss lets you catalog, recatalog, and uncatalog the data sets on the volumes to which you have DASDVOL authority. You also have access to the catalogs on the volumes to which you have DASDVOL-access authority.

When you restore a data set, DASDVOL-access checking is determined as follows:

- The volume serial number of the source volume from which the data set was originally dumped, regardless of the current status of that data set or volume.
- The volume serial number of the volume on which the data set is being restored. This volume (or volumes) may not be the same as the source volume.

Table 7. DASDVOL Access Authority. This table shows the DASDVOL-access authorities needed to perform volume and data-set-level functions.

Function		DASDVOL Access Needed		
Command	Volume	FULL	TRACKS	Data Set
COMPRESS	Source	N/A	N/A	UPDATE
CONVERT	N/A	N/A	N/A	N/A
COPY	Source	READ	READ	READ
	Target	ALTER	ALTER	UPDATE
COPY with DELETE	Source	N/A	N/A	ALTER
	Target	N/A	N/A	UPDATE
COPYDUMP	N/A	N/A	N/A	N/A
DEFrag	Source	N/A	N/A	UPDATE
DUMP	Source	READ	READ	READ
DUMP with DELETE	Source	N/A	N/A	ALTER
	Target	N/A	N/A	ALTER
PRINT	Source	N/A	READ	READ
RELEASE	Source	N/A	N/A	UPDATE
RESTORE	Source	N/A	N/A	READ
	Target	ALTER	ALTER	UPDATE

DASDVOL Limitations

DASDVOL-access authority is not supported for logical operations on SMS-managed data sets, nor does it let you define discrete profiles. For DFSMSdss to define or rename discrete profiles, you need the authority to do so.

While DASDVOL-access authority to a non-SMS-managed volume lets you move, copy, dump, or delete a catalog, you do not automatically have access to the data sets that are cataloged in it. Also, for certain operations (such as CONVERTV on an SMS-managed data set), an authorization check is performed to see if the data set owner is authorized to use the applicable MGMTCLAS and STORCLAS routines. This is true even if you have DASDVOL-access authorization.

Related reading:

Data Security

- For additional information about discrete profiles, see “Copy and Data Set Profile Considerations” on page 275.
- For additional information about discrete profiles, see “Restore and Data Set Profile Considerations” on page 281.

Data Set Access Authorization Levels

Access to a data set is controlled by a data set profile which permits or denies access. Access-authorization checking to data sets is performed if you do not have the required volume-level authority, the volume is unprotected, or DFSMSdss is unable to determine the protection status of the volume.

If the data set and its catalog are unprotected, no authorization is required to read, update, delete, rename, move, or scratch the data set. For protected data sets, the following access authorization may be required to perform DFSMSdss functions:

- Password specification is required. If the data set is password-protected and not RACF-protected, you must specify the appropriate password to read, update, delete, rename, move, or scratch the data set.
- The data set is RACF-protected. Your ability to perform operations is determined by your access-authorization level:

NONE denies you access to the data set.

READ lets you read the data set, and even make a copy of it, provided you have authority to create or replace the target data set.

UPDATE lets you modify (update) an existing RACF-protected data set. UPDATE access does not let you delete, rename, move, or scratch a data set.

ALTER lets you read, update, delete, rename, move, or scratch a RACF-protected data set.

ALTER access lets you create a user or group data set. CREATE authority in the group and UPDATE access also lets you create a new group data set.

The level of access authority you need to access data sets is also dependent on the following:

- Whether the data set is SMS-managed or not.
- Whether a catalog entry for the data set is added (cataloged) or deleted (uncataloged).
- Whether the catalog is an integrated catalog facility catalog.
- Whether the catalog is RACF-protected.

DFSMSdss supports VSAM data sets only if they are cataloged in an integrated catalog facility catalog. For VSAM data sets, access authorization is determined solely by your authority to access the base-cluster name: Data set profiles are ignored for any data component, index component, alternate index (processed as part of a sphere) or PATH.

Protected Catalogs

DFSMSdss only supports RACF-protection of catalogs; catalog passwords are never checked.

Catalog Access Authority

DFSMSdss supports cataloged non-VSAM and VSAM data sets in an integrated catalog facility catalog.

Related reading: For additional information about the access authority you need, see “Access Authorization for DFSMSdss Commands.”

Non-SMS Versus SMS Authorization

Access authorization requirements differ for SMS-managed data sets and non-SMS-managed data sets.

SMS-managed data sets:

Access to SMS-managed data sets gives you access to the user catalog for the data sets. However, for a RACF-protected master catalog, you also need UPDATE-access authority to add or delete an SMS-managed data set entry. DASDVOL-access authority is not supported for logical operations.

Non-SMS-managed data sets:

Besides access authority for data sets, you need UPDATE access to RACF-protected user or master catalogs to add or delete an entry in the catalog.

System Operator Authorization, Special Data Set Types

Some system and VSAM data sets accept authorization from the system operator when RACF or password checking cannot be performed. Unless you are using DFSMSdss with special authorization, such as DASDVOL or the ADMINISTRATOR keyword, system-operator authorization is required in order for you to update the following:

- Volume table of contents (VTOC)
- VTOC index data set
- VSAM volume data set (VVDS)
- Checkpoint/restart data set.

The system operator is only prompted for the first data set encountered in one of the above classes for the DFSMSdss command that is being processed. The reply given is used for that command invocation for all other data sets of that type on the volume.

Access Authorization for DFSMSdss Commands

&
&
&
&
&
&
&
&
&
&

This section covers the following commands:

- CGCREATE
- COMPRESS
- CONVERTV
- COPY
- COPYDUMP
- DEFrag
- DUMP
- PRINT
- RELEASE
- RESTORE

The tables in this section can help you determine if you have sufficient access authority to use the DFSMSdss commands. The access authority you need depends on what you are trying to do. For example:

Data Security

- If the data set is SMS-managed and you have access to the data set, you have access to its user catalog.
- If you have ALTER access to the data set, you can copy and delete (or dump and delete) the data set.
- If the source data set is cataloged, you may need UPDATE or ALTER access to its catalog if the data set is going to be deleted, uncataloged, or cataloged.
- To replace an existing data set using the COPY or RESTORE command, you need UPDATE access to the data set.
- If you are doing a restore with rename, ALTER-access authority lets you create a new user or group data set. If the data set belongs to a group, CREATE authority in the group and UPDATE access to that group data set name also lets you create that group data set.
- To restore a data set, you need READ access to a data set with the same name as the one that was dumped. You also need either UPDATE or ALTER access to the target data set.
- If the target data set is going to be discretely protected, and you do not own the data set, you need additional authority to define a discrete profile. This is true even if you are acting as a DFSMSdss-authorized storage administrator for the COPY or RESTORE command.
- To copy or dump a master catalog, you need ALTER access if you are not acting as a DFSMSdss storage administrator. This level of access is needed because the master catalog may contain passwords.
- If you are acting as a DFSMSdss storage administrator, you have access to the applicable data sets and catalogs.

&
&
&
&
&
&

CGCREATE

Anyone can use the CGCREATE command. However, your installation can limit usage of the CGCREATE command by use of a RACF FACILITY-class profile if:

- RACF FACILITY class is active.
- The FACILITY-class profile STGADMIN.ADR.CGCREATE has been defined. Then need READ access authorization to that profile to use the CGCREATE command.

COMPRESS

To compress partitioned data sets on a specified volume, either you need DASDVOL-access authority to the volume or UPDATE access authority to the data sets, or you must be acting as a DFSMSdss-authorized storage administrator.

CONVERTV

Anyone can use the CONVERTV command. However, your installation can limit usage of the CONVERTV command by use of a RACF FACILITY-class profile if:

- RACF FACILITY class is active.
- The FACILITY-class profile STGADMIN.ADR.CONVERTV has been defined. Then need READ access authorization to that profile to use the CONVERTV command.

To convert a volume to SMS-managed, the user and group IDs of the owners of all the data sets on that volume must be RACF-defined, and they must not be revoked.

COPY

Table 8 on page 274, which shows the access authority needed to perform the copy function, is based on the following assumptions:

- The data set is a user data set.
- Both the data set and the catalog are RACF-protected.
- If the data set is going to be cataloged in the master catalog, then you need UPDATE access to that catalog.

Table 8 on page 274 also applies to group data sets. To create a new group data set, you need one of the following:

- UPDATE access to the catalog when a non-SMS-managed group data set is unprotected, but the catalog is RACF-protected.
- CREATE in the group and UPDATE access to the data set.
- ALTER access to the data set and be a member of the group that owns the data set. To discretely protect a group data set, you need additional authority such as CREATE in the group.

Data Security

Table 8. COPY Command Access Authority. This table shows the minimum access levels required to perform a COPY command.

SOURCE DATA SET			TARGET DATA SET		
Non-SMS			Non-SMS		
With DELETE	Access Level		Preallocated Target	Access Level	
	Data Set	Catalog		Data Set	Catalog
NO	READ	Automatic	NO	ALTER	UPDATE
NO	READ	Automatic	YES	UPDATE	Automatic
YES	ALTER	Automatic	NO	ALTER	UPDATE
YES	ALTER	Automatic	YES	UPDATE	Automatic
Non-SMS			SMS		
With DELETE	Access Level		Preallocated Target	Access Level	
	Data Set	Catalog		Data Set	Catalog
NO	READ	Automatic	NO	ALTER	Automatic
NO	READ	Automatic	YES	UPDATE	Automatic
YES	ALTER	Automatic	NO	ALTER	Automatic
YES	ALTER	Automatic	YES	UPDATE	Automatic
SMS			Non-SMS		
With DELETE	Access Level		Preallocated Target	Access Level	
	Data Set	Catalog		Data Set	Catalog
NO	READ	Automatic	NO	ALTER	UPDATE
NO	READ	Automatic	YES	UPDATE	Automatic
YES	ALTER	Automatic	NO	ALTER	UPDATE
YES	ALTER	Automatic	YES	UPDATE	Automatic
SMS			SMS		
With DELETE	Access Level		Preallocated Target	Access Level	
	Data Set	Catalog		Data Set	Catalog
NO	READ	Automatic	NO	ALTER	Automatic
NO	READ	Automatic	YES	UPDATE	Automatic
YES	ALTER	Automatic	NO	ALTER	Automatic
YES	ALTER	Automatic	YES	UPDATE	Automatic

Notes:

- “Automatic” means that you have automatic access to the user catalog if you can access the data sets.
- In the course of copying data sets, DFSMSdss will rename the source data set using the conventions described under DFSMSdss Temporary Data Set Names. (See 263 under Temporary Copied Data Sets.) Whenever DFSMSdss renames a RACF protected data set to a temporary name, a RACF profile must exist for the temporary data set name.

COPY and Deleting Data Sets

For SMS-managed data sets, ALTER access lets you delete the data set. For non-SMS-managed data sets, either ALTER access to the data set or ALTER access to the catalog and READ access to the data set lets you copy and delete the data set. The ability to delete and uncatalog a data set with READ access to the data set

and ALTER access to the catalog lets you move a data set without having ALTER access to the data set. You only need UPDATE access to the catalog to delete an unprotected, non-SMS-managed data set.

Copy and Data Set Profile Considerations

When you copy a RACF-indicated data set, DFSMSdss performs special target data set processing. A predefined, discrete profile is not used to check your access authority unless the data set is already RACF-indicated.

Copy and Data Set Profiles: If a data set is protected by a generic or discrete profile when it is copied, DFSMSdss tries to ensure that the target data is also protected (see Table 9). To do so, DFSMSdss uses the RACF DEFINE function or the catalog ALTER function. You need authority to define or rename discrete profiles, even if you are acting as a DFSMSdss-authorized storage administrator.

Table 9. COPY Command and RACF Profiles. This table summarizes how DFSMSdss defines discrete profiles to protect the target set.

COPY with:		Source Data Set Protected?	RACF Profile:	
RENAME	DELETE		Source	Target
NO	NO	NO	None	As predefined
NO	NO	GENERIC	No change	As predefined
NO	NO	DISCRETE	No change	DEFINE
NO	YES	NO	None	As predefined
NO	YES	GENERIC	No change	As predefined
NO	YES	DISCRETE	DELETE	DEFINE
YES	NO	NO	None	As predefined
YES	NO	GENERIC	No change	MENTITY
YES	NO	DISCRETE	No change	DEFINE
YES	YES	NO	None	As predefined
YES	YES	GENERIC	No change	MENTITY
YES	YES	DISCRETE	DELETE	DEFINE

Terms defined:	
None	The data set remains unprotected.
As predefined	DFSMSdss does not ensure that the target data set is RACF-protected. If a covering data set profile is not already predefined, the data set may be unprotected.
No change	The data set profile that is protecting the source data set remains unchanged.
DEFINE	A discrete profile is defined by DFSMSdss.
DELETE	The data set and the discrete profile are deleted.
MENTITY	If the target data set is not RACF-protected by a generic or discrete profile and MENTITY is specified, DFSMSdss defines a discrete profile. Otherwise, no profile is defined.

COPY and the MENTITY Keyword: By default, when DFSMSdss defines a discrete profile to protect a copied data set, it uses the discrete profile of the source data set as a model. The MENTITY keyword lets you specify a different data set profile as the model. Besides the model profile, you can also specify a volume serial (through MVOLSER). If you do so, specify the volume serial number of the volume containing the non-VSAM model entity or the volume containing the catalog in which the VSAM model entity is cataloged.

Data Security

Use the MENTITY keyword of the COPY command when one of the following is true:

- The source data set is RACF-protected by a generic profile, but the target data set name is unprotected.
- The source data set is RACF-protected by a discrete profile, but you wish to use a different data set profile as the model for defining a new discrete profile.

COPY and Define Discrete Profile Summary: Table 10 summarizes the circumstances during copy operation when a discrete profile is defined, when MENTITY is effective, when the target data set is RACF-indicated, and when related messages are issued. To define discrete profiles for data sets that you do not own, you need additional authorization such as the system-OPERATIONS attribute or CREATE in the group.

Table 10. Copy and Define Discrete Profile Summary. This table summarizes what happens when DFSMSdss defines discrete profiles for a copy operation.

Protection		MENTITY Specified?	Define Profile?	RACF Indicated?	Warning Message?	
Source	Target					
None	None	Yes	No	No	No	
		No	No	No	No	
None	Generic	Yes	No	No	No	
		No	No	No	No	
None	RACF Indicated	Yes	No	Yes	No	
		No	No	Yes	No	
Generic	None	Yes	Yes	Yes	No (W)	
		No	No	No	Yes	
Generic	Generic	Yes	No	No	No	
		No	No	No	No	
Generic	RACF Indicated	Yes	No	Yes	No	
		No	No	Yes	No	
RACF Indicated	None	Yes	Yes	Yes	No (W)	
		No	Yes (C)	Yes	No (E)	
RACF Indicated	Generic	Yes	Yes	Yes	No (W)	
		No	Yes (C)	Yes	No (E)	
RACF Indicated	RACF Indicated	Yes	No	Yes	No	
		No	No	Yes	No	
Terms defined:						
Yes (C) If there is a discrete profile for the source data set, a discrete profile is defined for the target data set.						
No (W) If the define of the discrete profile fails and the copy is successful, then the target data set is left as RACF-indicated and a warning message is issued. This is true even if DELETE is specified.						
No (E) If the define of the discrete profile fails and DELETE is specified, then the copy is unsuccessful and an error message is issued.						
If the define of the discrete profile fails and DELETE is not specified, then the target data set is left as RACF-indicated and a warning message is issued.						

Protection States after a Copy or Move: When a data set is copied or moved, the protection state of the target depends on the initial protection states of the source and the target data sets, whether MENTITY was specified, and whether discrete profiles are predefined. The following conditions may exist:

The source data set is unprotected.

The target data set may or may not be protected after the copy. A discrete profile is not defined by DFSMSdss, even if you specify MENTITY.

The source data set is protected but not RACF-indicated.

The target data set may or may not be protected after the copy:

- If the target data set is not already protected and MENTITY is not specified, the data set remains unprotected. If you do specify MENTITY, DFSMSdss defines a discrete profile:
 - If the define is successful, the target data set is RACF-indicated. It is accessible according to the access list for the model profile that was used.
 - If the define is unsuccessful, the target data set remains RACF-indicated. It may be inaccessible until a data set profile is successfully defined.
- If the target data set is already protected because it is RACF-indicated or RACF-protected, DFSMSdss does not define a discrete profile (even if you specify MENTITY). The target data set is accessible according to the access list of a predefined, protecting, data set profile.

The source data set is RACF-indicated.

The target data set is RACF-indicated after the copy:

- If the target data set is not RACF-indicated and if either there is a discrete profile for the source or MENTITY is specified, DFSMSdss defines a discrete profile:
 - If the define is successful, the data set is RACF-indicated. It is accessible according to the access list for the model profile that was used.
 - If the define is unsuccessful and MENTITY was specified or DELETE was not specified, the data set is RACF-indicated. It may be inaccessible until a data set profile is successfully defined.
 - If the define is unsuccessful and DELETE was specified (without MENTITY), the copy or move is unsuccessful. The target data set is deleted, and the source data set is kept.
- If the target data set is already RACF-indicated, a discrete profile is not defined by DFSMSdss (even if you specify MENTITY or DELETE). The target data set remains RACF-indicated.

Other COPY Command Considerations

To copy or move a data set that is discretely protected, you need authority to define discrete profiles because the move operation temporarily renames the data set and the discrete profile.

COPYDUMP

The access authority you need to the input and output dump data sets depends on many factors not under DFSMSdss control because access-authorization checking is performed by Open/Close/End-of-Volume (O/C/EOV) processing. The primary considerations are as follows:

- You need READ access if the input dump data set is protected.

Data Security

- You need either UPDATE or ALTER access if the output dump data set is protected.
- You may need UPDATE access to the catalog if the output data set is going to be cataloged.

DEFRAG

To use the DEFrag command to relocate data extents on a DASD volume, you need DASDVOL-access authority to that volume or READ-access to the data sets that are relocated. Otherwise, only unprotected data sets are relocated.

As part of the DEFrag operation, DFSMSdss creates a work data set (SYS1.DFDSS.DEFRAG.xxxxxxx.volser.DUMMY), which is deleted after successful completion of the DEFrag function. If the DEFrag operation is prematurely ended, rerun the DEFrag function to clean up this data set.

DUMP

The access authority you need to the output dump data set depends on many factors not controlled by DFSMSdss because access-authorization checking is done by O/C/EOV. The primary considerations are as follows:

- You need access to a protected dump data set with:
 - ALTER access to create it.
 - UPDATE access if it is already allocated.
 - CREATE authority in the group and UPDATE access if it is a group data set.
- You may need UPDATE access to a catalog if the dump data set is going to be cataloged.
- You may have sufficient authority to the dump data set if you have the system-OPERATIONS attribute.
- You also need access authority to the data sets that are going to be dumped as shown in Table 11.

Table 11 is for the following conditions:

- Both the data set and the catalog are RACF-protected.
- You have the indicated level of access authority to the data set.
- The data set is cataloged in a user catalog.

Table 11. DUMP Command and Access Authority. This table shows the minimum access levels required to perform a DUMP command.

SMS-Managed Data Set?	With DELETE?	Access Level:	
		Source Data Set	Catalog
NO	NO	READ	Automatic
NO	YES	ALTER	UPDATE
YES	NO	READ	Automatic
YES	YES	ALTER	Automatic

Note: "Automatic" means that you have automatic access to the catalog if you can access the data set.

Dumping and Deleting Data Sets

For SMS-managed data sets, ALTER access to the data set lets you delete a data set. If the data set is cataloged in the master catalog, you also need UPDATE access to the catalog.

For non-SMS-managed data sets, you need one of the following to dump and delete a data set:

- ALTER access to a protected data set.
- READ access to a protected data set and ALTER access to a protected catalog.
- UPDATE access to a protected catalog if the data set is unprotected.
- DASDVOL-access authority to the volume containing the data set.
- Authority to specify DELETE as a DFSMSdss-authorized storage administrator.

PRINT

To use the PRINT command, you need either DASDVOL-access authority at the READ level or READ access to the data sets.

RELEASE

To use the RELEASE command, you need either DASDVOL-access authority at the UPDATE level or UPDATE access to the data sets.

RESTORE

The access authority you need to restore a data set from an input dump data set depends on many factors not controlled by DFSMSdss because access-authorization checking to the input dump data set is performed by O/C/EOV processing. The primary considerations are as follows:

- You need at least READ access to the dump data set if the dump data set is protected.
- You may need READ access to the catalog if the dump data set is cataloged.
- You may have sufficient authority to the dump data set if you have the system-OPERATIONS attribute.
- You also need access authority to the source and target data sets and their catalogs as shown in Table 12 on page 280.

Restore and Access Authorization

When a data set is either restored or both restored and renamed, DFSMSdss tries to verify that you have READ access to a DASD data set with the same name as the data set that was dumped. This is done to ensure that an unauthorized person is not allowed to restore and rename one of your data sets. For example, the restore described below would be unsuccessful:

- Data set USER1.PERSONAL.PAYHIST is RACF-protected so that USER2 cannot read it.
- Data set USER1.PERSONAL.PAYHIST is dumped.
- USER2 tries to restore USER1.PERSONAL.PAYHIST as USER2.TEST.DATA.

When you restore a data set, your authority to read that data set is checked using the current data set profiles rather than the data set profiles that were defined when the data set was dumped:

- If a discrete profile is predefined for the source data set name, it is used regardless of the current state of the data set.
- Otherwise, if a generic profile is defined, it is used.

Data Security

- If neither a discrete nor a generic profile is defined, you are given access. However, for VSAM data sets that are already cataloged and RACF-indicated, access is denied in this situation.
- For VSAM data sets, if the catalog no longer exists or the data set is no longer cataloged, only generic-profile checking is performed.

Table 12. Restore Command and Access Authority. This table shows the minimum access-levels required to perform a RESTORE command.

SOURCE DATA SET			TARGET DATA SET		
Non-SMS			Non-SMS		
RENAME or RENAMEU	Access Level		Preallocated Target	Access Level	
	Data Set	Catalog		Data Set	Catalog
NO	No check	No check	NO	ALTER	UPDATE
NO	No check	No check	YES	UPDATE	Automatic
YES	READ	Automatic	NO	ALTER	UPDATE
YES	READ	Automatic	YES	UPDATE	Automatic
Non-SMS			SMS		
RENAME or RENAMEU	Access Level		Preallocated Target	Access Level	
	Data Set	Catalog		Data Set	Catalog
NO	No check	No check	NO	ALTER	Automatic
NO	No check	No check	YES	UPDATE	Automatic
YES	READ	Automatic	NO	ALTER	Automatic
YES	READ	Automatic	YES	UPDATE	Automatic
SMS			Non-SMS		
RENAME or RENAMEU	Access Level		Preallocated Target	Access Level	
	Data Set	Catalog		Data Set	Catalog
NO	No check	No check	NO	ALTER	UPDATE
NO	No check	No check	YES	UPDATE	Automatic
YES	READ	Automatic	NO	ALTER	UPDATE
YES	READ	Automatic	YES	UPDATE	Automatic
SMS			SMS		
RENAME or RENAMEU	Access Level		Preallocated Target	Access Level	
	Data Set	Catalog		Data Set	Catalog
NO	No Check	No check	NO	ALTER	Automatic
NO	No check	No check	YES	UPDATE	Automatic
YES	READ	Automatic	NO	ALTER	Automatic
YES	READ	Automatic	YES	UPDATE	Automatic

Notes:

- “No check” means that no check is made.
- “Automatic” means that you have automatic access to the catalog if you can access the data sets.

Table 12 on page 280 is based on the following assumptions:

- The data set is a user data set.
- Both the data set and the catalog are RACF-protected.
- You have the indicated access authority to the source and target data sets.
- If an SMS-managed data set is going to be cataloged in the master catalog, you have UPDATE access to that catalog.

Table 12 on page 280 also applies to group data sets. To create a group data set by restoring it, you need one of the following:

- UPDATE access to the catalog for unprotected, non-SMS, group data sets. For unprotected, SMS-managed data sets, you automatically have access to the user catalog.
- CREATE authority in the GROUP and UPDATE access to the data set.
- ALTER access to the data set and be a member of the group that owns the data set. To discretely protect a group data set, you need additional authority such as CREATE in the group.

Restore and Data Set Profile Considerations

If the data set being restored was RACF-indicated when it was dumped, DFSMSdss does special target data set processing.

Restore and the MENTITY Keyword: You can restore a data set and use the MENTITY keyword to request that DFSMSdss is to define a data set profile when one of the following is true:

- The data set to be restored was RACF-protected by a generic profile when it was dumped, and the target data set name is unprotected.
- The data set to be restored was RACF-indicated when it was dumped, and the target data set name is either unprotected or it is only protected by a generic profile.

DFSMSdss uses the data set profile specified by the MENTITY keyword, and any volume serial specified with MVOLSER, as a model to define a discrete profile to protect the target data set. If MVOLSER is not specified, the volume serial is one of the following:

- The volume on which a non-VSAM data set resides.
- The volume that contains the catalog for a VSAM data set.

To define discrete profiles for data sets that you do not own, you need additional authorization such as the system-OPERATIONS attribute or CREATE in the group.

Restore and Physical Data Sets: DFSMSdss provides limited support for defining discrete profiles during physical restore of a data set:

- For VSAM data sets, a discrete profile is never defined.
- For non-VSAM data sets, a discrete profile is only defined if MENTITY is specified and the data set was RACF-indicated when it was dumped.

Restore and Define Discrete Profile Summary: Table 13 on page 282 summarizes the circumstances during a restore when MENTITY is effective, when a discrete profile is defined, when the target data set is RACF-indicated, and when related messages are issued.

Table 13. Copy and Define Profile Summary. This table summarizes when DFSMSdss defines discrete profiles for a copy operation.

Protection		MENTITY Specified?	Define Profile?	RACF Indicated?	Warning Message?
Source	Target				
None	None	Yes	No	No	No
		No	No	No	No
None	Generic	Yes	No	No	No
		No	No	No	No
None	RACF Indicated	Yes	No	Yes	No
		No	No	Yes	No
Generic	None	Yes	Yes	Yes	No (W)
		No	No	No	Yes
Generic	Generic	Yes	No	No	No
		No	No	No	No
Generic	RACF Indicated	Yes	No	Yes	No
		No	No	Yes	No
RACF Indicated	None	Yes	Yes	Yes	No (W)
		No	No	Yes	Yes
RACF Indicated	Generic	Yes	Yes	Yes	No (W)
		No	No	Yes	Yes
RACF Indicated	RACF Indicated	Yes	No	Yes	No
		No	No	Yes	No
Note: "No (W)" means that if the define of the discrete profile fails, the restore is successful, but the target data set remains RACF-indicated. A warning message is issued.					

Protection States after a Restore: When a data set is restored, the protection state of the target depends on the initial protection states of the source and the target data sets, whether MENTITY is specified, and whether discrete profiles are predefined. The following conditions may exist:

The data set was unprotected when it was dumped.

The target data set may or may not be protected after the restore. Even if you specify MENTITY, a discrete profile is not defined.

The data set was generically protected when it was dumped.

The target data set name may or may not be protected after the restore:

- If the target data set is not already protected and MENTITY is not specified, the data set remains unprotected. If MENTITY is specified, a discrete profile is defined:
 - If the define is successful, the data set is RACF-indicated. It is accessible according to the access list for the model profile that was used.
 - If the define is unsuccessful, the data set remains RACF-indicated. It may be inaccessible until a data set profile is successfully defined.
- If the target data set is already protected, a discrete profile is not defined (even if you specify MENTITY). The target data set is accessible according to the access list of a predefined, protecting, data set profile.

The source data set was RACF-indicated when it was dumped.

The target data set is RACF-indicated after the data set is restored:

- If the target data set is not already RACF-indicated and if MENTITY is not specified, a discrete profile is not defined. If MENTITY is specified, DFSMSdss defines a discrete profile:
 - If the define is successful, the data set is accessible according to the access list for the model profile that was used.
 - If the define is unsuccessful, the data set may be inaccessible until a data set profile is successfully defined.
- If the target data set is already RACF-indicated, a discrete profile is not defined by DFSMSdss (even if you specify MENTITY). The target data set remains RACF-indicated.
- DFSMSdss gives you the ability to change its default operation. Through the ADRPATCH Serviceability Aid, DFSMSdss logical restore can be instructed not to turn on the RACF indicator for the target data set when the source data set was RACF-indicated at dump time. The MENTITY keyword is not specified and the target data set is protected by a generic profile.

Related reading: For additional information about changing the DFSMSdss default operation, see the *z/OS DFSMSdss Storage Administration Guide*.

RESTORE command and the IMPORT Keyword

By default, anyone can use the IMPORT keyword of the RESTORE command to bypass access checking of source data sets. You still need access authorization to create or update the target data sets and catalogs. Your installation may limit use of the IMPORT keyword as previously discussed in “Protecting Usage of DFSMSdss” on page 257.

Appendix A. Coexistence Considerations

This appendix presents several considerations whenever you migrate to a newer version of DFSMSdss.

ALLMULTI Keyword

The ALLMULTI keyword is no longer supported in DFSMS/MVS V1R4 and later releases, and has been replaced by the SELECTMULTI keyword. For information see the descriptions for the SELECTMULTI keyword in the CONVERTV, COPY, and DUMP command sections.

Restoring Existing DFDSS Dumps using DFSMSdss

Dumps created with an older version or release of DFDSS or DFSMSdss can be restored with a newer version or release. IBM does not recommend **and does not guarantee** that dumps created with a newer version or release of DFSMSdss can be restored with an older version or release of DFDSS or DFSMSdss. However, if the dump does not contain any functional changes introduced in a newer release, a restore using an older release may be successful.

HFS, compressed-format sequential and extended-format KSDS data sets cannot be restored by an older version or release of DFDSS, or by DFSMSdss prior to DFSMS/MVS V1R2. Striped VSAM data sets cannot be restored by DFSMSdss prior to OS/390 V2R10.

The following cannot restore DFSMSdss dumps of VM-format volumes that are created with DFSMS/MVS V1R3 with APAR OW18174, or later releases, using the CPVOLUME keyword:

- DFSMS/MVS V1R3 without APAR OW18174
- All releases of DFDSS
- The DFSMSdss Stand-Alone Restore program

zFS data sets can be restored by DFSMSdss prior to z/OS Version 1 Release 3; however, the zFS indicator in the catalog is not set. Therefore, zFS data sets do not have the zFS indication in the LISTCAT listing. The zFS indicator is turned back on the next time the data set is mounted to a z/OS Version 1 Release 3 system. When this occurs, the zFS indication is displayed in the LISTCAT listing.

Appendix B. Data Integrity—Serialization

DFSMSdss uses **volume serialization** and **data set serialization** functions to ensure that data sets are not modified during the processing of DFSMSdss commands. Volume serialization is accomplished by using the RESERVE macro. Data set serialization is accomplished by using the ENQ macro and the DFSMSdss DYNALLOC function. In the case of shared DASD, volume serialization ensures that data sets are not being added, deleted, renamed, or extended from either the same processor or other processors.

You can control volume serialization with the enqueue installation exit routine. This allows other programs to access volumes while DFSMSdss is running. Volume serialization, however, cannot be overridden.

DFSMSdss supports data sets that are always open, or open for long periods of time, with **backup-while-open serialization**.

DFSMSdss supports backup-while-open processing for DFSMStvs and for two types of database applications: Customer Information Control System (CICS) and information management system (IMS).

CICS support includes both RLS CICS and non-RLS CICS backup-while-open.

DFSMStvs supports backup-while-open with or without CICS®, as “Backup-While-Open Data Sets (CICS and DFSMStvs)” on page 297 describes.

DFSMSdss also supports backup-while-open serialization for IMS data sets for:

- Indexed VSAM data sets, such as KSDS
- Non-indexed VSAM data sets, such as ESDS
- Non-VSAM data sets, such as OSAM

Related reading:

- For additional information about enqueue installation exit routine, see *z/OS DFSMS Installation Exits*.
- For additional information about RLS CICS and non-RLS CICS backup-while-open support, see “Backup-While-Open Data Sets (CICS and DFSMStvs)” on page 297.
- For additional information about IMS backup-while-open support, see “Backup-While-Open Data Sets (IMS)” on page 301.
- For additional information about DFSMStvs, see *z/OS DFSMStvs Planning and Operating Guide* and *z/OS DFSMStvs Administration Guide*.

Volume Serialization

For volume serialization, DFSMSdss issues the RESERVE macro against the volume to prevent direct access device storage management (DADSM) functions (such as addition, deletion, extension, or renaming of data sets) from changing the VTOC entries during DFSMSdss processing. The RESERVE is issued before processing is begun on a particular volume, and released when processing is completed, during the following operations:

- DUMP (except logical)
- COPY (except logical)

Data Integrity—Serialization

- RESTORE (except logical and physical data set)
- PRINT (except data set)
- DEFrag

Note: Be aware that volume serialization (RESERVE performed against the volume) does not ensure integrity at the data set level. For example, in the case of a physical full volume dump, data sets are not serialized and the data sets could change while the DUMP is being taken.

Note that you can use the DFSMSdss enqueue installation exit to request that DFSMSdss only reserve the VTOC for the duration of VTOC access during the following operations:

- Full, tracks, and physical data set DUMP
- Full and tracks COPY
- Tracks PRINT

In addition, DFSMSdss issues the RESERVE macro against a volume while updating information in the VTOC, such as the RACF-defined flag and the data-set-changed flag. DFSMSdss also issues the RESERVE macro against a volume while accessing the information in the VTOC to perform data set selection during the following operations:

- Logical DUMP with input volumes specified
- Logical COPY with input volumes specified
- CONVERTV
- COMPRESS
- RELEASE

Related reading: For additional information about the enqueue installation exit routine, see *z/OS DFSMS Installation Exits*.

Avoiding Lockout

The VTOC is locked to prevent activity on the VTOC at the volume level during the copy, defrag, dump, print (tracks) or restore (tracks and full) functions. These functions may require access to the catalogs for the data sets on the volume. A lockout can result between DFSMSdss and another job on the same system that has already locked the catalog but needs the VTOC for a DADSM function. To avoid this lockout, you should not run any jobs (for example, Access Method Services jobs) that require control of both the catalog and the VTOC while DFSMSdss is executing.

Programming Interface information

When an application program attaches a DFSMSdss dump or restore task and a task that invokes dynamic allocation to allocate a data set (for example, logical restore operation) or invokes DADSM to scratch a data set, the two tasks could lock each other out. To avoid this lockout, DFSMSdss uses the following ENQ scheme:

- Before DFSMSdss invokes DADSM SCRATCH, an exclusive ENQ is requested for the resource with a major name of ADRLOCK and a minor name of the volume serial number (padded with blanks to 8 bytes).
- Before DFSMSdss invokes dynamic allocation to allocate a new data set on a specific volume, a shared ENQ is requested for the resource with a major name of ADRLOCK and a minor name of the volume serial number (padded with

blanks 8 bytes). If this is a nonspecific dynamic allocation request, an exclusive ENQ with the resource with a major name of ADRLOCK and a minor name of NONSPEC is requested.

- If an application program invokes a DFSMSdss FULL dump operation and plans on attaching another DFSMSdss logical restore task, the application should request the first resource in exclusive mode (ADRLOCK, volser). The application should request the second resource in shared mode (ADRLOCK, NONSPEC) before attaching the FULL DUMP command task. Multiple dumps can be active in the same address space, and a logical restore request will not lock out that task.
 - If an application program issues a DADSM request, the application must enqueue on ADRLOCK/volser in a shared ENQ or ADRLOCK/NONSPEC in an exclusive ENQ.

Notes:

1. When DFSMSdss is invoked with JCL, lockout does not occur because DFSMSdss allocates a volume before invoking DADSM. If a FULL DUMP command has the volume enqueued and a restore task has been applied against the same volume, the restore task is held until the volume is available.
2. Only non-SMS, non-VSAM DADSM requests require serialization.

End of Programming Interface information

The WAIT Option

DFSMSdss lets you change the WAIT/RETRY values for system resources.

Related reading: For additional information about WAIT/RETRY, see *z/OS DFSMSdss Storage Administration Guide*.

Data Set Serialization

This section describes the ENQUEUE and SERIALIZATION options for data set serialization, the WAIT option, and provides an example of RESERVE-ENQUEUE processing.

Enqueuing—ENQ

The following material describes enqueue options for data set serialization. Enqueues are done on the data set name for the data set copy, data set dump, data set restore, defragment, print, compress, and release operations to prevent multiple, simultaneous updates to the same data set.

The SHARE option has unique properties when applied to the following commands:

- For the RESTORE command, SHARE applies to non-VSAM data sets only.
- For the DUMP and COPY commands, SHARE applies to non-VSAM data sets and VSAM data sets that are defined with share options other than (1,3) and (1,4).

If you do not specify the SHARE option, DFSMSdss tries to provide the highest level of data integrity by defaulting to exclusive enqueueing. The command is in a wait state for X seconds if the enqueue fails. WAIT specifies X (numsecs, numretries), and retries the enqueue. If the wait-enqueue sequence fails after Y retries (where WAIT specifies Y (numsecs, numretries)), data set processing ends.

Data Integrity—Serialization

In the case of a multistep job where the initiator holds a shared enqueue on the data set, DFSMSdss upgrades the enqueue to exclusive unless SHARE is specified. The initiator holds the exclusive enqueue until the last step in the job that references the data set has completed.

If you specify the SHARE keyword, DFSMSdss tries an enqueue for share. If it fails, it goes through the same logic as if SHARE had not been specified. If the retries all fail, processing ends for the data set.

You can specify TOLERATE(ENQFAILURE) in addition to the default, ENQ, or the SHARE option. If TOLERATE(ENQFAILURE) is specified, DFSMSdss attempts to get the specified level of enqueue, exclusive or share. If the enqueue fails after the specified or default number of retries, DFSMSdss processes the data set without an enqueue. Specify TOLERATE(ENQFAILURE) if you are willing to tolerate the exposure of not having data integrity in order to force the successful completion of that particular data set operation. This is particularly useful when an installation has duplicate data sets (different data sets with the same name but on different volumes) and you want to run DFSMSdss on a data set on one volume while the data set with the same name is being used by the system or another job on a different volume.

Note: Because of ENQ contention, SYSPRINT data sets should not be allocated on volumes being processed. TOLERATE(ENQFAILURE) cannot apply to data movements that involve the use of utilities.

Table 14 shows the data set enqueue options.

TOLERATE(ENQFAILURE) can be used on VSAM and non-VSAM data sets.

Table 14. Data Set Enqueue Options for Non-VSAM Data Sets Specified on DFSMSdss Commands

Options	None	SHARE	TOLERATE(ENQFAILURE)	SHARE and TOLERATE(ENQFAILURE)
Type of enqueue attempted	Exclusive	Share	Exclusive	Share
If enqueue is successful	Process	Process	Process	Process
If enqueue is not successful	Do not process	Do not process	Process without enqueue	Process without enqueue

Dumping HFS Data Sets

The serialization mechanism operates differently depending on whether you are performing a logical dump or a physical dump. Whether or not you mount the data set before you dump it affects serialization also.

Logical Dump

The serialization mechanism used during a logical dump of a mounted HFS data sets consists primarily of the SYSZDSN enqueue macro and the BPX1QSE quiesce macro. A shared SYSZDSN enqueue provides serialization if a data set that you want to dump is not mounted for update. To mount an HFS data set for update, z/OS UNIX System Services (z/OS UNIX) must obtain an exclusive enqueue on the SYSZDSN. The shared enqueue on the SYSZDSN prevents z/OS UNIX from mounting the data set for update during the dump.

If you mount an HFS data set for update before you dump it, DFSMSdss issues the BPX1QSE macro to prevent updates to the data set for the duration of the dump. For non-shared HFS environments, you must run the DFSMSdss dump job on the same system where the data set is mounted for update. For shared HFS environments, you can run the DFSMSdss dump job from any system that is part of the shared HFS plex.

DFSMSdss can honor the DELETE keyword for an HFS data set during logical dump only if the data set is unmounted. For delete processing, the data set must be enqueued exclusively.

Physical Dump

DFSMSdss relies on a SYSDSN enqueue for physical dump operations. By default, DFSMSdss attempts an exclusive SYSDSN enqueue during physical dump. As long as the data set is not mounted at dump time, the exclusive enqueue on the SYSDSN provides sufficient serialization. If the data set is mounted before the physical dump job begins, z/OS UNIX will have a shared enqueue on the SYSDSN. DFSMSdss will therefore not be able to obtain an exclusive SYSDSN enqueue.

Avoid performing a physical dump of mounted HFS data sets because quiesce is not available during a physical dump. If you specify either the SHARE or TOL(ENQF) keyword during a physical dump, the internal control information and data inside the HFS can change during the dump. This can result in a dumped data set which contains an HFS data set that might not be usable after you have restored it. If you must physically dump a data set that is in use, use TOL(ENQF) instead of SHARE. With TOL(ENQF), you receive a return code of four, along with a warning message, if serialization was not adequate during the dump.

Guideline: Exercise caution when you use TOL(ENQF) during a physical dump of HFS data sets. Unlike other types of data sets, if an HFS is updated during a physical dump with TOL(ENQF), a subsequent restore can likely result in an unusable data set.

Dumping zFS Data Sets

The serialization mechanism operates differently depending on whether you are performing a logical dump or a physical dump. Whether or not you mount the data set before you dump it affects serialization also.

Logical Dump

The serialization mechanism used during a logical dump of zFS data sets consists primarily of the SYSDSN enqueue macro, SYSVSAM enqueue macros and the zFS quiesce macro. A shared SYSDSN enqueue and SYSVSAM enqueues provide serialization if a data set that you want to dump is not mounted.

If a zFS data set is mounted before you dump it, the data set can still be quiesced. You must, however, run the DFSMSdss dump job on the same system on which the data set is currently mounted. The quiesce action prevents updates to the data set for the duration of the dump.

Physical Dump

DFSMSdss relies on a SYSDSN and SYSVSAM enqueues for physical dump operations. By default, DFSMSdss attempts exclusive enqueues during physical dump. As long as the data set is not mounted at dump time, the exclusive enqueues provide sufficient serialization. If the data set is mounted before the physical dump job begins, z/OS UNIX will have the zFS data set allocated and

Data Integrity—Serialization

opened. Allocation and open processing will have obtained SYSDSN and SYSVSAM enqueues. DFSMSdss will therefore not be able to obtain the exclusive enqueues.

Physical dump of mounted zFS data sets is not recommended because quiesce is not available during a physical dump. If you specify the TOL(ENQF) keyword during a physical dump, the internal control information and data inside the zFS can change during the dump. This can result in a dumped data set which contains a zFS data set that might not be usable after you have restored it.

Dynamic Allocation (DYNALLOC)

DYNALLOC is an option for the data set DUMP, data set COPY, data set RESTORE, data set PRINT, DEFrag, COMPRESS, and RELEASE commands. It allows serialization of data sets across processors with shared DASD, with JES3, and with the interface between dynamic allocation and JES3 enabled. Dynamic allocation is used by DYNALLOC to serialize data sets. Processing time increases because overhead is involved in dynamic allocation and serialization across multiple processors.

If you use DYNALLOC to serialize data sets (as opposed to ENQ) the job run time increases. If you use INDYNAM instead of DD statements to allocate DASD volumes you do not appreciably increase run time and coding of JCL, and command input is easier.

DFSMSdss does *not* honor DYNALLOC for HFS source data sets for either logical data set copy or logical data set dump.

Enqueuing versus Dynamic Allocation of Data Sets

For data set operations, the default method of serializing usage of data sets is to enqueue on the data set name. If the DYNALLOC keyword is coded in the control cards, the time taken to serialize is much greater than that taken by using the ENQ macro. So use DYNALLOC keyword judiciously. Use it only on a JES3 system if the interface between the allocation function and JES3 is not disabled. If the interface is disabled, then the end result of serialization using ENQ is the same as using dynamic allocation; ENQ takes much less time to serialize data sets.

When you use DYNALLOC keyword, the selected data sets are allocated to DFSMSdss. If some data sets are allocated to the system, they can only be processed by DFSMSdss with DYNALLOC if SHARE is also specified. Table 16 on page 295 provides additional information on data set serialization.

ReadWrite Serialization Scheme

Table 15 shows the serialization scheme used by all operations, except COPYDUMP, for read/write access.

Table 15. Read/Write Access Serialization Scheme

	ENQ Names		ENQ Control, Exclusive (E) or Share (S)	ENQ Scope
	Major Name	Minor Name		
For Source HFS Data Sets:				
followed by:	ADRDSN SYSZDSN	Data set name Data set name	E (see Note 1) S (see Note 1) E (see Note 2) S (see Note 2)	STEP SYSTEMS
For zFS Data Sets:				
On the component during logical data set DUMP or COPY operations:	SYSVSAM	See Note 3	E (see Note 5) S (see Note 5)	SYSTEMS
On the component during physical data set DUMP operations:	SYSVSAM	See Note 3	E (see Note 1) S (see Note 1)	SYSTEMS
On the component during other operations:	SYSVSAM	See Note 3	E	SYSTEMS
On the cluster during logical DUMP or COPY operations: followed by:	ADRDSN SYSDSN	Cluster name Cluster name	E (see Note 1) S (see Note 1) E (see Note 5) S (see Note 5)	STEP SYSTEM
On the cluster during other operations: followed by:	ADRDSN SYSDSN	Cluster name Cluster name	E (see Note 1) S (see Note 1) E (see Note 1) S (see Note 1)	STEP SYSTEM
For Non-VSAM Data Sets:				
followed by:	ADRDSN SYSDSN	Data set name Data set name	E (see Note 1) S (see Note 1) E (see Note 1) S (see Note 1)	STEP SYSTEM
For GDG Data Sets:				
On the GDG base: followed by:	ADRDSN SYSDSN	GDG base name GDG base name	E (see Note 1) S (see Note 1) E (see Note 1) S (see Note 1)	STEP SYSTEM
On each GDG data set: followed by:	ADRDSN SYSDSN	Data set name Data set name	E (see Note 1) S (see Note 1) E (see Note 1) S (see Note 1)	STEP SYSTEM
For VSAM Data Sets:				
On each component during data set DUMP or COPY operations:	SYSVSAM	See Note 3	E (see Note 1) S (see Note 1)	SYSTEMS
On each component during other operations:	SYSVSAM	See Note 3	E	SYSTEMS

Data Integrity—Serialization

Table 15. Read/Write Access Serialization Scheme (continued)

	ENQ Names		ENQ Control, Exclusive (E) or Share (S)	ENQ Scope
	Major Name	Minor Name		
On the cluster: followed by:	ADRDSN	Cluster name	E (see Note 1) S (see Note 1)	STEP
	SYSDSN	Cluster name	E (see Note 1) S (see Note 1)	SYSTEM
For Integrated Catalog Facility Catalogs:				
In addition to the above VSAM data set information:	SYSIGGV2	Catalog name	E	SYSTEMS
On the volume:	SYSVTOC	Volume serial number	E	SYSTEMS
For VVDS Data Sets:				
On each data set: Input access: Output access:	SYSZVVDS	SYS1.VVDS.Vvolser	S E	SYSTEMS
Notes:				
<p>1. If the selected control for the ENQs is EXCLUSIVE or SHARE, the SHARE keyword determines the type of control. If SHARE is <i>not</i> specified, exclusive control of the data set is obtained. Whereas, if SHARE is specified, shared control of the data set is obtained.</p> <p>If shared control is obtained for a VSAM data set because the SHARE keyword was specified, other programs are able to obtain read access, but not write access, to the data set, while it is being processed. The SHARE keyword is honored for VSAM data sets only during DUMP and COPY processing, and only for VSAM data sets that are defined with SHARE options other than (1,3) or (1,4).</p>				
<p>2. For a source HFS data set, DFSMSdss ignores DYNALLOC. If you specify DELETE, then DFSMSdss attempts to get an exclusive SYSZDSN enqueue. If you do not specify DELETE, then DFSMSdss attempts to get a shared SYSZDSN enqueue.</p>				
<p>3. If DYNALLOC is used:</p> <p>For a source HFS data set, DFSMSdss ignores DYNALLOC.</p> <p>For non-VSAM data sets and GDG bases, instead of ENQ, the data set is dynamically allocated automatically, with a disposition of OLD if SHARE is not used; otherwise, SHARE is the disposition.</p> <p>For VSAM data sets, in addition to the ENQ on the components, the cluster is allocated dynamically just as for non-VSAM data sets.</p>				
<p>4. The minor name used for enqueueing VSAM components consists of the following: component name, catalog name, L1, L2, L3, A (where L1 is the total length of the minor name, L2 is the component name length, and L3 is the catalog name length).</p> <p>On a data set DUMP, data set RESTORE, data set COPY, and DEFrag operation, an enqueue is performed once with the character A=I and iteratively with A=O.</p>				
<p>5. If you specify DELETE, then DFSMSdss attempts to get exclusive control of the zFS. If you do not specify DELETE, then DFSMSdss attempts to get a shared control of the zFS.</p>				

Programming Interface information

Note: Although the ENQ on the major name ADRDSN is mostly intended to coordinate access to data sets between multiple DFSMSdss commands, it might be necessary for application programs that invoke DFSMSdss to make use of the ENQ; data sets which are serialized by the application program using dynamic allocation or an ENQ on the major name SYSDSN could be processed by DFSMSdss unless the application uses the ENQ on ADRDSN.

End of Programming Interface information

Table 16. Resource Serialization

Function	Data Set Type	Volume Level Serialization VTOC VVDS	Data Set Level Serialization (ENQ or DYNALLOC)	
COMPRESS	Non-VSAM	Yes	N/A	DSName
CONVERTV	Non-VSAM	Yes	N/A	DSName
Data Set COPY	Non-VSAM	Yes	N/A	DSName
	VSAM	Yes	Yes	Component and Cluster Names
DEFrag	Non-VSAM	Yes	N/A	DSName
	VSAM	Yes	Yes	Component and Cluster Names
Full DUMP	Non-VSAM	Yes	N/A	N/A
	VSAM	Yes	Yes	N/A
Data Set DUMP	Non-VSAM	Yes	N/A	DSName
	VSAM	Yes	Yes	Component and Cluster Names
Data Set PRINT	Non-VSAM	Yes	N/A	DSName
	VSAM	Yes	Yes	Component Name
Tracks PRINT	N/A	Yes	N/A	N/A
RELEASE	Non-VSAM	Yes	N/A	DSName
	VSAM	Yes	Yes	Component and Cluster Names
Full RESTORE	Non-VSAM	Yes	N/A	N/A
	VSAM	Yes	Yes	N/A
Data Set RESTORE	Non-VSAM	N/A	N/A	DSName
	VSAM	N/A	Yes	Component and Cluster Names
Notes:				
<ul style="list-style-type: none"> • Refer to “Backup-While-Open Data Sets (CICS and DFSMStvs)” on page 297 for more information on resource serialization. • VSAM data sets must be cataloged in an integrated catalog facility catalog. • N/A means not applicable. 				

WAIT Option

This option allows you to specify how long, in seconds, DFSMSdss is to wait for a resource and the number of times DFSMSdss is to retry, should an ENQ or RESERVE fail. The default is WAIT(2,2).

This means DFSMSdss is to retry twice at 2-second intervals. The WAIT keyword does not apply to system resources such as the VTOC and the VVDS (see below).

For a data set copy or logical data set dump operation, the WAIT option has a different meaning for serializing data sets when multiple data sets are to be processed and WAIT(0,0) is not specified. Multiple passes are made through the list of data sets that are selected. On each pass, those data sets that can be serialized without waiting for the resource and that were not processed before are processed. At the end of a pass, if none of the data sets could be processed without waiting for a resource, then, in the next pass, at the first occurrence of a data set that was not processed a WAIT is issued.

That data set and the remainder of the list is processed if possible. The above procedure is repeated until all data sets are processed or the WAIT limits are reached.

Data Integrity—Serialization

For example, if WAIT(3,10) is specified and 5 data sets are left to be processed, up to 10 passes are made. On each pass, the first unprocessed data set is waited upon for 3 seconds. Thus, only a 30-second maximum is ever waited, not 150 (5 times 3 times 10).

For system resources, such as the VTOC and the VVDS data set, the default is WAIT(3,30). This means DFSMSdss is to retry 30 times at 3 second intervals, resulting in a total wait time of 90 seconds.

An Example of RESERVE-ENQUEUE Processing

If you were to enter the following:

```
DUMP INDD(IN1) OUTDD(OUT1) -
DATASET(INCLUDE(MY.DUMP.DATASET1,MY.DUMP.DATASET2)) -
WAIT(5,2) -
SHARE
```

The following would result:

1. DFSMSdss issues a RESERVE command on SYSVTAC (this is the default)
 - If the reserve operation fails, DFSMSdss retries thirty times at 3-second intervals, assuming that the installation has not changed the default for system resources.
 - If the RESERVE command is successful, DFSMSdss continues.
If either of the data set was a VSAM or SMS-managed data set, the VVDS would also be serialized by DFSMSdss. If VVDS serialization fails, none of the data sets are dumped.
2. DFSMSdss issues a shared enqueue on data set name. If the shared enqueue fails, DFSMSdss retries two times at 5-second intervals.
 - If the shared enqueue is successful:
 - DFSMSdss adds the data set name to the list to be dumped.
 - DFSMSdss loops back to the beginning of this step until an enqueue is tried for all data sets.
 - If the shared enqueue fails, DFSMSdss:
 - Issues a message indicating that the data set failed serialization, and does not process the data set.
 - Loops back to the beginning of this step until an enqueue is tried for all data sets.
3. After an enqueue is tried for all specified data sets and at least one was successful, DFSMSdss:
 - Dumps the VVDS (if an integrated catalog facility data set was selected) and the VTOC and dequeues the VTOC.
 - Dequeues VVDS if it was enqueued.
 - Dumps each data set that was successfully enqueued.
 - Dequeues the enqueued data sets.
 - Issues message indicating which data sets were successfully processed.

Note: If you specify TOLERATE(ENQFAILURE) option with the DUMP command and the installation options exit routine does not override it, the VTOC is not dequeued until all the data tracks for all the data sets are dumped.

Backup-While-Open Data Sets (CICS and DFSMStvs)

DFSMSSdss supports backup-while-open serialization, which can perform backup of data sets that are open for update for long periods of time. It can also perform a logical data set dump of these data sets even if another application has them serialized. Backup-while-open is a better method than using SHARE or TOLERATE(ENQFAILURE) for dumping Customer Information Control System (CICS) VSAM file-control data sets that are in-use and open for update. When you dump data sets that are designated by CICS as eligible for backup-while-open processing, data integrity is maintained through serialization interactions between CICS (data base control program), CICSVR (forward-recovery program), VSAM record management, DFSMSdfp, and DFSMSSdss. When you dump data sets that are designated by DFSMStvs as eligible for backup-while-open processing, data integrity is maintained through serialization interactions between DFSMStvs, VSAM record management, DFSMSdfp™, and DFSMSSdss.

Although the BWO(TYPECICS) parameter applies to both CICS and DFSMStvs, DFSMStvs enables you to back up a data sets while they are open whether or not you are running CICS.

Figure 7 shows the backup-while-open serialization for dumping CICS data sets that are open for update. Backup-while-open processing also ensures that any update activity that may invalidate the dump is detected. Simultaneous recovery or deletion of the data set while it is being dumped is also prevented.

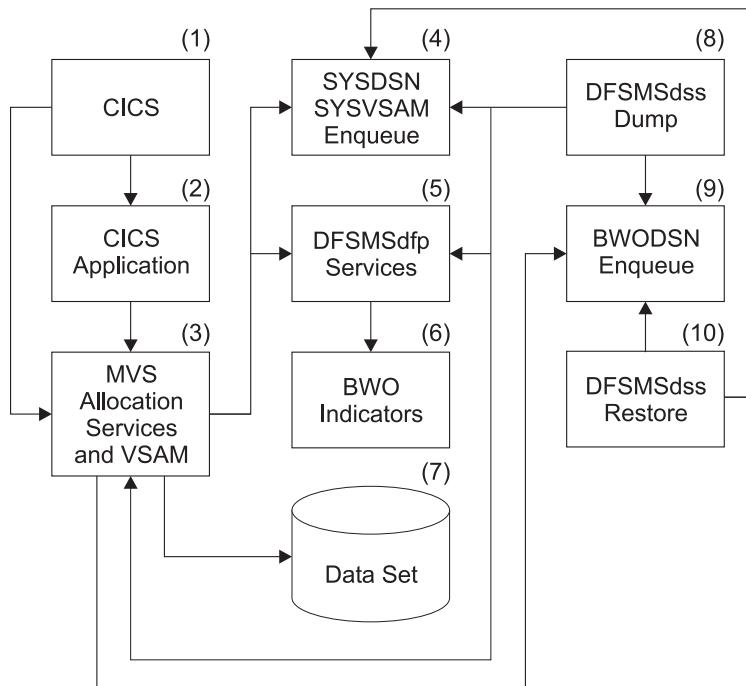


Figure 7. Block Diagram for Backup-While-Open Serialization

In Figure 7, a VSAM file-control data set (7) is allocated for CICS (1) through MVS allocation services (3) using JCL or dynamic allocation methods. This results in serialization through an enqueue on the name of the data set and the resource name SYSDSN (4). When the VSAM data set is opened (3), another level of serialization occurs through an enqueue on the names of the components of the

Data Integrity—Serialization

VSAM data set and the resource name SYSVSAM (4). For eligible data sets, CICS uses DFSMSdfp (5) to set a status in the backup-while-open indicators (6) in the catalog entry for the data set.

For a dump operation, DFSMSdss (8) attempts to acquire the SYSDSN, SYSVSAM, and backup-while-open (9) enqueues for the data set. When the enqueue on the cluster name of the data set and resource name of BWODSN is acquired, but not the enqueues for both SYSDSN and SYSVSAM, DFSMSdss uses DFSMSdfp to get the backup-while-open indicators, and starts to dump the open data set if it is backup-while-open eligible.

The backup-while-open enqueue is used to prevent more than one DFSMSdss operation, such as a simultaneous dump and restore (10), and to prevent the data set from being deleted while it is being dumped by DFSMSdss.

While the data set is being dumped, a data base application program may update the data set in a manner that invalidates the data set. For example, a control-interval or control-area split may occur. When this happens, VSAM record management uses DFSMSdfp to change the backup-while-open status. When the backup of the open data set is completed, DFSMSdss obtains the current backup-while-open indicators and invalidates the dump of the data set if the indicators are different from when the dump was started. When concurrent copy is used, updates made while the data set is being dumped do not cause the dump to be invalidated.

Backup-While-Open Status Definition

DFSMSdss interprets backup-while-open status numbers as follows:

Status	Meaning to DFSMSdss
000	Normal serialization and processing techniques are used to dump the data set.
001	CICSVR forward-recovers the data set.
010	A control-interval split, control-area split, or extend of the data set was either interrupted before the dump started or is currently in process.
011	A control-interval split, control-area split, or an extend of the data set completed successfully. CICS or DFSMStvs closed the data set and there is no mismatch between the base cluster and any alternate index.
100	The data set may be dumped while it is open for update.
101	The data set was restored and is down-level; CICSVR or DFSMStvs must process it before a database application uses it.
110	The data set was updated and a split or extend is completed, which invalidates any dump that was in progress. When concurrent copy is used, the chances of an in-progress-dump being invalidated by a split or extend is significantly reduced.
111	The data set is in an indeterminate state.

Backup-While-Open Processing

DFSMSdss actions resulting from dumping a data set are based on the following conditions:

- Success or failure to acquire an exclusive enqueue for the resource names of SYSDSN, SYSVSAM, and BWODSN.
- Backup-while-open status *before* the dump.
- Backup-while-open status indicators after the dump.

When DFSMSdss acquires all of the exclusive enqueues: The data set is not open for update, even though it has a non-zero backup-while-open status. The particular processing action is a direct result of the initial backup-while-open status as follows:

- Status is 000; the data set is dumped. When it is restored, the backup-while-open status is restored as 000.
- Status is 001. The data set is dumped and the backup-while-open status is preserved and restored when the data set is subsequently restored. This data set is in an indeterminate state because of an incomplete forward recovery by CICSVR, which cannot forward-recover the restored data set.
- DFSMSdss lets you dump and restore this data set for nonstandard recovery purposes. After restoring the data set and correcting all errors in the data set, reset the BWO status to 000 by using CICS-provided methods. The preferred action is for you to restore an earlier dump of the data set and use CICSVR to do forward-recovery processing. This maintains the integrity of the data.
- Status is 010. DFSMSdss did not dump the data set. Take corrective action before using the data set by performing the following actions:
 - Determine if alternate indexes (AIX) are present for the sphere. If they are, do either of the following tasks:
 - Rebuild the AIXs from the base cluster, and reset the BWO status to 000 by using the CICS' methods.
 - Restore an earlier dump of the sphere, and use CICSVR to do forward-recovery processing.
 - If there are no AIXs, the data set is usable, as it is. Reset the BWO status to 000 by using the CICS methods.
- Status is 101. The data set is dumped and the backup-while-open status is preserved and restored when the data set is subsequently restored. This lets you process the data set with CICSVR before a data base application uses it.
- Status is 011, 100, 110, 111. The backup-while-open status is altered to 000 before the data set is dumped. When it is restored, the backup-while-open status is restored as 000.

When DFSMSdss acquires an exclusive enqueue on BWODSN (but not SYSDSN and SYSVSAM): Another program is using the data set or the data set is open. The particular processing action is a direct result of the initial backup-while-open status as follows:

- Status is 000. The data set is not eligible for backup-while-open. The data set is not dumped unless SHARE or TOLERATE(ENQFAILURE) is specified and the necessary conditions for those keywords are met.
- Status is 001, 010, 011, 101, or 111; the data set is not dumped.
- Status is 110; the backup-while-open status is altered to 100 and the data set is dumped (see below).
- Status is 100. The data set is dumped, even though it is already in use (including being open for update by another program). When the data set is restored, the

Data Integrity—Serialization

backup-while-open status is set to 101. This ensures that CICSVR or DFMSMStvs can process the data set, prior to its use by a data base application. The dump is invalidated if the backup-while-open indicators change while the data set is being dumped. The chances of this invalidation occurring reduce significantly when you use concurrent copy.

Backup-While-Open and Concurrent Copy

Concurrent copy improves backup-while-open processing by significantly reducing the chances of the invalidation of a backup-while-open dump because of updates to the data set. To use concurrent copy, specify the CONCURRENT keyword when you dump backup-while-open data sets. The following is a comparison of the various kinds of dumps you can ask for:

- **Normal dump.** Use of the data set must be quiesced. DFSMSdss obtains serialization, dumps the data set, and then releases the serialization. The data set cannot be used for the entire time.
- **Concurrent copy dump.** Use of the data set must be quiesced. DFSMSdss obtains serialization, performs concurrent copy initialization, releases serialization, and then dumps the data set. DFSMSdss completes concurrent copy initialization within a very short time (compared to the actual time to dump the data set). DFSMSdss can use the data set as soon as the concurrent copy initialization is complete.
- **Backup-while-open dump.** Use of the data set does not need to be quiesced. DFSMSdss dumps the data set without obtaining serialization. The data set can remain in use for the entire time, but update activity can invalidate the dump at any time during the dump.
- **Backup-while-open dump using concurrent copy.** Use of the data set does not need to be quiesced. DFSMSdss does not obtain serialization. DFSMSdss performs the concurrent copy initialization. DFSMSdss completes concurrent copy initialization within a very short time (compared to the actual time to dump the data set). The data set can remain in use for the entire time. Like the Backup-while-open dump, update activity during the dump can invalidate the dump. However, only update activity that occurs *before* DFSMSdss performs the concurrent copy initialization can invalidate the dump. The chances of the update activity invalidating the dump are significantly reduced because the concurrent copy initialization completes very quickly.

TOLERATE(ENQFAILURE) and SHARE Considerations

Using TOLERATE(ENQFAILURE) modifies the processing and serialization for backup-while-open data sets. The data sets are dumped when the backup-while-open status is 100, even though none of the enqueues are successfully acquired.

To maintain data integrity and data security, do not specify either SHARE or TOLERATE(ENQFAILURE) when you dump backup-while-open data sets.

An exclusive enqueue on BWODSN resource name for the data set is required for DFSMSdss to alter the backup-while-open status. DFSMSdss only attempts an exclusive enqueue on the BWODSN resource name. Unless the backup-while-open status is already 100, the dump fails even though you specified SHARE or TOLERATE(ENQFAILURE).

CICS Recovery Data

CICS maintains recovery data in the form of a date and time-stamp in the catalog entry for backup-while-open data sets. This information is not used or processed by DFSMSdss, but it is dumped and restored to preserve that information for CICSVR. The recovery data is also printed in selected messages to assist you in your recovery efforts.

DFSMStvs maintains recovery data in forward recovery log records. DFSMSdss does not use or process this recovery data. The data is dumped and restored to preserve the information for CICSVR or possibly for another forward recovery utility. The recovery data is also printed in selected messages to assist you in your recovery efforts.

Backup-While-Open Data Sets (IMS)

DFSMSdss supports backup-while-open processing of IMS data sets by providing for concurrent copy dumps. IMS requests backup through DFSMSdss's application programming interface, described in Appendix C, "Application Programming Interface," on page 303.

Backup-while-open serialization is applicable to HISAM, SHISAM, and index (primary and secondary) databases.

Backup as an open data set for IMS is triggered through a UIM request, rather than through a backup-while-open status or BWO(TYPEIMS) definition as CICS does.

Specifics of IMS backup-while-open support are outlined in Table 17.

Note: VALIDATE must be specified (directly or by default) to ensure that an IMS backup-while-open data set that is dumped while updates are being made can be successfully restored. VALIDATE allows DFSMSdss to validate and correct the data set during the dump process, or to end the dump (with an ADR943E message) if the data set is in an unrecoverable state.

DFSMSdss issues an ADR943E message if the NOVALIDATE parameter is used with an indexed VSAM data set defined as BWO(TYPEIMS).

Table 17. DFSMSdss Support for IMS Backup-While-Open Data Sets

Topic:	For IMS data sets:
Can non-VSAM data sets be dumped as open data sets?	Yes
Can non-indexed VSAM data sets be dumped as open data sets?	Yes
Must indexed VSAM data sets be specifically designated as being BWO-eligible to be dumped as open data sets?	No
How are indexed VSAM data sets designated as being eligible for update activity detection through a BWO counter?	By defining the data set as BWO(TYPEIMS)
How is backup as an open data set triggered?	Through a UIM request
How is bypassing of enqueues on SYSDSN and SYSVSAM managed?	Automatically

Data Integrity—Serialization

Attention: The user must be aware that COPY with DELETE, DUMP with DELETE, and RESTORE with REPLACE can cause irreparable damage to an IMS data set for the following reasons:

- The enqueue serialization obtained by COPY and DUMP are insufficient to ensure that the data set is not also being used by an IMS application in an environment where GRS (or equivalent) is not being used.
- COPY and DUMP do not provide any special handling for any data set defined as BWO(TYPEIMS).
- RESTORE does not provide any protection against reallocating or overwriting any IMS data set for which RESTORE is able to obtain an enqueue serialization on SYSDSN and SYSVSAM.

Appendix C. Application Programming Interface

This appendix provides information that you may need when you use the user interaction modules. General-use programming interface and associated guidance information is contained in this appendix. Programming interface information is also included and is explicitly marked.

Calling Block Structure

The parameter structure outlined below is shown in block form in Figure 8 on page 306.

ADRDSUU	The name of the DFSMSdss module main entry point.
OPTPTR	<p>The pointer to the option list and similar to OPTIONADDR described in <i>z/OS DFSMSdfp Utilities</i>. This parameter lets you specify processing options and must be specified even if the list is a null list. If you do not want to specify any parameters to DFSMSdss, this pointer must point to a halfword of binary zeros.</p> <p>You can invoke DFSMSdss using the standard application interface. This API allows you to specify in the options list those values that you can specify in the EXEC PARM field when you invoke DFSMSdss using JCL. You can also invoke DFSMSdss using the cross memory application interface. The values that you can specify in the options list include those values that can be specified in the EXEC PARM field when invoking DFSMSdss via JCL. Additionally, you can specify values that are specific to the cross memory application interface. For a list of the values that can be specified in the EXEC PARM field when invoking DFSMSdss via JCL, see “How to Control DFSMSdss through PARM Information in the EXEC Statement” on page 6. For a list of the values that are specific to the cross memory application interface, see “Using the Cross Memory Application Interface to Control DFSMSdss” on page 308.</p>
DDPTR	<p>The first field, called the <i>option-list length field</i>, is a halfword field specifying the length of the list (excluding the field itself) as a binary value. If you do not want to specify any options, set the option-list length field to binary zeros. In your JCL for the calling program, if you code parameters (PARM=value) on the EXEC statement, DFSMSdss does not recognize them unless OPTPTR points to them.</p> <p>The options must comply with the parameter syntax of the DFSMSdss EXEC parameter values. If you do not want to specify subsequent parameters, you can omit them from the list.</p> <p>The pointer to the DDNAME list and similar to DDNAMEADDR described in <i>z/OS DFSMSdfp Utilities</i>. The DDNAME list provides a way to specify alternate names for the SYSIN and SYSPRINT data sets. The DDNAME list is a variable length field made up of a halfword field followed by unseparated 8-character, left-justified (right padded with blanks) fields. Each 8-character field is reserved for a specific DDNAME. DFSMSdss uses only two of these fields: SYSIN and SYSPRINT. The other fields are provided as a standard</p>

Application Interface

implementation consistent with existing system utility invocation procedures, must be filled with binary zeros, and are ignored by DFSMSdss.

The first field, called the *DDNAME-list length field*, is a halfword field specifying the length of the list (excluding the field itself) as a binary value for the number 48.

If you do not want to specify an alternate DDNAME for the SYSIN or SYSPRINT data sets, set the DDNAME-list length field to the correct length and set all of the 8-character fields to binary zeros.

If you want to specify an alternate DDNAME for the SYSIN data set, specify this parameter and enter the alternate DDNAME in the fifth 8-character field. Otherwise, enter all binary zeros in the field.

If you want to specify an alternate DDNAME for the SYSPRINT data set, specify this parameter and enter the alternate DDNAME in the sixth 8-character field. Otherwise, enter all binary zeros in the field.

If you do not want to specify subsequent parameters, you can omit them from the list.

PAGEPTR

The pointer to the page-number list and similar to HDINGADDR described in *z/OS DFSMSdfp Utilities*. This list provides a way to specify the starting page number for system output on the SYSPRINT data set. The page-number list is a fixed-length 6-byte field made up of a halfword field followed by a 4-byte EBCDIC page count, which specifies the starting page number for DFSMSdss to use for the SYSPRINT data set. If a value is specified both here and in the OPTPTR list, this value is used, and the OPTPTR value is ignored.

The first field, called the *page-number-list length field*, is a halfword field specifying the length of the list (excluding the field itself) as a binary value. The length is usually 4.

If you do not want to specify a starting page number, set the page-number-list length field to binary zeros. If you want to specify a starting page number, you must specify this parameter and a 4-character page value. If the page number is specified (page-number-list length field is not binary zeros), DFSMSdss resets this field to the current page number upon completion of the present invocation. If the page number is not specified, this field is not changed by DFSMSdss.

If you do not want to specify subsequent parameters, you can omit them from the list.

UIMPTR

The pointer to the user interaction module (UIM) list. There is no comparable parameter described in *z/OS DFSMSdfp Utilities*. This parameter provides a way to specify the name or address of a vector-implemented exit module that is to interact with DFSMSdss for the various I/O and data set operations. The UIM list is of variable length and consists of a halfword field followed by a 4-byte address or 8-character left-justified (padded on the right with blanks) string field that specifies the address of the UIM entry point or the load module name (located through the normal LINKLIB structure) of the UIM. If you do not want to specify a UIM, do not specify this parameter.

The first field, called the *UIM-list length field*, is a halfword field specifying the length of the list (excluding the field itself) as a binary value. If you do not want to specify an option, set the option-list length field to binary zeros. If the address of the UIM is being passed, it specifies the length as 4, or as 8 if the name of the UIM is being passed.

If you want to specify a UIM, you must specify the name or address of the UIM in the field following the UIM-list length field. See “System Programming Information” on page 310 for more details on the use of this module.

UAPTR	The pointer to the user-area list. There is no comparable parameter in <i>z/OS DFSMSdfp Utilities</i> . This parameter provides a way to specify the address of an area to be passed to the UIM at each DFSMSdss exit point. The user-area list is of fixed length, consisting of a halfword field called the <i>user-area-list length field</i> , and a single 4-byte address that locates the user area starting address. If you use the user area, the length must be set to 4 and the address of the user area must be specified in a second field.
ASIDPTR	The pointer to the address-space-identifier list. There is no comparable parameter described in <i>z/OS DFSMSdfp Utilities</i> . This optional parameter, which is applicable only when you use ADRXMAIA instead of ADRDSSU, lets you specify an identifier for the address space that ADRXMAIA uses for the DFSMSdss program. The address-space-identifier list is of fixed length, consisting of a halfword field and a single 8-byte character field. The default value, if not set by the application program, is "DFSMSDSS." This value is set to "DSSBATCH" when JCL is used to invoke ADRXMAIA, but can be specified through the ASPACE= <i>name</i> PARM field. ADRXMAIA automatically creates the address space as needed. The identified address space can also be created by the operator START command by specifying an appropriately named PROCLIB member name.
PARAM	On the ATTACH and LINK macros, the keyword with which you specify the names of the pointers that are passed to DFSMSdss
VL	Indicates that the list is variable length. Both the ATTACH and LINK macros require specifying VL=1.

Application Interface

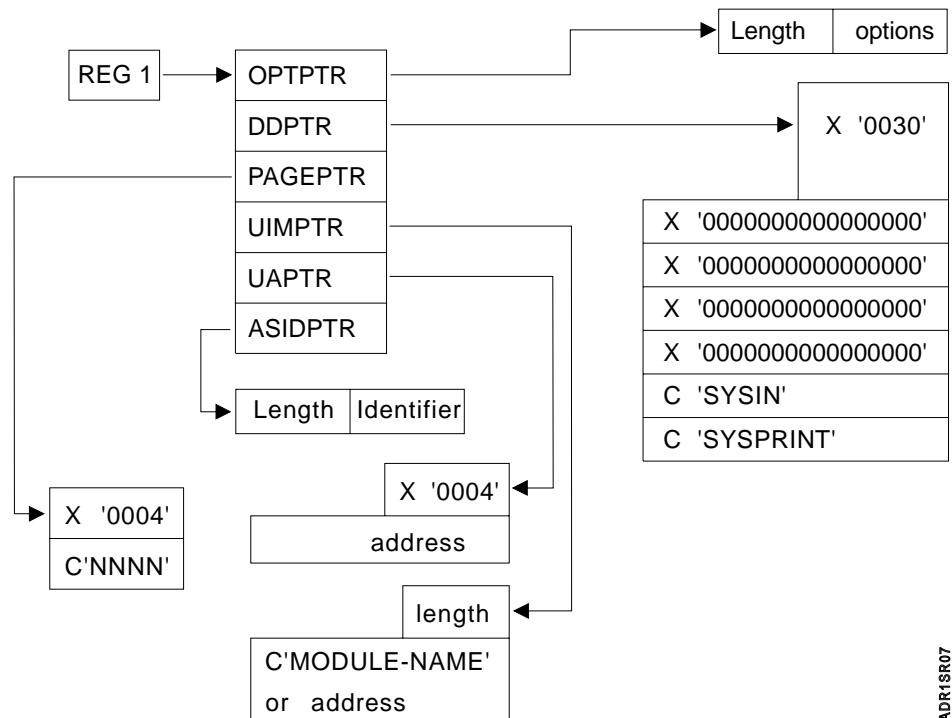


Figure 8. DFSMSdss Application Interface Structure

User Interactions

For user interactions to take place, the application must invoke DFSMSdss and must provide a pointer to a user interaction module (UIM) list. DFSMSdss can be invoked by any of the following system macros:

```
ATTACH EP=ADRDSU,PARAM=(OPTPTR,DDPTR,PAGEPTR,UIMPTR,UAPTR),VL=1
LINK EP=ADRDSU,PARAM=(OPTPTR,DDPTR,PAGEPTR,UIMPTR,UAPTR),VL=1
CALL (15),(OPTPTR,DDPTR,PAGEPTR,UIMPTR,UAPTR),VL
```

Optionally, to use the DFSMSdss Cross Memory Application Interface, use one of the following system macros:

```
ATTACH EP=ADRXMAIA,PARAM=(OPTPTR,DDPTR,PAGEPTR,UIMPTR,UAPTR),VL=1
LINK EP=ADRXMAIA,PARAM=(OPTPTR,DDPTR,PAGEPTR,UIMPTR,UAPTR),VL=1
CALL (15),(OPTPTR,DDPTR,PAGEPTR,UIMPTR,UAPTR),VL
```

Optionally, to use the DFSMSdss Cross Memory Application Interface and specify an 8-character Address Space name, use one of the following system macros:

```
ATTACH EP=ADRXMAIA,PARAM=(OPTPTR,DDPTR,PAGEPTR,UIMPTR,UAPTR,ASN PTR),VL=1
LINK EP=ADRXMAIA,PARAM=(OPTPTR,DDPTR,PAGEPTR,UIMPTR,UAPTR,ASN PTR),VL=1
CALL (15),(OPTPTR,DDPTR,PAGEPTR,UIMPTR,UAPTR,ASN PTR),VL
```

As shown in Figure 8 on page 306, a pointer to the UIM exit is passed in the parameter list (UIMPTR). User interactions involve:

- Record management
- Controlling processing of data sets
- Gathering statistics on DFSMSdss operations

When a UIM exit routine is specified, DFSMSdss processes normally, then at each point in the process (DFSMSdss exit points), the UIM exit routine is called conditionally to allow some types of user operations.

If a DFSMSdss subtask abnormally ends for any reason, it cannot make any further calls to the UIM, including the *function ending* call.

Notes:

1. DFSMSdss runs as an authorized problem program.
2. Any program invoking DFSMSdss must also be authorized and in a nonsupervisor state.
3. ADRXMAIA is the name of the DFSMSdss Cross-Memory Application Interface module main entry point. Use ADRXMAIA instead of ADRDSSU when DFSMSdss functions are to be executed in a specific address space.
4. ADRXMAIA runs as an authorized program and supports both supervisor-state callers and problem-state callers.

Related reading: For additional information about the various UIM exits, see Appendix D, “Examples of the Application Program with the User Interaction Module (UIM),” on page 345.

Cross-Memory Application Interface Overview

The DFSMSdss Cross Memory Application Interface is based on a client/server model. IBM products that make use of the DFSMSdss Cross Memory Application Interface include IMS Image Copy (IC2) and DFSMShsm. DFSMSdss also provides a JCL interface for this support.

DFSMSdss Cross Memory support is designed to be invoked using the LINK, CALL, or ATTACH commands. The support may also be invoked with JCL or through system invocations. To use the JCL interface, modify the JCL that is normally used for DFSMSdss batch mode processing to execute program ADRXMAIA instead of program ADRDSSU.

Example:

```
//S1 EXEC PGM=ADRMAIA,PARM='TYPRUN=NORUN'
      in place of
//S1 EXEC PGM=ADRDSU,PARM='TYPRUN=NORUN'
```

The DFSMSdss Cross Memory Application Interface support provides the capability for client applications to connect to and interact with a separate DFSMSdss server address space using a user-provided Interaction Module (UIM). DFSMSdss processing that is performed on behalf of a connected client application is controlled by one or more jobstep tasks that are executing in the server address space. The DFSMSdss server address space may be created by issuing a START command or may result from the ASCRE macro that is issued by the Cross Memory Application Interface support that is invoked by the client application.

After a DFSMSdss server address space begins execution, module ADRXMAIB is given control by the system, and directs server processing. The server processing that is performed on behalf of a client is referred to as a work thread, and it

Application Interface

executes under a unique jobstep task. The jobstep task invokes the ADRDSSU program for each work thread request that the client application makes. A server can concurrently process multiple client work threads for multiple client-connected address spaces. Each work thread is a separate instance of an ADRDSSU jobstep task. Multiple server address spaces may exist and be concurrently processing on behalf of multiple client address spaces.

A client application invokes module ADRXMAIA to utilize the DFSMSdss Cross Memory Application Interface support. When a client application invokes ADRXMAIA, DFSMSdss attempts a connection to the client-identified server address space. If the client-identified DFSMSdss server address space does not exist, the Cross Memory Application Interface invokes the ASCRE macro to create the server. If the client does not provide a server address space identifier, "DFSMSDSS" is used as the default server address space identifier. The JCL interface uses "DSSBATCH" as the default identifier unless the ASPACE parameter supplies it.

You can also create a DFSMSdss server address space using a START command that specifies an appropriately named member in SYS1.PROCLIB. The procedure must invoke module ADRXMAIB. The proclib member name should match the server identifier that is used by client applications or match batch jobs that use the JCL interface that will be connecting to the server that is created.

A DFSMSdss server address space that was created using the ASCRE macro goes into termination mode if there is no more work to do after all connected client applications terminate. There is a delay period prior to a server termination. Any connection request during this delay period causes the server to revert to processing mode to process the work threads that are associated with the connecting clients.

DFSMSdss server address spaces remain active as long as a client remains active unless the operator MODIFY command informs the server that it should stop processing after all current work threads are completed.

Example:

```
F DFMSDSS.DSSBATCH,STOP  
      or  
F DSSBATCH,STOP
```

A DFSMSdss server that is started with a START command does not enter termination mode when all work is completed. You must use the MODIFY command to stop it.

Using the Cross Memory Application Interface to Control DFSMSdss

You can control DFSMSdss through PARM Information in the EXEC Statement by using the Cross Memory Application Interface. The EXEC statement for DFSMSdss, when invoking DFSMSdss via the Cross Memory Application Interface, can contain PARM information that is used by the client and server, as well as the ADRDSSU program itself. The same values that can be specified in the EXEC PARM field in the JCL for ADRDSSU can also be specified in the EXEC PARM field for ADRXMAIA.

SRVRTIME=([minutes]{:seconds})

The SRVRTIME parameter specifies the amount of time the server address space should wait to shut down after the last piece of work has finished. If the Cross Memory Application Interface is invoked again, specifying this

particular server before the specified time has passed, this server will take the piece of work. When the time expires and no more pieces of work have been submitted, the server will shut down, and a subsequent invocation of the Cross Memory Application Interface specifying this server name will cause a new server to be created. The first invocation of the Cross Memory Application Interface for a particular server determines the length of time while that particular server is running. Subsequent invocations to the same server while it is running will not change the time even if the SRVRTIME parm is specified on those later invocations.

minutes

Specifies the maximum number of minutes the server will wait after the last piece of work is finished before shutting down. The minutes must be a number from 0 through 357912 (248.55 days).

seconds

Specifies the maximum number of seconds the server will wait after the last piece of work is finished before shutting down. The seconds must be a number from 0 through 59.

The following examples demonstrate the affect on the system when you designate a SRVRTIME value:

- SRVRTIME=(1:30) - The server will wait for 1 minute 30 seconds before shutting down.
- SRVRTIME=(24:00) - The server will wait for 24 minutes before shutting down.
- SRVRTIME=(0:25) or SRVRTIME=(:25) - The server will wait for 25 seconds before shutting down.
- SRVRTIME=(0:00) - The server will shutdown immediately after the last piece of work is finished.

Restrictions: If the SRVRTIME parameter is not specified, the time before the server shuts down is determined as follows:

- If the Cross Memory Application Interface is invoked from JCL, but the ASPACE parameter is not specified, the server will shut down after 4 minutes.
- If the Cross Memory Application Interface is invoked via JCL and the ASPACE parameter is specified, the server will wait 1 minute.
- If the Cross Memory Application Interface is invoked using the LINK, CALL, or ATTACH commands, but an ASPACE name wasn't provided in the ASIDPTR field, the server will wait 8 minutes.
- If the Cross Memory Application Interface is invoked using the LINK, CALL, or ATTACH commands and an ASPACE name was provided in the ASIDPTR field, the server will wait 1 minute.

ASPACE=*id*

Where *id* is determined by the installation to identify which server to use for processing DFSMSdss SYSIN command streams. ASPACE=AFFINITY is a special use. It causes ADRDSSU to be used within the client address space instead of running in a separate address space.

NOTE: ASPACE is only available when using JCL to invoke DFSMSdss via the cross memory application interface.

The following example demonstrates how you might designate the use of the ASPACE parameter:

```
//S1 EXEC PGM=ADRXMAIA,PARM='ASPACE=BACKUP'
```

Application Interface

SNAPX=* | **nn** | **(nn,nn[,nn])**

The SNAPX parameter can be used to debug your user interaction module. The SNAPX parameter can be specified on JCL and system invocations of the Cross Memory Application Interface (API). Specifying the SNAPX parameter requests that ADRXMAIA write the contents of the EIDB and EIRECPTR to SYSPRINT whenever the specified exit is called. When an application writes its own messages to SYSPRINT, ADRXMAIA calls exit 2 of the user interaction module for that application.

When specifying the SNAPX parameter, you can specify:

- One exit to snap. For example, you can specify SNAPX=12 to see the contents of exit 12.
- More than one exit to snap. For example, you can specify SNAPX=(1,6,2) to see the contents of exits 1,2, and 6.
- All command processing exits. For example, you can specify SNAPX=*.

The following example demonstrates how you might designate the use of the SNAPX parameter:

```
//S1 EXEC PGM=ADRUMAIA,PARM='SNAPX=(21,22,23)'
```

System Programming Information

Programming Interface information

Some bit definitions in the installation options control block (ADRUFO) permit DFSMSdss to communicate with the installation options exit routine. When the installation options exit does not want a function to be scheduled, it returns a code of 8. For the UIM to do this requires a bit definition of UFSTOP to be set in the ADRUFO. If a SYSIN or SYSPRINT data set is not to be allocated or all SYSIN/SYSPRINT is to be handled in storage, UFSYSIN and UFSYSPR specify this to DFSMSdss.

If the application allows DFSMSdss to handle SYSPRINT by not setting UFSYSPR, then the application can not write to SYSPRINT directly. The application can only insert SYSPRINT records at UIM exit points 2 or 10. If the application chooses to handle SYSPRINT by setting UFSYSPR, then the application is responsible for printing DFSMSdss messages from UIM exit points 2 or 10.

Three additional bits determine allowances within the UIM interaction: UFAINV, UFUIMAL and UFUIMCH. Two additional bits indicate the existence or absence of an input restore/copydump data set or an output dump/copydump data set: UFNOIN and UFNOOUT respectively. Refer to *z/OS DFSMS Installation Exits* for more information about the installation options exit routine.

End of Programming Interface information

When DFSMSdss has been invoked by using the Application Interface and the UIM has been specified and is to be called, the following information is passed to the UIM on every exit call:

- Register 1, which points to the interface parameter list pointer.
- The interface parameter list pointer, which points to the DFSMSdss exit identification block. See “ADREID0 Data Area” on page 331 for a detailed description of this block. This list consists of:
 - A halfword field specifying the length of the remainder of the list.

- The remainder of the list that is mapped by the macro ADREID0. See “Exit Identification Block” below for the information contained in this block.

Upon return from the UIM, the user return code field is examined to determine the disposition of the current I/O record or data set (within the limits allowed to the exit).

Application Interface Blocks

The parameter structure described in Figure 8 on page 306 can be viewed in block form as shown in Figure 9.

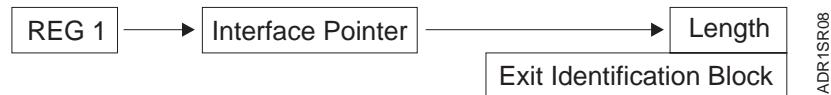


Figure 9. DFSMSdss Exit Interface Structure

Exit Identification Block

The exit identification block is passed to the user interaction module every time DFSMSdss gives control to it. Each field is described below, but see “ADREID0 Data Area” on page 331 for the formal declarations.

Control Block Eye-Catcher

A 4-character string field that is supplied by DFSMSdss. It contains the character string *EIDB* and can aid in locating the control block when you are viewing a storage dump during DEBUG processing.

TASK-ID

A fullword binary field that is supplied by DFSMSdss. The number contained in this field is assigned by DFSMSdss to each function command statement submitted in SYSIN, whether obtained from the data set or from the UIM. This number is binary zero when DFSMSdss is calling the UIM for a function that is not related to a user command statement. Each command is numbered in sequence. When a task is scheduled to process this command, all messages associated with this task and all calls to the user interaction module for this task are accompanied by this unique number. In this way, the UIM can identify which task is being processed and what function is associated with that task.

User Exit Allowance

A fullword binary field supplied by DFSMSdss. The 32 bits defined in this field can be used as flags to determine what actions the UIM can perform with respect to the record presented. The following actions are conditionally allowed:

- View and conditionally override the installation options.
- Insert data prior to current record.
- Replace the current data record with an exit-record supplied.
- Delete the current data record.
- Modify the current data record.
- Disconnect the exit from further interaction.
- Recognize when a disallowed option has been attempted.
- End processing of the data set.
- End processing of the task.

DFSMSdss Processing Option

A halfword binary field that is supplied by DFSMSdss. The number

Application Interface

contained in this field can be used by the user interaction module (UIM) to vector branch to the appropriate processing routine for the record or data set being presented to the exit.

At appropriate locations in the DFSMSdss processing modules, the user interaction module is considered for receiving control. These locations are referred to as DFSMSdss exit points.

User Return Code

A halfword binary field that is supplied by the UIM. This return code identifies the action expected by the exit on the record or data set being presented to the exit. DFSMSdss examines this field when control returns from the UIM. Not all the following return codes are allowed at any given exit call. If a disallowed code is returned from the UIM, DFSMSdss passes the EIDB back to the UIM with the EIXERR flag set. This allows the UIM one chance to correct the option. If a disallowed code is returned again, it is ignored by DFSMSdss and the record is processed as though the exit had returned the code zero (0). If a valid code is returned, DFSMSdss processing is the same as if the valid code had been passed back on the initial UIM invocation. This sequence is followed for any subsequent incorrect return codes. The return codes that are allowed at any given exit call are specified in the EIXALLOW field.

- The meaning of each return code for those exits presenting records to the UIM (Eoptions 01 to 26) is:
 - 0** The record is processed as would normally have happened if there had not been a UIM. The original record is not changed in any way by the exit.
 - 4** The record was replaced by the exit. The new record must be placed in the area pointed to by the original record pointer field and its length stored in the original record length field.
 - 8** The record is to be inserted. The address of the new record must be stored in the original record pointer field or the new record must be stored in the area pointed to by the original record pointer field and its length must be stored in the original record length field. When this exit is next called, the original record is presented again.
 - 12** The record is to be deleted. The record presented to the UIM is ignored in the processing, thus deleting it.
 - 16** The record was modified by the exit. This return code is the only one allowing the original record to be altered and then to be processed by DFSMSdss. Any changes made to the record must be logically correct because DFSMSdss cannot assure the validity of these changes.

Notes:

1. You cannot change the record length when you modify the record (as contrasted with reason code 4).
2. If the record being processed is the installation options record (ADRUFO) and any values have been changed, return code 16 must be returned or the changes are ignored.

- 20** The record is to be processed as though a return code zero (0) had been given, but this particular DFSMSdss exit point is no longer called from the current functional task, although it might be called from others. You must be cautious in using this code

because multiple record types use the same DFSMSdss exit point. It would be better for the UIM to inspect each record type and only give a return code zero (0) when a record is not of interest to the exit.

- 24 The record is to be processed but at all future visits of DFSMSdss at this exit point, only user statistical records are to be presented to the UIM.
- 28 The notification-of-response return code. The WTOR has been handled by the UIM and the UIM has supplied the proper response from the WTOR in the area pointed to by the original record pointer. The interface allows you to handle all WTOR processing in the UIM exit routine and to supply the required response to DFSMSdss in lieu of DFSMSdss's issuing the WTOR itself. Note that the UIM must set the original record length to the proper value.
- 32 DFSMSdss ends the current functional task and issues message ADR356E.
- The meaning of each return code for those exits controlling data set processing (E'options 21, 22, 23, and 26) is:
 - 0 The data set is processed as would normally have happened if there had not been a UIM. Data set processing is not altered in any way.
 - 16 This return code is valid from exit 22 only. It indicates to DFSMSdss to examine the bypass processing flags and to modify processing of the data set according to which flags are on. These bypass flags are only examined by DFSMSdss if a return code of 16 is returned. See "Bypass Verification Exit (Eoption 22)" on page 323 for more details about these flags.
 - 20 The record is to be processed as though a return code zero (0) had been given, but this particular DFSMSdss exit point is no longer called from the current functional task, although it might be called from others. You must be cautious in using this code because multiple record types use the same DFSMSdss exit point. It would be better for the UIM to inspect each record type and only give a return code zero (0) when a record is not of interest to the exit.
 - 32 DFSMSdss ends the current functional task and issues message ADR356E.
 - 36 DFSMSdss ends processing of the data set named in the parameter passed to the exit. Processing continues with the next data set (if any). If Exit 23 sets this return code after the data set has already been processed, then DFSMSdss does not undo the processing, but deletes the data set from the successfully-processed message list (if applicable) and includes it in the unsuccessfully-processed message list even though the data set may have been successfully processed up to that point

Record Area Length

A fullword binary field that is supplied by DFSMSdss. The number contained in this field represents the total length of the area in which the original record is stored. The number can be used by the UIM to verify

Application Interface

that replacement and inserted records fit in the provided buffer area pointed to by the original record pointer field.

Original Record Length

A fullword binary field that is supplied by DFSMSdss but can be changed by the UIM. The number contained in this field represents the total length of the record pointed to by the original record pointer. The length includes just the length of the record pointed to by the original record pointer field and does not include the length of this field itself. The value of this field is zero when the UIM is called to supply in-storage SYSIN data.

Original Record Pointer

A fullword address field that is supplied by DFSMSdss but can be changed by the UIM. This field contains the address of the record being passed to the UIM on this call. This address normally is not changed by the exit unless an insertion is being used. Refer to the proper return code descriptions for the effect on this field. The value of this field is zero when the UIM is called to supply in-storage SYSIN data. The location of the original record, above or below the 16-megabyte (MB) virtual storage line, is controlled by using the installation options exit. If this record is to be above 16MB, the UIM has to run in 31-bit addressing mode in order to address the record.

User Area Pointer

A fullword address field that is supplied by the application program and is maintained by DFSMSdss. This field contains the address of the user data/work area that was supplied by the application program as a communications area for UIM internal process controls. DFSMSdss saves this pointer and supplies it in the EIUSEPTR field of the exit identification block on each call to the UIM. If any UIM exit changes the user area pointer, DFSMSdss presents the updated pointer on subsequent calls to the UIM.

Each DFSMSdss functional task keeps its own copy of the user area pointer. If the pointer is changed by the UIM for one task, it is not changed for any other task. At the beginning of each task, the user area pointer is the one passed to DFSMSdss by the application program.

DDNAME/VOLID Pointer

A fullword address field supplied by DFSMSdss. It contains the address of an area containing the DDNAME of the output dump data set, left-justified in an 8-byte area, followed by a 6-byte area containing the volume serial number of the volume containing the dump data set, also left-justified. This pointer is valid only for the full-volume dump exit, EIOP06.

Cross-Memory Application Interface Restrictions

Use SDUMP and SMSGCNT in place of ABEND and AMSGCNT when the Cross-Memory Application Interface is used. Although ABEND and AMSGCNT are supported by DFSMSdss, no dump is printed because DD allocations (SYSABEND, SYSUDUMP) in the application address space are unavailable to the address space producing the abend.

Serialization may operate differently, as application address space DD allocations are unavailable to the address space executing the DFSMSdss functions. Evaluate your serialization requirements when considering the use of the Cross-Memory Application Interface.

When multiple ADRDSSU tasks are executing in a DFSMSdss server address space, and identical DDNAMEs are passed in the SYSIN stream, allocation errors (such as MSGIKJ56246I – “file in use”) can result. To avoid this possibility, use 8-character DDNAMEs or leave enough room after the DDNAME to allow ddname replacement. The Cross Memory Application Interface replaces common DDNAMEs passed in the SYSIN stream with unique 8-character, system-generated DDNAMEs whenever possible.

If you use the DFSMSdss Cross Memory Application Interface for logical COPY or logical DUMP and RESTORE of HFS and zFS type data sets, you must define a DFSMSdss user ID for the DFSMSdss server address spaces to be able to access the HFS and zFS data sets. The DFSMSdss user ID must be set up for the z/OS UNIX System Services (z/OS UNIX) access as follows:

- The default group for the DFSMSdss user ID must have a z/OS UNIX segment defined and a group ID associated with it.
- The home directory should be the root file system.
- The DFSMSdss user ID must be defined as a superuser (UID 0). Use the following RACF commands to make this assignment:

```
ADDGROUP OMVSGRP OMVS(GID(1))
ADDUSER DFSMSDSS DFLTGRP(OMVSGRP) OMVS(UID(0) HOME('/'))
```

Specifying multiple commands (DUMP, RESTORE, or COPYDUMP) that process DFSMSdss dump data sets on the same tape volume is not supported when invoking DFSMSdss using the Cross-Memory Application Interface. This restriction applies whether or not the PARALLEL command is specified. To process multiple DFSMSdss dump data sets on the same tape volume using the Cross-Memory Application Interface, use multiple invocations of DFSMSdss with one command per invocation and ensure that the operations are executed serially.

Related reading: For additional information about z/OS UNIX, see *z/OS UNIX System Services Planning*, GA22-7800.

User Interaction Module Exit Option Descriptions

Following are the descriptions for all exit points available with DFSMSdss.

Function Startup (Eioption 00)

This exit point is called during DFSMSdss initialization and again during function initialization (such as a dump or restore operation). The EIRECPTR points to the EIREC00 data area. The EIREC00 data area contains a field, EI00SBPL, which the application may use to return a subpool number that is to be shared between all DFSMSdss tasks. EI00SBPL is only valid when EITSKID is zero.

Programming Interface information

The application can set EI00NOLK to request that DFSMSdss bypass ADRLOCK ENQUEUE or DEQUEUE processing during full-volume and tracks RESTORE operations. The application must be able to resolve any deadlock conditions that result from the request to bypass ADRLOCK ENQUEUE or DEQUEUE processing.

End of Programming Interface information

Programming Interface information

The application can also set EI00SENQ to request that VTOC serialization be held only for the duration of VTOC access. This flag is only valid for full volume dump operations and when EITSKID is nonzero. This flag provides a function similar to that provided by the Enqueue Installation Exit Routine (seez/OS DFSMS Installation Exits). The VTOC serialization is released after VTOC access is complete if either the EI00SENQ flag is set or the Enqueue Installation Exit Routine requests it.

End of Programming Interface information

Programming Interface information

The application can also set EI00NENQ to request that DFSMSdss not reserve on VTOC of the source volume. This flag is only valid for physical COPY and DUMP operations during function startup (for example, when EITSKID is nonzero).

End of Programming Interface information

Programming Interface information

The application can also set EI00BSEC flag to request that RACF verification and all other data set security checking is bypassed. This flag is only valid for full/tracks COPY, full/tracks DUMP, and full/tracks RESTORE operations during function startup (i.e., when EITSKID is nonzero).

End of Programming Interface information

Programming Interface information

DFSMSdss provides the source VOLSER to the UIM. This field is only valid for full/tracks COPY, full/tracks DUMP, and full/tracks RESTORE during function startup (i.e., when EITSKID is nonzero). For RESTORE, this field contains the volume serial number of the input volume where the dump data set resides.

End of Programming Interface information

The valid return codes for function startup are:

- 0** Continue normal processing
- 16** Record modified

Programming Interface information

- 20** Inhibit all UIM calls—Not valid when EITSKID is zero and either UFSYSIN or UFSYSPR was set in the installation options exit, or EITSKID is nonzero and either UFNOIN or UFNOOUT was set in the installation options exit.

End of Programming Interface information

- 32** End processing. This return code is only valid for a full dump and a full restore operation.

Reading SYSIN Record (Eioption 01)

This exit point is called after DFSMSdss reads a SYSIN record. You can replace, insert, delete, or modify a SYSIN record at this exit point. The EIRECPTR points to the SYSIN record. The valid return codes are:

- 0** Continue normal processing
- 4** Record replaced
- 8** Insert record
- 12** Delete record
- 16** Record modified
- 20** Disconnect exit

Printing SYSPRINT Record (Eioption 02)

This exit point is called when DFSMSdss is ready to print a SYSPRINT record. You can replace, insert, delete, or modify a SYSPRINT record at this exit point. The EIRECPTR points to the SYSPRINT record. Task-related SYSPRINT records are presented in task order unless the UIM requested that there be no SYSPRINT (see “System Programming Information” on page 310).

The valid return codes for printing SYSPRINT records are:

- 0** Continue normal processing
- 4** Record replaced
- 8** Insert record
- 12** Delete record
- 16** Record modified
- 20** Disconnect exit

Guidelines:

- You can only modify or delete a page header record during the print operation.
- EIXNTER is set ON when the following conditions exist:
 - You have specified the TOL(IOERR) keyword.
 - DFSMSdss issues e-type messages with the exception of messages ADR324E, ADR347E, and ADR348E.
 - DFSMSdss issues t-type messages; the TOL(IOERR) keyword does not have to be specified.
- EIXNTER is set OFF when the following conditions exist:
 - An I-type or W-type message is issued by DFSMSdss.
 - DFSMSdss issues the ADR324E, ADR347E, and ADR348E (all or some combination) messages, and the TOL(IOERR) keyword is specified.
- EIXNTER is unchanged when DFSMSdss issues a nonprefixed message.

Reading Physical Tape Record (Eioption 03)

This exit point is called when DFSMSdss has read a record from a data set that has been dumped (on tape or DASD) by using the DUMP command. The EIRECPTR points to the tape record. The valid return codes are:

Application Interface

- 0** Continue normal processing

Programming Interface information

- 8** Insert record. This return code is only valid when UFNOIN is set in the installation options exit

End of Programming Interface information

Programming Interface information

- 20** Disconnect exit. This return code is only valid when UFNOIN is not set in the user installation options exit

End of Programming Interface information

- 32** End function

Reading Logical Tape Record (Eioption 04)

This exit point is called when DFSMSdss has read a record from a data set which was created with a DUMP command and dumped on tape or DASD. The EIRECPTR points to the tape record. The valid return codes are:

- 0** Continue normal processing
- 8** Insert record
- 20** Disconnect exit
- 24** Select user statistics records
- 32** End function.

Note: If code 24 is returned, DFSMSdss only calls this exit point when DFSMSdss processes user statistics records.

Writing Logical Tape Record (Eioption 05)

This exit point is called when DFSMSdss writes a logical record to a dump data set, on tape, or on DASD, while performing a dump operation. If you insert a record at this exit point, DFSMSdss marks it as a statistics record, not a data record. The EIRECPTR points to the tape record. The valid return codes are:

- 0** Continue normal processing
- 8** Insert record
- 20** Disconnect exit
- 32** End function.

Writing Physical Tape Record (Eioption 06)

This exit point is called when DFSMSdss writes a physical record to a dump data set, on tape, or on DASD. The EIRECPTR points to the tape record and the EIDDID points to EIDDINFO. The valid return codes are:

- 0** Continue normal processing

Programming Interface information

- 12** Delete record. This return code is only valid when UFNOOUT is set in the installation options exit

End of Programming Interface information

Programming Interface information

- 20** Disconnect exit. This return code is only valid when UFNOOUT is not set in the user installation options exit

End of Programming Interface information

- 32** End function.

Reading Disk Track (Eioption 07)

This exit point is called when DFSMSdss has read a track from DASD. The EIRECPTR points to the track buffer. The first 64 bytes (X'40') of the track buffer are IBM internal information which is followed by the ADRTAPB area, the DTTRK area, and the track image. The valid return codes are:

- 0** Continue normal processing
- 20** Disconnect exit
- 32** End function. This return code is only valid for a full dump

Related reading: For additional information about ADRTAPB and DTTRK, see the *z/OS DFSMSdss Storage Administration Guide*.

Writing Disk Track (Eioption 08)

This exit point is called when DFSMSdss is ready to write a track to DASD. The EIRECPTR points to the track buffer. The valid return codes are:

- 0** Continue normal processing
- 20** Disconnect exit
- 32** End function. This return code is only valid for a full restore operation

Reading Utility SYSPRINT (Eioption 09)

This exit point is called when DFSMSdss is reading output from an attached utility. You can replace, insert, delete, or modify a utility SYSPRINT record at this exit point. The EIRECPTR points to the Utility SYSPRINT record. The valid return codes are:

- 0** Continue normal processing
- 4** Record replaced
- 8** Insert record
- 12** Delete record
- 16** Record modified
- 20** Disconnect exit

Writing SYSPRINT Record (Eioption 10)

This exit point is called when DFSMSdss is ready to write a SYSPRINT record. You can replace, insert, delete, or modify the SYSPRINT record at this exit point. The

Application Interface

EIRECPTR points to the SYSPRINT record. Task-related SYSPRINT records are not presented in task order. The valid return codes are:

- 0** Continue normal processing
- 4** Record replaced
- 8** Insert record
- 12** Delete record
- 16** Record modified
- 20** Disconnect exit

Writing WTO Message (Eioption 11)

This exit point is called when DFSMSdss is ready to write a WTO message. You can replace, insert, delete, or modify the WTO message at this exit point. The EIRECPTR points to the WTO message. The valid return codes are:

- 0** Continue normal processing
- 4** Record replaced
- 8** Insert record
- 12** Delete record
- 16** Record modified
- 20** Disconnect exit

Writing WTOR Message (Eioption 12)

This exit point is called when DFSMSdss is ready to write a WTOR message (that is, ADR369D, ADR345D, and ADR371D). You can insert or modify the WTOR message at this exit point. You can also return the response to DFSMSdss with return code 28. The EIRECPTR points to the WTOR message. The valid return codes are:

- 0** Continue normal processing
- 8** Insert record
- 16** Record modified
- 20** Disconnect exit
- 28** WTOR response

Presenting ADRUFO Record (Eioption 13)

Programming Interface information

This exit point is called when DFSMSdss is ready to set up the installation options specified in the ADRUFO control block. You can modify the control block to override any options that have been specified thus far. You must use return code 16 for DFSMSdss to recognize the new options. The EIRECPTR points to ADRUFO. The valid return codes are:

- 0** Continue normal processing
- 16** Record modified
- 32** End function. This return code is only valid for a full dump and a full restore operation.

End of Programming Interface information

Function Ending (Eioption 14)

This exit point is called when DFSMSdss is ready to end the task. The EIRECPTR points to the last message with highest nonzero return code. The record contains the DFSMSdss return code and the DFSMSdss message (message number and message type). The valid return code is:

- 0** Continue normal processing

Presenting WTOR Response (Eioption 15)

This exit point is called when the operator has responded to the WTOR (either a DFSMSdss WTOR or a user inserted WTOR, see exit point 12). You can only examine the response at this exit point. If you would like to change the response, you must use exit point 12, change the response, and use a return code of 28 to DFSMSdss. The EIRECPTR points to the response. The valid return code is:

- 0** Continue normal processing

OPEN/EOV Tape Volume Security and Verification Exit (Eioption 16)

This exit point is called when DFSMSdss is ready to open a tape. You can bypass password and expiration date checking for tape volumes or reject the volume and request a scratch tape with this exit by placing a return code in the first word of the record pointed to by EIRECPTR. DFSMSdss passes this return code to OPEN/EOV. The EIRECPTR points to the parameter list described by the IECHOEVSE macro.

The valid return codes are:

- 0** Continue normal processing
- 16** Record modified
- 20** Disconnect exit
- 32** End function. This return code is only valid for a full dump and a full restore operation

Related reading: For additional information about OPEN/EOV, see *z/OS DFSMS Using Data Sets*.

OPEN/EOV Nonspecific Tape Volume Mount (Eioption 17)

This exit point is called when a nonspecific tape is passed to DFSMSdss. The EIRECPTR points to the DCB exit parameter list. You can specify a specific volume serial with this exit by placing the volume serial in the first 6 bytes of the record pointed to by EIRECPTR. DFSMSdss passes this volume serial and return a code of 4 to OPEN/EOV informing it to use the specified volume serial. The EIRECPTR points to the parameter list described by the IECHOENTE macro.

Note: DEFER must be specified in the JCL for this exit to be called.

The valid return codes are:

- 0** Continue normal processing
- 16** Record modified

Application Interface

- 20** Disconnect exit
- 32** End function. This return code is only valid for a full dump and a full restore operation

Insert Logical VSAM Record During Restore (Eioption 18)

This exit point is called during a logical restore operation of a VSAM KSDS using record-level I/O. You can replace or modify a record at this exit point. The EIRECPTR points to the logical record that DFSMSdss is preparing to write to the data set. The valid return codes are:

- 0** Continue normal processing
- 4** Record replaced
- 16** Record modified
- 20** Disconnect exit

Related reading: For additional information about Eioption 18, see *z/OS DFSMS Using Data Sets*.

Output Tape I/O Error (Eioption 19)

This exit point is called during a dump when a tape has a permanent I/O error. The EIRECPTR points to EIDDINFO. The valid return codes are:

- 0** Continue normal processing
- 20** Disconnect exit
- 32** End function

Volume Notification (Eioption 20)

During physical data set restore operations, this exit is called to provide information about the data set being allocated. This information includes:

- Cluster/data set name (for RENAME and RENAMEUNCONDITIONAL keywords, the new data set name is presented)
- An indication of whether the data set is VSAM or non-VSAM
- Number of data and index components (VSAM only)
- For each component or data set:
 - The number of volumes it resides on
 - The volume serial number of each volume
 - An RBA token for each volume.

EIRECPTR points to EIREC20. EI20DA@ and EI20IX@ point to EI20DSI.

Note: SMS-managed multivolume non-VSAM data sets have an RBA token only for the first volume. Non-SMS-managed data sets never have an RBA token.

The valid return codes are:

- 0** Continue normal processing
- 20** Disconnect exit
- 32** End function. This return code is only valid for full or physical data set dump and full or physical data set restore operations

Data Set Verification (Eioption 21)

This exit lets the UIM end logical copy, dump, and restore processing for individual data sets. This exit is given control through the UIM at the start of logical copy, dump, and restore processing for each data set. DFSMSdss provides the UIM with the data set name through the EIREC21 structure within the exit identification block, ADREID0. EIREC21 is shown in Table 18 on page 331. The name provided is the original data set name prior to rename processing, if any, for copy and restore functions. The UIM returns to DFSMSdss with a return code in the exit identification block. The valid return codes are:

- 0** Continue normal processing
- 20** Disconnect exit
- 32** End function
- 36** End data set

DFSMSdss continues processing with the next data set, if any. If the user specifies the SPHERE keyword and processing for a base cluster is ended through this exit, processing for all AIXs related to the base cluster is also ended. If the user specifies the SPHERE keyword and the UIM attempts to end processing for an AIX that is related to a base cluster that was selected for processing, DFSMSdss invokes the UIM again. If the UIM attempts to end processing for the AIX again, DFSMSdss issues warning message ADR770W and ignores the request. If the user does not specify the SPHERE keyword, AIXs can be ended even if they are related to base clusters selected for processing. Ending a base cluster without the SPHERE keyword does not cause any related AIXs to be ended.

Note: See the “Note for Eioptions 21, 22, and 23” at the end of the description for eioption 23.

Bypass Verification Exit (Eioption 22)

This exit lets a user force DFSMSdss bypass serialization and security verification during logical copy, dump, and restore processing for individual data sets. It also lets a user turn on a tolerate-migrated-volser indicator that forces DFSMSdss to restore a data set with a migrated volser. This exit is given control through the UIM at the start of logical copy, dump, and restore processing for each data set. DFSMSdss provides the UIM with the data set name through the EIREC22 structure within the exit identification block, ADREID0. EIREC22 is shown in Table 18 on page 331. The data set name provided is the original name prior to rename processing, if any, for COPY or RESTORE. The UIM returns to DFSMSdss with a return code in the exit identification block. The valid return codes are:

- 0** Continue normal processing
- 16** Bypass one or more of the following:
 - If the Reset indicator (EI22RSET) is set on by the UIM for a VSAM data set during a logical data set dump, DFSMSdss resets the data-set-changed flag in the VTOC if the data set is successfully serialized and processed. DFSMSdss will not reset the data-set-changed flag for data sets dumped with RLS access and BWO data sets, even if EI22RSET is set on by the UIM.
 - If the DB2 Source Bypass indicator (EI22DB2) is set on by the UIM during a logical data set copy operation, and RENAMEU is specified for a DB2 VSAM linear data set, then DFSMSdss does not verify that the old component duplicates the DB2 naming convention when it derives the

new component name. Alternatively, if the EI22DB2 bit is set on, DFSMSdss generates a DB2 component name for the target component—provided the new cluster name (specified in the RENAMEU parameter) matches the DB2 naming convention.

- If the Shared SYSDSN ENQ indicator (EI22SSYS) is set on by the UIM for a VSAM data set during logical data set dump, DFSMSdss attempts to obtain a shared SYSDSN enqueue. If EI22SSYS is not set on, then the type of SYSDSN enqueue (shared or exclusive) that DFSMSdss attempts to obtain depends upon whether or not the SHARE keyword was specified.
- If the Mark-As-Recovery-Required indicator (EI22RRB) is set on by the UIM during logical restore, DFSMSdss marks the target data set as recovery required, provided that the target data set is SMS-managed.
- If the Log Information Passed indicator (EI22LINF) is set on by the UIM during a logical restore, then DFSMSdss uses the log parameter (EI22LPRM) and the log stream ID (EI22LSID) as the log information for the target data set, provided that the target data set is SMS-managed. If EI22LINF is not set, or the target data set is not SMS-managed, then the contents of EI22LPRM and EI22LSID are ignored.
- If the BWO_ALLOWED Passed indicator (EI22BWOP) is set on by the UIM during a logical restore, the DFSMSdss uses the BWO_ALLOWED field (EI22BWOA) for the target data set, provided the target data set is SMS-managed. If EI22BWOP is not set or the target data set is not SMS-managed, then the contents of EI22BWOA are ignored.
- Serialization is bypassed if the UIM exit turns on the Bypass Serialization indicator, EI22BSER. Serialization is bypassed for the source data set for copy and dump processing, and for the target data set for copy and restore processing. DFSMSdss assumes that all necessary serialization has been performed by the invoker of DFSMSdss, but does not ensure that this is true. DFSMSdss performs normal serialization if the UIM exit does not turn on the bypass serialization indicator.
DFSMSdss does not delete or uncatalog data sets that were bypassed for serialization even if the DELETE or UNCAT keywords are specified. If a preallocated data set is not large enough during a data set restore operation and the bypass serialization indicator is on, DFSMSdss does not scratch and reallocate that target data set and the restore operation fails.
- RACF verification and all other data set security checking is bypassed if the UIM exit turns on the Bypass RACF indicator, EI22BSEC. If EI22BSEC is set on, DFSMSdss checks to ensure that the application program is authorized to bypass RACF and security processing. The application program is authorized to bypass RACF and security processing if NOPASS was specified on the PPT statement of the SCHEDxx parmlib member. If the EI22BSEC indicator is on and NOPASS was specified, RACF verification and all other data set security processing including password checking is bypassed. DFSMSdss does not do any RACF authorization checks. DFSMSdss does not create any RACF profiles for the copied or restored target data set. DFSMSdss assumes that all necessary RACF authorization and security checking has already been performed by the invoker of DFSMSdss. For a copy or restore operation, RACF profiles are not created for the target data set. If the user turns on the bypass RACF indicator and PASS was specified on the PPT statement of the SCHEDxx parmlib member, DFSMSdss sets the error flag, EIXERR, on in the exit identification block and invokes the

UIM again. If the user sets the bypass RACF indicator again, DFSMSdss issues error message ADR772W and processing continues as normal. DFSMSdss still allows serialization to be bypassed and tolerates migrated volume serial number processing if those indicators are set.

DFSMSdss performs normal RACF and security processing if the UIM exit does not turn on the bypass RACF indicator.

- If the Tolerate-Migrated-Volser indicator is set on by the UIM, DFSMSdss takes special action to support the restoration of a non-VSAM data set with a volume serial number of MIGRAT for a logical data set restore operation in an SMS-managed environment or when the CATALOG option has been specified. DFSMSdss ignores the EI22BMIG indicator for copy and dump operations and VSAM data sets.

When DFSMSdss is restoring a non-VSAM data set and the Tolerate-Migrated-Volser indicator is set on, a catalog LOCATE is issued to determine the status of the data set being restored. Based on the result of that LOCATE, different actions are taken.

No catalog entry is found:

The data set being restored is not considered to be migrated. A normal DFSMSdss logical restore is performed. The Tolerate-Migrated-Volser indicator is ignored.

A catalog entry is found but the VOLSER is not MIGRAT:

The data set being restored is not considered to be migrated. Normal processing continues as though the indicator had not been set.

A catalog entry is found and the VOLSER is MIGRAT:

The data set is migrated and requires special processing. Instead of cataloging the data set after allocating it, DFSMSdss alters the existing MIGRAT volume serial number entry to the actual volume the data set was restored to. Once the catalog entry has been changed from MIGRAT to the new data set's volume serial numbers, the restore continues as normal.

Note: Because the migrated data set is not recalled during this restore, the user of this interface is responsible for deleting the migrated copy of the data set and updating the necessary control files.

DFSMSdss performs normal non-VSAM data set cataloging if the UIM exit does not turn on the tolerate-migrated-volser indicator.

- If the Extent Reduction bit (EI22EXTR) is set on by the UIM for a given data set and DFSMSdss is running in an SMS-managed environment, DFSMSdss tries to allocate the original volume where the data set was dumped.
- If the Restore-To-Like-Device bit (EI22LIKE) is set on by the UIM for a given data set and DFSMSdss is running in an SMS-managed environment, DFSMSdss tries to allocate the target data set on a device whose unit is the same as that of the source data set. If DFSMSdss can not allocate the target data set on a device whose unit type matches that of the source, the data set is not processed.
- Serialization by enqueueing on the major name of SYSDSN for a data set is bypassed if the UIM exit turns on the bypass SYSDSN indicator, EI22NSYS; other enqueues for a data set (for example, SYSVSAM) are not bypassed. The SYSDSN-level of enqueue serialization is bypassed for

the source data set for DUMP and the target data set for RESTORE. DFSMSdss performs normal serialization if both EI22BSER and EI22NSYS are turned off.

DFSMSdss does not delete or uncatalog data sets if EI22NSYS is set, even if the DELETE or UNCATALOG keywords are specified. If EI22NSYS is on and the preallocated data set is too small, the data set is not scratched and reallocated, and the RESTORE fails.

If the UIM has set the “set reconnect” flag on, DFSMSdss then sets the reconnect flag in the catalog on. DFSMSdss ignores the EI22SFSM flag for VSAM data sets.

If the UIM exit sets the return code to EIRC16 but fails to turn on any of the bypass verification indicators, DFSMSdss ignores all bypass options and performs normally. If the UIM exit sets the return code to EIRC16 for a VSAM data set and turns on the tolerate-migrated-volser indicator, DFSMSdss does not treat it as an error condition and ignores the tolerate-migrated-volser indicator.

20 Disconnect exit

32 End function

36 End data set

Refer to EIRC36 in “Data Set Verification (Eioption 21)” on page 323 for details on ending spheres.

Note: See the “Note for Eioptions 21, 22, and 23” at the end of the description for eioption 23.

Data Set Processed Notification Exit (Eioption 23)

This exit indicates to the UIM whether or not the logical copy, dump, or restore processing for individual data sets was successful. This exit is given control through the UIM at the conclusion of logical copy, dump, and restore processing for each data set. DFSMSdss provides the UIM with information through the EIREC23 structure within the exit identification block, ADREID0. EIREC23 is shown in Table 18 on page 331.

DFSMSdss provides the UIM with the following information at the conclusion of processing for logical dump of each data set:

- The data set name that was dumped
- The RLS time stamps associated with the dump
- A flag indicating whether or not the data set is marked “recovery required”
- A return code for the data set that indicates whether processing was successful:

0 Data set completely successful (informational)

4 Data set partially successful (warning)

8 Data set unsuccessful (error)

12 Ending error

16 Ending error

Programming Interface information

- The source SMS flag (DS1SMSFG) from the Format 1 DSCB

- The source data set organization (DS1DSORG) from the Format 1 DSCB for a non-VSAM data set

End of Programming Interface information

- Flags to indicate data set type for a VSAM data set

DFSMSdss provides the UIM with the following information at the conclusion of processing for a logical copy and a logical restore of each data set:

- The original data set name that was copied or restored. This is the data set name prior to rename processing, if any.
- The new data set name being copied or restored if rename processing was performed.
- A return code for the data set that indicates whether processing succeeded:
 - 0** Data set completely successful (informational)
 - 4** Data set partially successful (warning)
 - 8** Data set unsuccessful (error)
 - 12** Ending error
 - 16** Ending error

Programming Interface information

- The source and target SMS flags (DS1SMSFG) from the Format 1 DSCB.
- The source and target data set organization (DS1DSORG) from the Format 1 DSCB for a non-VSAM data set.

End of Programming Interface information

- Flags to indicate data set type for a VSAM data set.
- A count of the volumes that the data set was copied or restored to.
- A list of the volume serial numbers (VOLSERs) that the data set was copied or restored to.

The UIM returns to DFSMSdss with a return code in the exit identification block. The valid return codes are:

- 0** Continue normal processing
- 20** Disconnect exit
- 32** End function
- 36** End data set

It does not undo any successful processing, but deletes the data set from the successfully-processed message list (if applicable) and includes it in the unsuccessfully-processed message list. It is the responsibility of the user of this interface to delete the copied, dumped, or restored data set from the target volumes. DFSMSdss continues processing with the next data set, if any. Refer to EIRC36 in “Data Set Verification (Eoption 21)” on page 323 for details on ending spheres.

Note for Eioptions 21, 22, and 23

If Eioptions 21, 22, and 23 end the function (UIM passes back a return code 32 to DFSMSdss) during a data set copy operation, DFSMSdss can be processing more than one data set at the time. This can happen if utilities are needed to process some of the data sets. DFSMSdss does the following before ending:

- No unprocessed data sets are scheduled for processing.
- All data sets in utility processing are allowed to complete normally.
- No new calls are made to exits 21, 22, or 23. This includes data sets in utility processing that are allowed to complete.
- Spheres that are being processed because of the SPHERE keyword and that are not complete at the end of the function have the target parts deleted to preserve sphere integrity. Preallocated spheres are left partially copied.
- Spheres being processed without the SPHERE keyword or individual sphere components being copied are left partially completed.

Concurrent Copy Initialization Complete (Eioption 24)

Programming Interface information

DFSMSdss calls the UIM with option code 24 to inform it that the initialization of the concurrent copy session for a given data set or volume has completed. For full-volume or tracks operation, there is only one call (because there is only one input volume). For a physical data set operation, there is one call for each input volume. For a logical data set operation, there is one call for every data set. DFSMSdss does not call the UIM with this option code if the CONCURRENT keyword is not specified. DFSMSdss provides the UIM with information through the EIREC24 structure within the Exit Identification Block, ADREIB (in the ADREID0 macro).

DFSMSdss provides the UIM with the following information:

- A return code indicating whether or not the concurrent copy session initialization succeeded:
 - 0** Concurrent copy session initialization was successful and any serialization on the data has been released.
 - 4** Concurrent copy session initialization failed. If this is a logical data set operation, this data set is not processed using concurrent copy, but other data sets may be. If this is a full-volume, tracks, or physical data set operation, concurrent copy is not used. In either case, serialization is obtained and released as if CONCURRENT were not specified. When this exit is called, DFSMSdss may not be holding any serialization.
- A reason code providing further details about the status of the concurrent copy session initialization (always valid, even for a zero return code):
 - 0** The concurrent copy operation is logically complete at this time (data movement has not been done yet).
 - 4** All parts of the data being dumped or copied were not on hardware supporting concurrent copy.
 - 8** Hardware limits exceeded.

- 12** System data mover failed.
- 16** Host limits exceeded.
- 20** System data mover not available.
- 24** Other host error.
- 28** Data set type not supported for concurrent copy.
- 32** The concurrent copy operation is logically and physically complete at this time (data movement has been done).
- 36** The data being processed requires the use of SnapShot, but the SnapShot software support is not available on the system.
- The volume serial of the volume on which the concurrent copy initialization was attempted (valid only if the reason code is not 16).
- The name of the data set on which the concurrent copy initialization was attempted (valid only if a logical data set operation is being performed).
- A flag indicating whether or not DFSMSdss has reset the data-set-changed flag in the VTOC for the data set (valid only if a logical data set dump operation is being performed).

The UIM returns to DFSMSdss with a return code in the Exit Identification Block. DFSMSdss continues to process based on that return code, as follows:

- 00** Continue normal processing
- 20** Disconnect this exit
- 32** The DFSMSdss function is ended (not valid for a physical data set dump or logical data set copy)

Error handling is the same as described under “User Return Code” on page 312.

————— End of Programming Interface information —————

Backspace Physical Tape Record (Eioption 25)

This exit point is called when DFSMSdss requires the input to be repositioned to the previous tape block (tape or DASD). The EIRECPTR is zero. The valid return codes are:

- 0** Continue normal processing
- 20** Disconnect exit
- 32** End function

Dump Volume Output Notification (Eioption 26)

Exit 26 of the DFSMSdss UIM may be used to determine the following:

- When a new dump output volume has been added for each DDNAME being processed. (DFSMSdss sets EI26VOL with EI26DDN and EI26VSER.)
- When a dump output volume associated with a specific DDNAME receives a terminating error condition. (DFSMSdss sets EI26TERM with EI26DDN and EI26VSER. EI26VTRC contains the failing function return code.)
- When an R0 count mismatch occurs during dump processing for BWO(TYPEIMS) KSDS data sets when EI22IMS is on. (DFSMSdss sets EI26ROCE. The data set name is located by EI26DSN.)

Application Interface

- When a dump output volume associated with a specific DDNAME is closed. (DFSMSdss sets EI26VCLO with EI26DDN and EI26VSR. EI26VTRC contains the return code from CLOSE.)
Exit 26 is invoked with EI26VCLO set only during logical dump operations and only for output data sets residing on DASD devices.
If Exit 26 is not driven for a given DDNAME or is driven with EI26VTRC greater than zero, then the output data set associated with the DDNAME was not closed successfully and should not be depended upon for subsequent restore operations.

The UIM presents a return code to DFSMSdss in the Exit Identification Block. DFSMSdss continues to process based on that return code, as follows:

- 00 DFSMSdss continues normal processing
- 20 Disconnect this exit
- 32 DFSMSdss ends the function (valid only for EI26TERM)

Avoiding Lockout

Refer to “Avoiding Lockout” on page 288 for a description of the ENQ scheme to prevent lockout.

Application Interface Summary

For **Record Processing**, if the UIM makes any changes to the presented record (assuming it can validly do so), return code 16 must be returned or the changes are ignored. If the record is to be replaced in total, return code 4 must be returned or the original record is used. If a record is to be inserted before the current record, return code 8 must be returned or the record is ignored. If the current record is to be deleted, return code 12 must be returned or the current record is processed. If the exit is no longer interested in processing any records appearing at the current DFSMSdss exit point, return code 20 must be returned or the exit is called at the next visitation of DFSMSdss to the exit point. The current record is still processed in either event. If you want to receive only user statistical records at any specific exit, return code 24 must be returned. If you want to process a WTOR within the UIM, the response must be returned to DFSMSdss with a return code 28.

If a record is being returned that is longer than the original record, the exit must either replace the record in the area pointed to by the original record pointer (as long as it does not exceed the length in EIRECALN) or supply another area and store the address to the record in the original record pointer field. In either case, the length must be stored in the original record length field and a return code of 4 must be used.

If a record is being returned that is shorter than the original record, the exit can supply an area and store the address to the record in the original record pointer field or, if allowed, can replace the original record with the shorter one. In either case, the length of the new record must be placed in the original record length field. If the new area option was used, return code 4 must be used. If the new record overlaid the original, return code 16 must be used. Table 18 on page 331 describes the data area corresponding to the DFSMSdss exit identification block.

For **Data Set Processing**, Eioptions 21, 22, and 23 are three exits that give you added control over data set processing during a logical data set copy, dump, and

restore operations. These exits are called immediately before and immediately after processing each data set, on a data set by data set basis, and allow you to make a number of processing changes.

Eioptions 21 and 22 are called, consecutively, at the start of processing of each data set. DFSMSdss passes the name of the data set being processed to each exit. Based on that name, each exit permits the following:

- End processing of that one data set (return code 36).
- End processing of the entire functional task (return code 32).
- Disconnect the exit so that it will not be called before subsequent data sets task processing (return code 20).
- Perform nothing (return code 0).
- Eioption 22 only: Modify how the data set will be processed (return code 16). See “Bypass Verification Exit (Eioption 22)” on page 323 for details.

Eioption 23 is called immediately after a data set is processed. Eioption 23 lets you end processing of both the data set and the task, disconnect the exit, or do nothing.

ADREID0 Data Area

Table 18 describes the data area corresponding to the DFSMSdss exit identification block.

Table 18. ADREID0 Mapping Macro

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
0	(0)	STRUCTURE	42	ADREIB	
0	(0)	SIGNED	2	EIDLLEN	LENGTH OF ADREIB - 2
2	(2)	CHARACTER	4	EIID	BLOCK IDENTIFIER EBCDIC "EIDB"
6	(6)	SIGNED	4	EITSKID	TASK ID NUMBER
10	(A)	BITSTRING	4	EIXALLOW	USER EXIT ALLOWANCE OPTIONS
10	(A)	BITSTRING	1	EIXALLOW0	ALLOWANCE OPTION BYTE 1
				EIXREP	ALLOW REPLACE OF RECORD
				EIXINS	ALLOW INSERTION OF RECORD
				EIXDEL	ALLOW DELETION OF RECORD
				EIXMOD	ALLOW MODIFICATION OF RECORD
				EIXDIS	ALLOW DISCONNECT OF EXIT
				EIXWTOR	ALLOW WTOR RESPONSE
				EIXSTAT	ALLOW SELECTION OF USER STATS
				EIXTERM	ALLOW FUNCTION TERMINATION
11	(B)	BITSTRING	1	EIXALLOW1	ALLOWANCE OPTION BYTE 2
				EEXTDSET	ALLOW DATA SET TERMINATION
12	(C)	BITSTRING	1	EIXALLOW2	UNUSED ALLOWANCE OPTION BYTE 3

Application Interface

Table 18. ADREID0 Mapping Macro (continued)

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
12	(C)	BITSTRING	1	*	RESERVED
13	(D)	BITSTRING	1	EIXALOW3	ALLOWANCE OPTION BYTE 4
		1...		EIXERR	DISALLOWED OPTION ATTEMPTED
		.111 1111		*	RESERVED
14	(E)	SIGNED	2	EIOPTION	PROCESSING OPTION
16	(10)	SIGNED	2	EIRETCOD	EXIT RETURN CODE
18	(12)	SIGNED	4	EIRECALN	RECORD AREA LENGTH
22	(16)	SIGNED	4	EIRECLEN	ORIGINAL RECORD LENGTH
26	(1A)	ADDRESS	4	EIRECPTR	ORIGINAL RECORD ADDRESS
30	(1E)	ADDRESS	4	EIUSEPTR	USER DATA AREA ADDRESS
34	(22)	ADDRESS	4	EIDDID	EIOP06 DDNAME/VOLID PTR
38	(26)	BITSTRING	4	EIXFLAGS	OTHER FLAGS
38	(26)	BITSTRING	1	EIXFLAG0	FLAG BYTE 0
		1...		EIXABEND	FOR EIOP02 ONLY, 1=MESSAGE IS ADR013 INDICATING AN ABEND CONDITION
		.1..		EIXNTERR	FOR EIOP02 ONLY, 1=MESSAGE IS TYPE 'E' AND IS NOT 324 OR 347
		..1.		EIXWNGOK	FOR EIOP14 ONLY, 1=WARNING MSGS WERE ISSUED AND NONE ARE FATAL
		...1		EIXTRKER	TRACK IS IN ERROR
	 1111		*	RESERVED
39	(27)	BITSTRING	1	EIXFLAG1	FLAG BYTE 1
		1...		*	RESERVED
		.1..		*	RESERVED
		..1.		EIXTDUMS	DUMMY SOURCE TAPE
		...1		EIXTDUMT	DUMMY TARGET TAPE
	 1...		EIXTISXM	APPL USING XMAPI
	1..		EIXDASYS	XMAPI - DYNALOC SYS
	11		*	RESERVED
40	(28)	BITSTRING	1	EIXFLAG2	FLAG BYTE 2
40	(28)	BITSTRING	1	*	RESERVED
41	(29)	BITSTRING	1	EIXFLAG3	FLAG BYTE 3
41	(29)	BITSTRING	1	*	RESERVED
42	(2A)	CHARACTER		*	FORCE SIZE OF CONTROL BLOCK TO WORD BOUNDARY

Note: THE FOLLOWING MAPS THE RECORD PRESENTED BY EXIT 6 AND POINTED TO BY EIDDID.

0	(0)	STRUCTURE	16	EIDDINFO	UIM-06 INFO
0	(0)	CHARACTER	8	EIDDNAME	O/P DEVICE DDNAME
8	(8)	CHARACTER	6	EIVOLID	O/P VOLSER
14	(E)	UNSIGNED	1	EIRETC	RESERVED FOR RETCODE
15	(F)	UNSIGNED	1	*	RESERVED FOR FLAGS

Note: THE FOLLOWING MAPS THE RECORD PRESENTED BY EXIT 00 (FUNCTION STARTUP).

0	(0)	STRUCTURE	16	EIREC00	
---	-----	-----------	----	---------	--

Table 18. ADREIDO Mapping Macro (continued)

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
0	(0)	UNSIGNED	1	EI00SBPL	SHARED SUBPOOL NUMBER
1	(1)	BITSTRING 1...1...1.1.... 1...	1	EI00FLGS EI00SENQ EI00NONF EI00NOLK EI00NENQ EI00BSEC	STARTUP FLAGS SHORT VTOC ENQUEUE DO NOT ALLOW IGWNOTIF CALL DO NOT PERFORM ADRLOCK ENQUEUE OR DEQUEUE NO INPUT VOLUME SERIALIZATION. VALID DURING PHYSICAL COPY AND DUMP FUNCTION STARTUPS BYPASS SECURITY VERIFICATION FOR DATA SETS ON THE VOLUME DURING COPY FULL/TRACKS, DUMP FULL/TRACKS, AND RESTORE FULL/TRACKS FUNCTION STARTUPS
2	(2)111 CHARACTER	6	EI00SVOL	FULL/TRACKS, DUMP FULL/TRACKS, AND RESTORE FULL/TRACKS FUNCTION STARTUPS. FOR RESTORE, THIS FIELD CONTAINS THE VOLSER OF THE INPUT VOLUME WHERE THE DUMP DATA SET RESIDES
8	(8)	CHARACTER	8	*	RESERVED
Note: THE FOLLOWING MAPS THE RECORD PRESENTED BY EXIT 14: DSS MESSAGE NUMBER ASSOCIATED WITH NON-ZERO RETCODE.					
0	(0)	STRUCTURE	16	EIREC14	
0	(0)	CHARACTER	4	EI14RC	DSS RETURN CODE
4	(4)	CHARACTER	4	EI14MESS	DSS MESSAGE
4	(4)	CHARACTER	3	EI14MNUM	MESSAGE NUMBER
7	(7)	CHARACTER	1	EI14MTYP	MESSAGE TYPE
8	(8)	CHARACTER	8	EI14CPU	DSS CPU TIME
Note: THE FOLLOWING MAPS THE RECORD PRESENTED BY EXIT 20: VOLUME NOTIFICATION EXIT.					
0	(0)	STRUCTURE	56	EIREC20	
0	(0)	CHARACTER	44	EI20DSN	DATA SET NAME/CLUSTER NAME
44	(2C)	BITSTRING 1...1...11 1111	1	EI20FLGS EI20VSAM EI20RACF	SOME FLAGS: 1=DATA SET IS VSAM 0=DATA SET IS NONVSAM 1=DATA SET IS PROTECTED BY A DISCRETE RACF PROFILE
45	(2D)	CHARACTER	1	*	RESERVED
				*	RESERVED

Application Interface

Table 18. ADREID0 Mapping Macro (continued)

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
46	(2E)	UNSIGNED	1	EI20DA#	NUMBER OF DATA COMPONENTS
47	(2F)	UNSIGNED	1	EI20IX#	NUMBER OF INDEX COMPONENTS (0 IF NONVSAM)
48	(30)	ADDRESS	4	EI20DA@	POINTER TO DATA COMPONENT INFO FOR VSAM/DATA SET INFO FOR NONVSAM
52	(34)	ADDRESS	4	EI20IX@	POINTER TO INDEX COMPONENT

Note: INFO FOR VSAM/0 FOR NONVSAM FOR A NON-VSAM DATA SET, EI20DA@ POINTS TO A SINGLE EI20DSI STRUCTURE. FOR A VSAM CLUSTER, EI20DA@ POINTS TO AN ARRAY OF STRUCTURES (ONE FOR EACH DATA COMPONENT), AND EI20IX@ POINTS TO A SIMILAR ARRAY FOR THE INDEX COMPONENTS.

0	(0)	STRUCTURE	60	EI20DSI	DATA SET INFO
0	(0)	CHARACTER	44	EI20CON	COMPONENT NAME (BLANKS FOR NONVSAM)
44	(2C)	UNSIGNED	2	EI20NVOL	# OF VOLUMES IN DS
46	(2E)	CHARACTER	2	*	RESERVED
48	(30)	CHARACTER	12	EI20VLI	VOLUME INFORMATION
48	(30)	CHARACTER	6	EI20VOL	VOLSER
54	(36)	CHARACTER	2	*	RESERVED
56	(38)	UNSIGNED	4	EI20RBA	RBA TOKEN

Note: THE FOLLOWING MAPS THE RECORD PRESENTED BY EXIT 21: DATA SET VERIFICATION EXIT.

0	(0)	STRUCTURE	44	EIREC21	
0	(0)	CHARACTER	44	EI21DSN	DATA SET/CLUSTER NAME

Note: THE FOLLOWING MAPS THE RECORD PRESENTED BY EXIT 22: BYPASS VERIFICATION EXIT.

0	(0)	STRUCTURE	102	EIREC22	
0	(0)	CHARACTER	44	EI22DSN	DATA SET/CLUSTER NAME
44	(2C)	BITSTRING	1	EI22FLGS	EXIT 22 FLAGS:
		1...		EI22BSER	1=BYPASS SERIALIZATION
		.1..		EI22BSEC	1=BYPASS DATASET LEVEL, STORCLAS & MGMTCLAS SECURITY CHECKS. ALSO BYPASS JES3 INTEGRITY CHECKS AND DO NOT CREATE DISCRETE DATASET PROFILES.
		..1.		EI22BMIG	1=TOLERATE MIGRATED VOLSER FOR NONVSAM DSETS (RESTORE ONLY) @DS007
		...1		EI22NSYS	BYPASS SYSDSN ENQ SWAPPED W/ EI22SIRS
	 1...		EI22LIKE	RESTORE/COPY DATA SET TO A LIKE DEVICE TYPE
	1..		EI22EXTR	RESTORE DATA SET TO EXTENT REDUCTION VOLUME
	1.		EI22SIRS	SET INCMPLT RECALL SWAPPED W/ EI22NSYS
44	1		EI22DB2	BYPASS SOURCE FOR DB2 RENAME

Table 18. ADREIDO Mapping Macro (continued)

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
45	(2D)	CHARACTER	5	EI22SFLG	SOURCE DATA SET FLGS
45	(2D)	CHARACTER	1	E22SSMSF	SOURCE SMS FLAGS
46	(2E)	CHARACTER	2	E22SDSRG	SOURCE DATA SET ORG
48	(30)	BITSTRING	2	E22SVFLG	SRCE VSAM DSET FLAGS
		1...		E22SESDS	1=ESDS DATA SET
		.1...		E22SKSDS	1=KSDS DATA SET
		..1...		E22SKRDS	1=KRDS DATA SET
		...1....		E22SLDS	1=LINEAR DATA SET
	1...		E22SRRDS	1=RRDS DATA SET
	1..		E22SPSSI	1=PAGE/SWAP/STGINDEX, ETC. UNSUPPRTED DATA SETS
	1.		E22SVVDS	1=VVDS DATA SET
	1		E22SBCS	1=BCS DATA SET
49	(31)	1...		E22SAIX	1=AIX DATA SET
		.1...		E22SVRRD	VRRDS DATASET
		..11 1111		*	RESERVED
50	(32)	BITSTRING	1	EI22FLG2	MORE EXIT 22 FLAGS
		1...		EI22RSET	RESET DS CHANGED FLAG
		.1...		EI22SSYS	GET SHARED SYSSN ENQ
		..1...		EI22RRB	MARK TGT RECOVRY REQ'D
		...1....		EI22LINF	LOG INFO PASSED, SEE EI22LPRM AND EI22LSID
	1...		EI22BWOP	BWO_ALLOWED PASSED, SEE EI22BWOA
	1..		EI22IMS	CALLER IS IMS
	1.		EI22BWOE	ENQ ON BWODSN ONLY, NO ADR OR SYS ENQ'S
	1		*	UNUSED
51	(33)	CHARACTER	1	EI22LPRM	LOG PARAMETER FOR TGT
52	(34)	CHARACTER	26	EI22LSID	LOG STREAM ID FOR TGT
78	(4E)	BITSTRING	1	EI22FLG3	MORE EXIT 22 FLAGS
		1111		EI22BWOA	BWO_ALLOWED
	 1...		EI22BRLS	BYPASS RLS QUIESCE
	1..		EI22SFSM	SET RECONNECTABLE FLAG
	11		*	UNUSED
79	(4F)	CHARACTER	23	*	AVAILABLE

Note: THE FOLLOWING MAPS THE RECORD PRESENTED BY EXIT 23: DATA SET PROCESSED NOTIFICATION EXIT.

0	(0)	STRUCTURE	*	EIREC23	
0	(0)	CHARACTER	141	EI23CNST	CONSTANT LENGTH PORTION OF CONTROL BLK
0	(0)	CHARACTER	44	EI23DSN	DATA SET/CLUSTER NAME
44	(2C)	CHARACTER	44	EI23NEWN	NEW DSET/CLUSTER NAME
88	(58)	SIGNED	2	EI23DSRC	RETURN CODE FOR DATA SET PROCESSING
90	(5A)	CHARACTER	5	EI23SFLG	SOURCE DATA SET FLAGS
90	(5A)	CHARACTER	1	E23SSMSF	SOURCE SMS FLAGS
91	(5B)	CHARACTER	2	E23SDSRG	SOURCE DATA SET ORG
93	(5D)	BITSTRING	2	E23SVFLG	SOURCE VSAM DSET FLAGS
		1...		E23SESDS	1=ESDS DATA SET

Application Interface

Table 18. ADREID0 Mapping Macro (continued)

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
94	(5E)	CHARACTER	5	E23SKSDS	1=KSDS DATA SET
				E23SKRDS	1=KRDS DATA SET
				E23SLDS	1=LINEAR DATA SET
				E23SRRDS	1=RRDS DATA SET
				E23SPSSI	1=PAGE/SWAP/STGINDEX, ETC. UNSUPPORTED DATA SETS
				E23SVVDS	1=VVDS DATA SET
				E23SBCS	1=BCS DATA SET
				E23SAIX	1=AIX DATA SET
				E23BCSEL	1=BASE CLUSTER SELECTED
				E23SVRRD	VRRDS DATASET
95	(5F)	CHARACTER	1	EI23RRB	RECOVERY REQUIRED
				*	UNUSED
				EI23TFLG	TARGET DATA SET FLAGS
				E23TSMSF	TARGET SMS FLAGS
				E23TDSRG	TARGET DATA SET ORG
				E23TVFLG	TARGET VSAM DSET FLAGS
				E23TESDS	1=ESDS DATA SET
				E23TKSDS	1=KSDS DATA SET
				E23TKRDS	1=KRDS DATA SET
				E23TLDS	1=LINEAR DATA SET
99	(63)	BITSTRING	2	E23TRRDS	1=RRDS DATA SET
				E23TPSSI	1=PAGE/SWAP/STGINDEX, ETC. UNSUPPORTED DATA SETS
				E23TVVDS	1=VVDS DATA SET
				E23TBCS	1=BCS DATA SET
				E23TAIX	1=AIX DATA SET
				E23TVRRD	VRRDS DATASET
				*	RESERVED
				.11 1111	
				EI23RLST	RLS TIME STAMPS
				EI23GMT	RLS GMT TIME STAMP
100	(64)	CHARACTER	16	EI23LOC	RLS LOCAL TIME STAMP
100	(64)	CHARACTER	8	E23BYTES	DS BYTE COUNT
108	(6C)	CHARACTER	8	E23FLGS	MISC FLAGS
116	(74)	CHARACTER	8	E23BSET	1=E23BYTES SET
124	(7C)	CHARACTER	1	E23BPDS	1=BROKEN PDS
				E23BPSE	E23BYTES IS FOR PSE
				*	RESERVED
125	(7D)	CHARACTER	6	EI23DNAM	NAME OF DEVICE NEEDED TO DO NOPACKING FOR THE BROKEN PDS (I.E. A LIKE DEVICE)
131	(83)	CHARACTER	9	*	RESERVED
140	(8C)	UNSIGNED	1	EI23VOL#	NUMBER OF VOLUMES
141	(8D)	CHARACTER	6	EI23VSER (*)	VOLSER ARRAY
Note: THE FOLLOWING MAPS THE RECORD PRESENTED BY EXIT 24: CONCURRENT COPY INITIALIZATION COMPLETE.					
0	(0)	STRUCTURE	83	EIREC24	
0	(0)	UNSIGNED	2	EI24RTCD	RETURN CODE
2	(2)	UNSIGNED	2	EI24RSCD	REASON CODE
4	(4)	UNSIGNED	4	*	RESERVED
8	(8)	CHARACTER	6	EI24VOL	VOLUME SERIAL

Table 18. ADREIDO Mapping Macro (continued)

Offsets		Type	Len	Name (Dim)	Description
Dec	Hex				
14	(E)	CHARACTER BITSTRING 1...111 1111	44 1 * 24	EI24DSN	DATA SET NAME
58	(3A)			EI24FLGS	FLAGS
				EI24RSET	DS CHANGE FLAG RESET
59	(3B)			*	UNUSED
Note: THE FOLLOWING MAPS THE RECORD PRESENTED BY EXIT 26: DUMP OUTPUT VOLUME MOUNT NOTIFICATION EXIT.					
0	(0)	STRUCTURE BITSTRING 1...1...1.1 1111	128 4 * * * * * * * *	EIREC26	EXIT TYPE
0	(0)			EI26TYPE	OUTPUT VOLUME
				EI26VOL	NOTIFICATION
				EI26TERM	OUTPUT VOLUME
					TERMINATED
				EI26R0CE	BWO R0 COUNT ERROR
				EI26VCLO	OUTPUT VOLUME CLOSE - ONLY FOR DASD OUTPUTS DURING LOGICAL DUMP OPERATIONS
					UNUSED
0	(0)			*	RESERVED FOR EXPANSION
4	(4)			EI26DSN	DSNAME IF EI26ROCE = '1'B
68	(44)	UNSIGNED	64	EI26DSNL	LENGTH OF DSNAME
69	(45)	UNSIGNED	1		RESERVED FOR ALIGNMENT
72	(48)	CHARACTER	3		RESERVED FOR ALIGNMENT
80	(50)	CHARACTER	64	EI26DDN	OUTPUT DDNAME IF EI26VOL, EI26TERM, OR EI26VCLO SET
86	(56)	CHARACTER	6	EI26VSER	VOLSER - PRESENT IF EI26VOL, EI26TERM, EI26VCLO SET
88	(58)	CHARACTER	2	EI26VTRC	RESERVED FOR ALIGNMENT
92	(5C)	ADDRESS	4		RETURN FOR VOLUME TERM AND VOLUME CLOSE
			36	*	RESERVED FOR EXPANSION

Constants

Len	Type	Value	Name	Description
4	CHARACTER	EIDB	ADREIBID	BLOCK IDENTIFIER
2	DECIMAL	0	EIRC00	CONTINUE NORMAL PROCESS
2	DECIMAL	4	EIRC04	RECORD REPLACED
2	DECIMAL	8	EIRC08	INSERT RECORD
2	DECIMAL	12	EIRC12	DELETE RECORD
2	DECIMAL	16	EIRC16	RECORD MODIFIED
2	DECIMAL	20	EIRC20	DISCONNECT EXIT
2	DECIMAL	24	EIRC24	SELECT USER STATS RECS
2	DECIMAL	28	EIRC28	WTOR RESPONSE
2	DECIMAL	32	EIRC32	TERMINATE FUNCTION
2	DECIMAL	36	EIRC36	TERMINATE DSET ONLY
2	DECIMAL	0	EIOP00	FUNCTION STARTUP ENTRY
2	DECIMAL	1	EIOP01	READING SYSIN RECORD
2	DECIMAL	2	EIOP02	PRINTING SYSPRINT RECORD
2	DECIMAL	3	EIOP03	READING PHYSICAL TAPE
2	DECIMAL	4	EIOP04	READING LOGICAL TAPE
2	DECIMAL	5	EIOP05	WRITING LOGICAL TAPE
2	DECIMAL	6	EIOP06	WRITING PHYSICAL TAPE
2	DECIMAL	7	EIOP07	READING DISK TRACK
2	DECIMAL	8	EIOP08	WRITING DISK TRACK
2	DECIMAL	9	EIOP09	READING UTILITY SYSPRINT
2	DECIMAL	10	EIOP10	WRITING UTILITY SYSPRINT
2	DECIMAL	11	EIOP11	WRITING WTO MESSAGE
2	DECIMAL	12	EIOP12	WRITING WTOR MESSAGE
2	DECIMAL	13	EIOP13	PRESENTING ADRUFO REC
2	DECIMAL	14	EIOP14	FUNCTION TERMINATION
2	DECIMAL	15	EIOP15	PRESENTING WTOR RESPONSE
2	DECIMAL	16	EIOP16	TAPE VOL SECURITY
2	DECIMAL	17	EIOP17	TAPE MOUNT(NON-SPEC)
2	DECIMAL	18	EIOP18	INSERT LOGICAL REC
2	DECIMAL	19	EIOP19	TAPE OUTPUT ERROR
2	DECIMAL	20	EIOP20	VOLUME NOTIFICATION
2	DECIMAL	21	EIOP21	DSET VERIFICATION
2	DECIMAL	22	EIOP22	BYPASS VERIFICATION
2	DECIMAL	23	EIOP23	DS PROCEDURE NOTIFICATION
2	DECIMAL	708	VOSSLST —	MAX SIZE OF VOLSER LIST FOR EIREC23 (118 6)
2	DECIMAL	24	EIOP24	CC INIT DONE
2	DECIMAL	25	EIOP25	BACKSPACE TAPEIN
2	DECIMAL	26	EIOP26	VOLUME OPEN FOR O/P
2	DECIMAL	48	EXITRECL	EXIT RECORD LEN
4	DECIMAL	42	EIDBLEN	LENGTH OF BLOCK

Cross Reference

Name	Hex Offset	Hex Value	Level
ADREIB	0		1
EIDDID	22		2
EIDDINFO	0		1
EIDDNAME	0		2
EIDLLEN	0		2
EIID	2		2
EIOPTION	E		2
EIRECALN	12		2
EIRECLEN	16		2
EIRECPTR	1A		2
EIREC00	0		1
EIREC14	0		1
EIREC20	0		1
EIREC21	0		1
EIREC22	0		1
EIREC23	0		1
EIREC24	0		1
EIREC26	0		1
EIRETC	E		2
EIRETCOD	10		2
EITSKID	6		2
EIUSEPTR	1E		2
EIVOLID	8		2
EIXABEND	26	80	4
EIXALLOW	A		2
EIXALOW0	A		3
EIXALOW1	B		3
EIXALOW2	C		3
EIXALOW3	D		3
EIXDASYS	27	04	4
EIXDEL	A	20	4
EIXDIS	A	08	4
EIXERR	D	80	4
EIXFLAGS	26		2
EIXFLAG0	26		3
EIXFLAG1	27		3
EIXFLAG2	28		3
EIXFLAG3	29		3
EIXINS	A	40	4
EIXMOD	A	10	4
EIXNTERR	26	40	4
EIXREP	A	80	4
EIXSTAT	A	02	4
EIXTDSET	B	80	4
EIXTDUMS	27	20	4
EIXTDUMT	27	10	4
EIXTERM	A	01	4
EIXTISXM	27	08	4
EIXTRKER	26	10	4
EIXWNGOK	26	20	4
EIXWTOR		04	4
EI00SBPL	0		2
EI00FLGS	1		2
EI00SENQ	1	80	3
EI00NONF	1	40	3
EI00NOLK	1	20	3

Application Interface

Name	Hex Offset	Hex Value	Level
EI00NENQ	1	10	3
EI00BSEC	1	08	3
EI00SVOL	2		2
EI14CPUT			2
EI14MESS	4		2
EI14MNUM	4		3
EI14MTYP	7		3
EI14RC	0		2
EI20CON	0		2
EI20DA#	2E		2
EI20DA@	30		2
EI20DSI	0		1
EI20DSN	0		2
EI20FLGS	2C		2
EI20IX#	2F		2
EI20IX@	34		2
EI20NVOL	2C		2
EI20RACF	2C	40	3
EI20RBA	38		3
EI20VLI	30		2
EI20VOL	30		3
EI20VSAM	2C	80	3
EI21DSN	0		2
EI22BMIG	2C	20	3
EI22BRLS	4E	08	2
EI22BSEC	2C	40	3
EI22BSER	2C	80	3
EI22BWOA	4E	F0	3
EI22BWOE	32	02	3
EI22BWOP	32	08	3
EI22DB2	2C		3
EI22DSN	0		2
EI22EXTR	2C	04	3
EI22FLGS	2C		2
EI22FLG2	32		2
EI22FLG3	4E		2
EI22IMS	32	04	3
EI22LIKE	2C	08	3
EI22LINF	32	10	3
EI22LPRM	33		2
EI22LSID	34		2
EI22NSYS	2C	10	3
EI22RRB	32	20	3
EI22RSET	32	80	3
EI22SFLG	2D		2
EI22SFSM	4E	04	2
EI22SIRS	2C	02	3
EI22SSYS	32	40	3
EI23CNST	0		2
EI23DNAM	7D		3
EI23DSN	0		3
EI23DSRC	58		3
EI23GMT	64		4
EI23LOC	6C		4
EI23NEWN	2C		3
EI23RLST	64		3
EI23RRB	5E	10	5
EI23SFLG	5A		3

Name	Hex Offset	Hex Value	Level
EI23TFLG	5F	3	
EI23VOL#	8C	3	
EI23VSER	8D	2	
EI24DSN	E	2	
EI24FLGS	3A	2	
EI24RSCD	2	2	
EI24RSET	3A	80	3
EI24RTCD	0		2
EI24VOL	8		2
EI26DDN	C		2
EI26DSNL	8		2
EI26DSNP	4		2
EI26R0CE	0	20	3
EI26TERM	0	40	3
EI26TYPE	0		2
EI26VCLO	0	10	3
EI26VOL	0	80	3
EI26VSER	14		2
EI26VTRC	1C		2
EI26VTRS	24		2
E22SAIX	31	80	4
E22SBCS	30	01	4
E22SDSRG	2E		3
E22SESDS	30	80	4
E22SKRDS	30	20	4
E22SKSDS	30	40	4
E22SLDS	30	10	4
E22SPSSI	30	04	4
E22SRRDS	30	08	4
E22SSMSF	2D		3
E22SVFLG	30		3
E22SVRRD	31	40	4
E22SVVDS	30	02	4
E23BCSEL	5E	40	5
E23BPDS	7C	40	4
E23BPSE	7C	20	4
E23BSET	7C	80	4
E23BYTES	74		3
E23FLGS	7C		3
E23SAIX	5E	80	5
E23SBCS	5D	01	5
E23SDSRG	5B		4
E23SESDS	5D	80	5
E23SKRDS	5D	20	5
E23SKSDS	5D	40	5
E23SLDS	5D	10	5
E23SPSSI	5D	04	5
E23SRRDS	5D	08	5
E23SSMSF	5A		4
E23SVFLG	5D		4
E23SVRRD	5E	20	5
E23SVVDS	5D	02	5
E23TAIX	63	80	5
E23TBCS	62	01	5
E23TDSRG	60		4
E23TESDS	62	80	5
E23TKRDS	62	20	5
E23TKSDS	62	40	5

Application Interface

Name	Hex Offset	Hex Value	Level
E23TLDS	62	10	5
E23TPSSI	62	04	5
E23TRRDS	62	08	5
E23TSMASF	5F		4
E23TVFLG	62		4
E23TVRRD	63	40	5
E23TVVDS	62	02	5

Example: Invoking DFSMSdss by Using an Application Program

The following example shows that a full DASD volume is to be dumped to a tape volume or volumes. DFSMSdss is LINKed somewhere in MYJOB to perform the dump, and, conditionally, the DEFrag functions.

```
//JOB1    JOB  accounting information,REGION=nnnnK
//STEP1   EXEC  PGM=MYJOB
//STEPLIB  DD   DSN=MY.LINKLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=A
//DASD     DD   UNIT=3380,VOL=(PRIVATE,SER=111111),DISP=OLD
//TAPE     DD   UNIT=3480,VOL=SER=(TAPE01,TAPE02),
//          LABEL=(1,NL),DISP=(NEW,KEEP)
//SYSIN    DD   *
      DUMP INDD(DASD) OUTDD(TAPE)
      IF LASTCC = 0 -
          THEN DEFR DDN(DASD)
/*

```

The preceding example does not show how the invocation of DFSMSdss was brought about, but does show that the user program, MYJOB, was run. At some point MYJOB needs to run DFSMSdss to perform the functions specified in the SYSIN data set. The next example shows the code needed at that point to LINK to DFSMSdss. Because no EXEC PARMs were specified and the standard SYSIN and SYSPRINT data set names are to be used, there is no need to pass special parameters.

```
.
.
LINK EP=ADRDSU,PARAM=(OPTPTR),VL=1
.
.
.
CNOP 2,4
OPTPTR DC H(0)
```

Related reading: For additional information about the Application Interface, see Appendix D, “Examples of the Application Program with the User Interaction Module (UIM),” on page 345.

How to Determine DFSMSdss Version, Release, and Modification Level

Subsystems that invoke DFSMSdss dynamically must determine if DFSMSdss is installed on the system, and if it is, its version, release, and modification level, and

features supported. A DFSMSdss-provided macro tries to determine the DFSMSdss version, release, and modification level and features supported and pass the requested information in a register.

ADRMCLVL (in SYS1.MACLIB) is an in-line executable assembler-language macro that can be invoked by a caller. The caller can be in problem program state and can have a user key. The caller must save registers 0, 1, 14, and 15 before invoking the macro. No other registers are disturbed. The caller can determine the installed level and features of DFSMSdss from the information returned in registers 1 and 14.

On return, register 1 contains information as follows:

- If the release level of ADRDSSU cannot be determined, register 1 contains X'04000000'.
- Otherwise, register 1 contains:

Byte 0 Product number (in binary):
 0 = DFDSS
 2 = MVS or OS/390 DFSMSdss
 3 = z/OS DFSMSdss

Byte 1 Version number (in binary):
 1 = Version 1
 2 = Version 2

Byte 2 Release number (in binary):
 1 = Release 1
 2 = Release 2
 3 = Release 3
 4 = Release 4
 5 = Release 5
 A = Release 10

Byte 3 Modification level (in binary)
 0 = Modification level 0
 1 = Modification level 1
 2 = Modification level 2

On return, register 14 contains the following information:

- If the release level of ADRDSSU is less than DFSMSdss Version 1, Release 4, Modification level 0 (see above), then the contents of register 14 are unpredictable.
- Otherwise, register 14 contains the following:

Byte 0 Feature Flags:
 Bit 0, when set to 1, means DFSMSdss cross-memory Application Programming Interface support for concurrent copy is available.
 Bits 1–7 are reserved.

Bytes 1–3 Reserved.

Application Interface

Appendix D. Examples of the Application Program with the User Interaction Module (UIM)

This appendix contains General-use Programming Interface and Associated Guidance information, and Product sensitive Programming Interface information.

Figure 10 shows the process by which DFSMSdss can call user interaction module (UIM) functions.

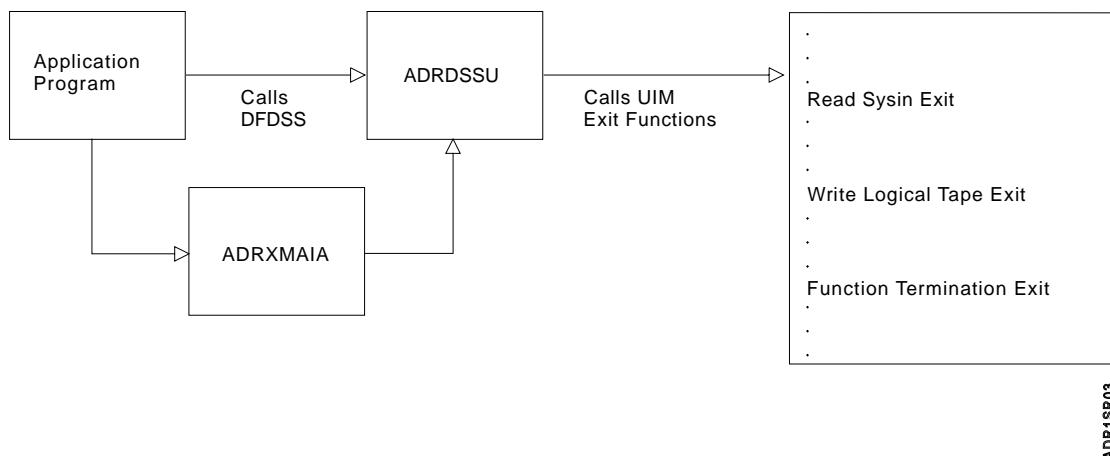


Figure 10. The Application Program Process

This appendix contains the following examples:

- The JCL to invoke an application program that invokes DFSMSdss
 - A complete sample program listing showing how a user can use all of the UIM exit functions to receive control from DFSMSdss (Figure 11 on page 346 and Figure 12 on page 349)
 - An output listing (Figure 13 on page 360) resulting from the sample program in Figure 11 on page 346 and Figure 12 on page 349

Note: The example shown is not written in reentrant code. If you are planning to share a UIM between tasks, you should code the module in reentrant code.

The example has not been submitted to any formal test and is distributed as it is, without any warranty either expressed or implied. The use of this example or the implementation of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. Although each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

The following JCL was used to invoke an application program, which then called DFSMSdss.

```
//STEPS014 EXEC PGM=USRAIPGM
//STEPLIB DD DISP=SHR,DSN=SYS1.LINKLIB
//SYSPRINT DD SYSOUT=*
//DASD DD UNIT=SYSDA,VOL=SER=D9S060,DISP=SHR
//TAPE DD DD DISP=(,CATLG),DSN=PUBSEXMP.DUMP,
//          UNIT=3390,VOL=SER=DAVIS4,
//          SPACE=(CYL,(10,10),RLSE)
//MYDATA DD *
      EOJ
/*
```

```
*****
*                                     *
* Module Name      = USRAIPGM           *
*                                     *
* Descriptive Name = DFSMSdss Application Interface Program Example   *
*                                     *
* Function         = Invoke DFSMSdss supplying alternate UIM: USRUIM   *
*                      or                                         *
*                      = Invoke DFSMSdss Cross Memory Application       *
*                          Interface UIM: USRUIM                   *
*                                     *
* Operation        = Place SYSIN records in the user area. The UIM   *
*                      will pass these records to DFSMSdss.           *
*                                     *
*****
```

USRAIPGM CSECT
 USRAIPGM AMODE 31
 USRAIPGM RMODE ANY

```
*****
* Standard entry linkage           *
*****
```

STM 14,12,12(13)
 BALR 12,0
 USING *,12
 LA 3,SAVEAREA
 ST 3,8(13)
 ST 13,4(3)
 LR 13,3

```
*****
* Load and call DFSMSdss           *
*****
```

LOAD EP=ADRDSU	0YA29363 01-LOAD
+ CNOP 0,4	
+ LA 0,*+8	LOAD PARAMETER INTO REGISTER ZERO 01-LOAD
+ B **+12	BRANCH AROUND CONSTANT(S) 01-LOAD
+ DC CL8'ADRDSU'	ENTRY POINT NAME 01-LOAD
+ SR 1,1	SHOW NO DCB PRESENT 01-LOAD
+ SVC 8	01-LOAD
LR 15,0	
LA 3,USERAREA	
ST 3,UA@	

Figure 11. Application Interface Program Example (Part 1 of 3)

```

CALL  (15),(OPTPTR,DDNPTR,PAGEPTR,UIMPTR,UAPTR),VL
+   DS  0H                                01-CALL
+   CNOP 0,4                               02-IHBOP
+   LA  1,IHB0003                         LIST ADDRESS @L1C 02-IHBOP
+   B   IHB0003A                          BYPASS LIST @ZMC3742 02-IHBOP
+IHB0003 EQU  *                           02-IHBOP
+   DC  A(OPTPTR)            PROB.PROG.PARAMETER 02-IHBOP
+   DC  A(DDNPTR)             PROB.PROG.PARAMETER 02-IHBOP
+   DC  A(PAGEPTR)            PROB.PROG.PARAMETER 02-IHBOP
+   DC  A(UIMPTR)             PROB.PROG.PARAMETER 02-IHBOP
+   DC  A(UAPTR+X'80000000')          @G860P40 02-IHBOP
+IHB0003A EQU  *                           02-IHBOP
+   BALR 14,15                            BRANCH TO ENTRY POINT 01-CALL
*****
* Load and call DFSMSdss Cross Memory Application Interface *
*****
*      LOAD EP=ADRXAIA
*      LR 15,0
*      LA 3,USERAREA
*      ST 3,UA@
*      CALL (15),(OPTPTR,DDNPTR,PAGEPTR,UIMPTR,UAPTR,ASN PTR),VL
*****
* Standard exit linkage
*****
L   13,4(13)
RETURN (14,12),,RC=(15)                   RESTORE REG 14 @L1C 01-RETUR
+   L   14,12(0,13)                      RESTORE THE REGISTERS 01-RETUR
+   LM  0,12,20(13)                      RETURN           01-RETUR
+   BR  14
*****
* Option area (same format as EXEC parameters)
*****
CNOP  2,4
OPTPTR DC  AL2(OPTLEN)
OPTIONS DC  C'SIZE=4096K,TRACE=YES'
OPTLEN EQU  *-OPTIONS
*****
* DDNAME area (SYSIN is replaced by MYDATA)
*****
CNOP  2,4
DDNPTR DC  AL2(DDNLEN)
DDNAMES DC  XL8'00'
          DC  XL8'00'
          DC  XL8'00'
          DC  XL8'00'
          DC  CL8'MYDATA'
          DC  CL8'SYSPRINT'
DDNLEN EQU  *-DDNAMES
*****
* Page number area (first page will be 1)
*****
CNOP  2,4
PAGEPTR DC  AL2(PAGELEN)
PAGENO  DC  CL4'0001'
PAGELEN EQU  *-PAGEPTR

```

Figure 11. Application Interface Program Example (Part 2 of 3)

UIM

```
*****
* User Interaction Module area (UIM is called USRUIM) *
*****
CNOP 2,4
UIMPTR DC AL2(UIMLEN)
UIM DC CL8'USRUIIM'
UIMLEN EQU *-UIM
*****
* User area pointer area *
*****
CNOP 2,4
UAPTR DC AL2(UALEN)
UA@ DS A
UALEN EQU *-UA@
*****
* User area *
*****
USERAREA DC AL1(COM1) Length of first command
          DC C' WTO ''USRAIPGM INVOKING DFSMSdss'''
COM1 EQU *-USERAREA-1
SECNDCOM DC AL1(COM2) Length of first command
          DC C' DUMP DS(EXC(SYS1.**)) LOGINDD(DASD) OUTDD(TAPE)'
COM2 EQU *-SECNDCOM-1
THIRDCOM DC AL1(COM3) Length of third command
          DC C' IF LASTCC=0 -
COM3 EQU *-THIRDCOM-1
FOURCOM DC AL1(COM4) Length of fourth command
          DC C' THEN DEFrag DDN(DASD)'
COM4 EQU *-FOURCOM-1
ENDCOMM DC X'00' End of command flag
*****
* Register save area *
*****
SAVEAREA DS 18F
*
END
```

Figure 11. Application Interface Program Example (Part 3 of 3)

```
*****
* Module Name      = USRUIM
*
* Descriptive Name = DFSMSdss User Interaction Module Example
*
* Function        = Perform various functions at different exit
*                   points.
*
*   Exit Point    Operation
*   -----        -----
*     0           Initialize counters for task
*     1           Insert SYSIN records from user area
*     2           Insert table of accounting data
*     3           Count Tape Blocks Read
*     4           Count Logical Tape Blocks Read
*     5           Count Logical Tape Blocks Written
*     6           Count Tape Blocks Written
*     7           Count DASD Tracks Read
*     8           Count DASD Tracks Written
*     9           Nothing
*    10          Nothing
*    11          Insert a WTO message
*    12          Never allow ADR369D
*    13          Force Reblocking
*    14          Convert counts to printable characters
*    15          Nothing
*    16          Always bypass password protection for tape
*    17          Supply TAPE01 as a tape volser
*    18          Nothing
*    19          Save Tape DDNAME and volser which had error
*    20          Save Data Set name
*    21          Do not process temporary data sets
*    22          Bypass serialization of SYS1 data sets
*    23          End function if processing errors
*    24          End function if initialization fails
*
*    25          Nothing
*    26          Nothing
*****
*
USRUIM CSECT
USRUIM AMODE 31
USRUIM RMODE ANY
*****
* Registers with special uses
*
*****
* EIDBASE EQU 2           Base reg for ADREIB.
OPTION EQU 3             Reg to test option
WORKREG EQU 4            Work register
LENGTH EQU 5             Reg to get length of records
TEMPBASE EQU 6            Temporary base reg for exits
RC EQU 7                Register for return code
SYSPBASE EQU 8            Register for sysprint recs
MSGOFF EQU 9             Register for offsets to table
*****
* Save Registers and Establish Addressability.
*
USING *,15                Initial Addressability
STM 14,12,12(13)         Save regs
```

Figure 12. User Interaction Module Example (Part 1 of 11)

UIM

```
BALR 12,0           New base addressability
USING *,12
DROP 15
B    BEGIN
DC    CL16'CSECT - USRUIM'
*****
* Establish Addressability to EIDB *
*****
BEGIN   LR   15,13          Save caller SA pointer
        LA   13,SAVEAREA      Set my SA pointer
        ST   15,SAVEAREA+4    Backward Chain
        B    BEGIN
        ST   13,8(15)         Forward Chain
        USING ADREIB,EIDBASE Addressability to EIDB
        L    EIDBASE,0(,1)     Address of EIDB
        LH   OPTION,EIOPTION  Get DFSMSdss processing option
        SLL  OPTION,2          Multiply times four
        LA   15,VECTABLE       Point to vector table start
        L    15,0(OPTION,15)   Point to processor routine
        BALR 14,15             Go to processor routine
        L    13,SAVEAREA+4    Restore SA pointer
        RETURN (14,12)         Return to DFSMSdss
+
+     LM   14,12,12(13)    RESTORE THE REGISTERS 01-RETURN
+     BR   14                 RETURN 01-RETURN
*****
* Processing Routines *
*****
***** AIOPT00: Function Startup *****
* Initialize counters for tape and dasd accounting. *
* Insert message identifying task into table *
*****
AIOPT00 XC   RTBCNT,RTBCNT      Init Tape Blocks Read
        XC   WTBCNT,WTBCNT      Init Tape Blocks Written
        XC   RLTBCNT,RLTBCNT    Init Log Tape Blocks Read
        XC   WLTBCNT,WLTBCNT    Init Log Tape Blocks Written
        XC   RDTCNT,RDTCNT      Init Disk Tracks Read
        XC   WDTCNT,WDTCNT      Init Disk Tracks Written
        ICM  WORKREG,15,EITSKID  Get current task identifier
        CVD  WORKREG,CVWORK     Convert to decimal
        UNPK OPT00TSK(2),CVWORK+6(2) Unpack into message insert
        OI   OPT00TSK+1,X'F0'    Make last character printable
        L    MSGOFF,MSGCOUNT    Get current MSGCOUNT
        MH   MSGOFF,MSGLEN      Multiply by MSGLEN to get
        LA   WORKREG,MSGTABLE   offset into message table
        LA   TEMPBASE,0(MSGOFF,WORKREG)
        USING TABLEMAP,TEMPBASE
        MVC  TABENTRY(133),OPT00MSG Move into msgtable
        DROP TEMPBASE
        L    WORKREG,MSGCOUNT   Increment MSGCOUNT
        LA   WORKREG,1(WORKREG)
        ST   WORKREG,MSGCOUNT   Save new count
        LH   RC,EIRC00          Set normal process retcode
        STH  RC,EIRETCOD       Store retcode in EIDB
        BR   14                 Return to intercept processor
*****
* AIOPT01: Read SYSIN Record *
* Get SYSIN records from the user area and give them to DFSMSdss *
* using the Insert Record return code. *
*****
```

Figure 12. User Interaction Module Example (Part 2 of 11)

AIOP01	TM	SYSINFL,SYSIFRST	Q. First entry to SYSIN exit?
	BO	NXTSYS	A. No, get next input
	OI	SYSINFL,SYSIFRST	Indicate first time taken
	ICM	TEMPBASE,15,EIUSEPTR	Get address of UA
	ST	TEMPBASE,CURCOMM	Save pointer to first record
NXTSYS	L	TEMPBASE,CURCOMM	Get current command pointer
	LA	WORKREG,1(,TEMPBASE)	Get past length
	STCM	WORKREG,15,EIRECPTR	Initialize the pointer
	SR	LENGTH,LENGTH	Clear workreg
	IC	LENGTH,0(,TEMPBASE)	Get length of record
	LTR	LENGTH,LENGTH	Q. End of SYSIN data?
	BZ	SYSDEL	A. Yes, bypass SYSIN DS
	STCM	LENGTH,15,EIRECLEN	Set record length
	LA	TEMPBASE,1(LENGTH,TEMPBASE)	Point to new record
	ST	TEMPBASE,CURCOMM	Save new pointer
	LH	RC,EIRC08	Set insert record retcode
	B	SYSEXIT	Continue setup
SYSDEL	ICM	LENGTH,15,EIRECLEN	Get current record length
	LTR	LENGTH,LENGTH	Q. EOF condition ? (reclen=0)
	BZ	SYSRC00	A. Yes, process it.
	LH	RC,EIRC12	Set delete record retcode
	B	SYSEXIT	Continue Setup
SYSRC00	LH	RC,EIRC00	
SYSEXIT	STH	RC,EIRETCOD	Store retcode in EIDB
	BR	14	Return to intercept processor
<hr/>			
* AIOP02: Write SYSPRINT *			
* Insert SYSPRINT records before message ADR012I *			
<hr/>			
AIOP02	ICM	SYSPBASE,15,EIRECPTR	
	USING	SYSPMAP,SYSPBASE	
	CLC	MSGID(8),ADR012I	Q. Last SYSPRINT rcd?
	BNE	NOTLAST	A. No, don't insert records
	TM	SYSPRFL,INSDONE	Q. Are we done inserting?
	BO	NOTLAST	A. Yes, don't insert records
	L	WORKREG,INSERTCT	
	LR	MSGOFF,WORKREG	Save INSERTCT for offset
	LA	WORKREG,1(WORKREG)	Increment INSERTCT and
	C	WORKREG,MSGCOUNT	Q. Have we printed all rcds?
	BNH	INSERT	A. No, insert another record
	OI	SYSPRFL,INSDONE	Indicate we are done insert
	LA	WORKREG,TRAILER	Point to trailer record
	STCM	WORKREG,15,EIRECPTR	Store into EIRECPTR
	LH	WORKREG,MSGLEN	Update EIRECLEN
	STCM	WORKREG,15,EIRECLEN	
	LH	RC,EIRC08	Set insert record retcode
	B	SYSPEXIT	
INSERT	MH	MSGOFF,MSGLEN	Multiply by MSGLEN to get
	LA	WORKREG,MSGTABLE	offset into message table
	LA	WORKREG,0(MSGOFF,WORKREG)	
	STCM	WORKREG,15,EIRECPTR	Store address in EIRECPTR
	LH	WORKREG,MSGLEN	
	STCM	WORKREG,15,EIRECLEN	Store length in EIRECLEN
	L	WORKREG,INSERTCT	Increment MSGCOUNT
	LA	WORKREG,1(WORKREG)	
	ST	WORKREG,INSERTCT	Save new count
	LH	RC,EIRC08	Set insert record retcode
	B	SYSPEXIT	
NOTLAST	LH	RC,EIRC00	Set normal process retcode
SYSPEXIT	STH	RC,EIRETCOD	Store retcode in EIDB
	BR	14	Return to intercept processor

Figure 12. User Interaction Module Example (Part 3 of 11)

UIM

```
*****
* AIOPT03: Read Tape Block Exit *
* Count Tape Block records read *
*****
AIOPT03 LH WORKREG,RTBCNT      Increment Read Tape Block
        LA WORKREG,1(WORKREG)    count
        STH WORKREG,RTBCNT
        LH RC,EIRC00            Set normal process retcode
        STH RC,EIRETCOD         Store retcode in EIDB
        BR 14                  Return to intercept processor
*****
* AIOPT04: Read Logical Tape Block Exit *
* Count Logical Tape Block records read *
*****
AIOPT04 LH WORKREG,RLTBCNT      Increment Read logical Tape
        LA WORKREG,1(WORKREG)    Block count
        STH WORKREG,RLTBCNT
        LH RC,EIRC00            Set normal process retcode
        STH RC,EIRETCOD         Store retcode in EIDB
        BR 14                  Return to intercept processor
*****
* AIOPT05: Write Logical Tape Block Exit *
* Count Logical Tape Block records written *
*****
AIOPT05 LH WORKREG,WLTBCNT      Increment Write logical Tape
        LA WORKREG,1(WORKREG)    Block count
        STH WORKREG,WLTBCNT
        LH RC,EIRC00            Set normal process retcode
        STH RC,EIRETCOD         Store retcode in EIDB
        BR 14                  Return to intercept processor
*****
* AIOPT06: Write Tape Block Exit *
* Count Tape Block records written *
*****
AIOPT06 LH WORKREG,WTBCNT      Increment Write Logical
        LA WORKREG,1(WORKREG)    Tape Block count
        STH WORKREG,WTBCNT
        LH RC,EIRC00            Set normal process retcode
        STH RC,EIRETCOD         Store retcode in EIDB
        BR 14                  Return to intercept processor
*
*****
* AIOPT07: Read DASD Track Exit *
* Count DASD Tracks read *
*****
AIOPT07 LH WORKREG,RDTCNT      Increment Read DASD Track
        LA WORKREG,1(WORKREG)    count
        STH WORKREG,RDTCNT
        LH RC,EIRC00            Set normal process retcode
        STH RC,EIRETCOD         Store retcode in EIDB
        BR 14                  Return to intercept processor
*****
* AIOPT08: Write DASD Track Exit *
* Count DASD Tracks written *
*****
AIOPT08 LH WORKREG,WDTCNT      Increment Write DASD Track
        LA WORKREG,1(WORKREG)    count
        STH WORKREG,WDTCNT
        LH RC,EIRC00            Set normal process retcode
        STH RC,EIRETCOD         Store retcode in EIDB
        BR 14                  Return to intercept processor
```

Figure 12. User Interaction Module Example (Part 4 of 11)

```
*****
* AIOPT09: Read Utility SYSPRINT *
*****
AIOPT09 LH RC,EIRC00 Set normal process retcode
      STH RC,EIRETCOD Store retcode in EIDB
      BR 14 Return to intercept processor
*****
* AIOPT10: Write SYSPRINT Record *
*****
AIOPT10 LH RC,EIRC00 Set normal process retcode
      STH RC,EIRETCOD Store retcode in EIDB
      BR 14 Return to intercept processor
*****
* AIOPT11: Write WTO Message *
* Delete the WTO and insert my own *
*****
AIOPT11 LH WORKREG,WTOCT Check WTO count
      LTR WORKREG,WORKREG Q. Is WTOCT 0?
      BZ OPT11INS A. Yes, insert one
OPT11DEL LH RC,EIRC12 Set delete record retcode
      B OPT11XIT (delete all other WTOs)
OPT11INS LA WORKREG,1(WORKREG) Increment WTOCT
      STH WORKREG,WTOCT
      LA WORKREG,UIMWTO Get address of WTO
      STCM WORKREG,15,EIRECPTR Store it in EIRECPTR
      LA WORKREG,WTOLEN Get length of WTO
      STCM WORKREG,15,EIRECLEN Store length it in EIRECLEN
      LH RC,EIRC08 Set insert record retcode
      B OPT11XIT
OPT11XIT STH RC,EIRETCOD Store retcode in EIDB
      BR 14 Return to intercept processor
*****
* AIOPT12: Write WTOR Message *
* Check for ADR369D, never allow it to write over VTOC, etc *
*****
AIOPT12 ICM WORKREG,15,EIRECPTR
      CLC 12(8,WORKREG),ADR369D Q. Authorize request?
      BNE OPT12RC0 A. No, let DFSMSdss do WTOR
      LA WORKREG,UIMRESP Get address of response
      STCM WORKREG,15,EIRECPTR Store in EIRECPTR
      LA WORKREG,RESPLEN Get length of response
      STCM WORKREG,15,EIRECLEN Store in EIRECLEN
      LH RC,EIRC28 Set WTOR Response retcode
      B OPT12XIT
OPT12RC0 LH RC,EIRC00 Set normal process retcode
OPT12XIT STH RC,EIRETCOD Store retcode in EIDB
      BR 14 Return to intercept processor
*****
* AIOPT13: Present ADRUFO Record *
* If parm call, set IO and AI buffers above 16M *
* If function call and Copy, force reblocking. *
*****
AIOPT13 ICM TEMPBASE,15,EIRECPTR
      USING ADRUFOB,TEMPBASE Get addressability to UFO
      CLC UFFUNCT,PARM Q. Is this a PARM call?
      BE PARMCALL A. Yes, go to parm call
FUNCTION TM UFFUNCT1,UFFUCOPY Is this a copy?
      BNO OPT13RC0 No, don't change options
      SR WORKREG,WORKREG
      ICM WORKREG,3,UFBDYOFF Get offset for PARM list
      AR WORKREG,TEMPBASE Add to address of UFO
*****
```

Figure 12. User Interaction Module Example (Part 5 of 11)

UIM

```

USING UFOFUNCT,WORKREG      Establish addressability
OI    UF03FLGS,UFOFRBLK     Force reblocking
DROP  WORKREG
LH    RC,EIRC16              Set modify record retcode
B    OPT13XIT
PARMCALL SR    WORKREG,WORKREG This is a parm call.
ICM   WORKREG,3,UFBDYOFF    Get offset for PARM list
AR    WORKREG,TEMPBASE      Add to address of UFO
USING UFOFUNC,WORKREG      Establish addressability
OI    UFXAFLAG,UFXABUFF    Make sure IO buffers are >16M
OI    UFXAFLAG,UFAI31B     Make sure AI buffers are >16M
DROP  WORKREG
LH    RC,EIRC16              Set modify record retcode
LA    WORKREG,2               Initialize message table:
ST    WORKREG,MSGCOUNT     Initialize msgcount
DROP  TEMPBASE
LA    TEMPBASE,MSGTABLE    Get address of table
USING TABLEMAP,TEMPBASE
MVC   TABENTRY(133),HEADER  Move header into msgtable
LA    TEMPBASE,133(TEMPBASE) Move past first message
MVC   TABENTRY(133),BLAN    Move blank line into table
DROP  TEMPBASE
B    OPT13XIT               Exit.
OPT13RC0 LH    RC,EIRC00    Set normal process retcode
OPT13XIT EQU  *             *
STH   RC,EIRETCOD          Store retcode in EIDB
BR    14                    Return to intercept processor
*****
* AIOPT14: Function Ending *
*****
AIOPT14 LH    WORKREG,RTBCNT  Get Tape Blocks Read
CVD   WORKREG,CVDWORK      Convert to decimal
UNPK  RTBINS,CVDWORK      Unpack into message insert
OI    RTBINS+7,X'F0'        Make last character printable
LH    WORKREG,WTBCNT       Get Tape Blocks Written
CVD   WORKREG,CVDWORK      Convert to decimal
UNPK  WTBINS,CVDWORK      Unpack into message insert
OI    WTBINS+7,X'F0'        Make last character printable
LH    WORKREG,RLTCNT       Get Tape Logical Blocks Read
CVD   WORKREG,CVDWORK      Convert to decimal
UNPK  RLTBINS,CVDWORK     Unpack into message insert
OI    RLTBINS+7,X'F0'        Make last character printable
LH    WORKREG,WLTBCNT      Get Tape Log Blocks Written
CVD   WORKREG,CVDWORK      Convert to decimal
UNPK  WLTBINS,CVDWORK     Unpack into message insert
OI    WLTBINS+7,X'F0'        Make last character printable
LH    WORKREG,RDTCNT       Get Disk Tracks Read
CVD   WORKREG,CVDWORK      Convert to decimal
UNPK  RDTINS,CVDWORK      Unpack into message insert
OI    RDTINS+7,X'F0'        Make last character printable
LH    WORKREG,WDTCNT       Get Disk Tracks Written
CVD   WORKREG,CVDWORK      Convert to decimal
UNPK  WDTINS,CVDWORK      Unpack into message insert
OI    WDTINS+7,X'F0'        Make last character printable
L    MSGOFF,MSGCOUNT       Get current MSGCOUNT
MH    MSGOFF,MSGLEN        Multiply by MSGLEN to get
LA    WORKREG,MSGTABLE     offset into message table
LA    TEMPBASE,0(MSGOFF,WORKREG)

```

Figure 12. User Interaction Module Example (Part 6 of 11)

```

USING TABLEMAP,TEMPBASE
MVC TABENTRY(133),READMSG      Move into msgtable
LA TEMPBASE,133(TEMPBASE)       Increment pointer for 2nd msg
MVC TABENTRY(133),WRITEMSG     Move into msgtable
DROP TEMPBASE
L  WORKREG,MSGCOUNT           Increment MSGCOUNT
LA WORKREG,2(WORKREG)
ST  WORKREG,MSGCOUNT           Save new count
LH  RC,EIRC00                  Set normal process retcode
STH RC,EIRETCOD                Store retcode in EIDB
BR  14                         Return to intercept processor
*****
* AIOP15: Present WTOR Response *
*****
AIOP15 LH  RC,EIRC00           Set normal process retcode
STH RC,EIRETCOD                Store retcode in EIDB
BR  14                         Return to intercept processor
*****
* AIOP16: OPEN/EOF Tape Security and Verification *
* Always bypass password protection *
*****
AIOP16 ICM TEMPBASE,15,EIRECPTR Get address of record
LH  RC,EIRC16                  Bypass password protection
ST  RC,0(TEMPBASE)             Store in EIRECPTR
LH  RC,EIRC16                  Set modify record retcode
STH RC,EIRETCOD                Store retcode in EIDB
BR  14                         Return to intercept processor
*****
* AIOP17: OPEN/EOF Nonspecific Tape Mount *
* Supply DFSMSdss with TAPE01 *
*****
AIOP17 ICM TEMPBASE,15,EIRECPTR Get address of record
MVC 0(6,TEMPBASE),TAPE01       Supply TAPE01 as volser
LH  RC,EIRC16                  Set modify record retcode
STH RC,EIRETCOD                Store retcode in EIDB
BR  14                         Return to intercept processor
*****
* AIOP18: Insert logical VSAM Record during Logical Restore *
*****
AIOP18 LH  RC,EIRC00           Set normal process retcode
STH RC,EIRETCOD                Store retcode in EIDB
BR  14                         Return to intercept processor
*****
* AIOP19: Output Tape I/O Error *
* Save DDNAME, Volser, and Return code, print record. *
*****
AIOP19 ICM TEMPBASE,15,EIRECPTR Get address of EIRECORD
USING EIDDINFO,TEMPBASE        Est addressability to DDINFO
MVC OPT19DD,EIDDNAME           Move DDNAME to message
MVC OPT19VS,EIVOLID            Move VOLSER to message
SR  WORKREG,WORKREG            Clear out work register
IC  WORKREG,EIRETC             Get return code
CVD WORKREG,CVDWORK            Convert to decimal
UNPK OPT19RC(2),CVDWORK+6(2)  Unpack into message insert
OI  OPT19RC+1,X'F0'            Make last character printable
DROP TEMPBASE
L   MSGOFF,MSGCOUNT            Get current MSGCOUNT
MH  MSGOFF,MSGLEN              Multiply by MSGLEN to get
LA  WORKREG,MSGTABLE           offset into message table

```

Figure 12. User Interaction Module Example (Part 7 of 11)

```

LA TEMPBASE,0(MSGOFF,WORKREG)
USING TABLEMAP,TEMPBASE
MVC TABENTRY(133),OPT19MSG Move into msgtable
DROP TEMPBASE L WORKREG,MSGCOUNT Increment MSGCOUNT
LA WORKREG,1(WORKREG)
ST WORKREG,MSGCOUNT Save new count
LH RC,EIRC00 Set normal process retcode
STH RC,EIRETCOD Store retcode in EIDB
BR 14 Return to intercept processor
*****
* AIOPT20: Volume Information Exit *
* Save Data Set name *
*****
AIOPT20 ICM TEMPBASE,15,EIRECPTR Establish addressability to
USING EIREC20,TEMPBASE EIREC20
MVC OPT20DSN(44),EI20DSN Move data set name into msg
DROP TEMPBASE
L MSGOFF,MSGCOUNT Get current MSGCOUNT
MH MSGOFF,MSGLEN Multiply by MSGLEN to get
LA WORKREG,MSGTABLE offset into message table
LA TEMPBASE,0(MSGOFF,WORKREG)
USING TABLEMAP,TEMPBASE
MVC TABENTRY(133),OPT20MSG Move into msgtable
DROP TEMPBASE
L WORKREG,MSGCOUNT Increment MSGCOUNT
LA WORKREG,1(WORKREG)
ST WORKREG,MSGCOUNT Save new count
LH RC,EIRC00 Set normal process retcode
STH RC,EIRETCOD Store retcode in EIDB
BR 14 Return to intercept processor
*****
* AIOPT21: Data Set Verification Exit *
* Do Not Process TEMP Data Sets *
*****
AIOPT21 EQU *
ICM TEMPBASE,15,EIRECPTR Get record pointer
USING EIREC21,TEMPBASE Get addressability
CLC EI21DSN(4),TMPL If first 4 char of dsn='TEMP'
DROP TEMPBASE Release addressability
BNE LABEL1 Then
LH RC,EIRC36 Do not process data set
B LABEL2 Else
LABEL1 LH RC,EIRC00 Set normal return code
LABEL2 STH RC,EIRETCOD Store retcode in EIDB
BR 14 Return to intercept processor
*****
* AIOPT22: Bypass Verification Exit *
* Bypass Serialization of SYS1 Data Sets *
*****
AIOPT22 EQU *
ICM TEMPBASE,15,EIRECPTR Get record pointer
USING EIREC22,TEMPBASE Get addressability
CLC EI22DSN(4),SYSL If first 4 char of dsn='SYS1'
BNE LAB1 Then
LH RC,EIRC16 Bypass serialization of dsn
OI EI22FLGS,B'10000000' Set EI22BSER bit on
B LAB2 Else
DROP TEMPBASE Release addressability
LAB1 LH RC,EIRC00 Set normal return code
LAB2 STH RC,EIRETCOD Store retcode in EIDB
BR 14 Return to intercept processor

```

Figure 12. User Interaction Module Example (Part 8 of 11)

```
*****
* AIOPT23: Data Set Processed Notification Exit *
* End Function If Errors in Processing *
*****
AIOPT23 EQU *
    ICM TEMPBASE,15,EIRECPTR      Get record pointer
    USING EIREC23,TEMPBASE        Get addressability
    LH WORKREG,EI23DSRC         If data set processing RC^=0
    DROP TEMPBASE                Release addressability
    LTR WORKREG,WORKREG          Then
    BZ LABL1
    LH RC,EIRC32                 End function
    B LABL2
LABL1 LH RC,EIRCO0              Set normal return code
LABL2 STH RC,EIRETCOD           Store retnode in EIDB
    BR 14                         Return to intercept processor
*****
* AIOPT24: Concurrent Copy initialization complete *
* End function if initialization is not successful *
*****
AIOPT24 EQU *                  Start EIOP24 processing
    LH RC,EIRCO0                 Assume goodness
    ICM TEMPBASE,15,EIRECPTR     Get record pointer
    USING EIREC24,TEMPBASE       Get addressability
    LH WORKREG,EI24RTCD          Get the return code
    DROP TEMPBASE                Release addressability
    LTR WORKREG,WORKREG          If zero
    BZ OPT24END                 End function
    LH RC,EIRC32                 Else end the function
OPT24END STH RC,EIRETCOD        Store retnode in EIDB
    BR 14                         Return to intercept processor
*
*****
* AIOPT25: Backspace tape input *
*****
AIOPT25 EQU *                  Start EIOP25 processing
    LH RC,EIRCO0                 Set normal process retnode
    STH RC,EIRETCOD              Store retnode in EIDB
    BR 14                         Return to intercept processor
*****
* AIOPT26: Volume open for output *
*****
AIOPT26 EQU *                  Start EIOP26 processing
    LH RC,EIRCO0                 Set normal process retnode
    STH RC,EIRETCOD              Store retnode in EIDB
    BR 14                         Return to intercept processor
*****
* Start of local variables *
*****
*
*
CURCOMM DC F'0'                 Address of current command
SYSINFL DC X'00'                 SYSIN Flag
SYSIFRST EQU X'80'               First entry SYSIN Performed
SYSPRFL DC X'00'                 SYSPRINT Flag
INSDONE EQU X'80'               Done inserting SYSPRINT recs
TMPL DC C'TEMP'                 TEMP high-level qualifier
SYSL DC C'SYS1'                 SYS1 high-level qualifier
*
SAVEAREA DC 18F'0'               My save area
*
VECTABLE DC A(AIOPT00)            Function Startup Exit
VEC01   DC A(AIOPT01)             Read SYSIN Exit
```

Figure 12. User Interaction Module Example (Part 9 of 11)

UIM

VEC02	DC	A(AIOPT02)	Write SYSPRINT Exit
VEC03	DC	A(AIOPT03)	Read Tape Block Exit
VEC04	DC	A(AIOPT04)	Read Logical Tape Exit
VEC05	DC	A(AIOPT05)	Write Logical Tape Exit
VEC06	DC	A(AIOPT06)	Write Tape Block Exit
VEC07	DC	A(AIOPT07)	Read DASD Track Exit
VEC08	DC	A(AIOPT08)	Write DASD Track Exit
VEC09	DC	A(AIOPT09)	Read Utility Sysprint Exit
VEC10	DC	A(AIOPT10)	Write SYSPRINT Record
VEC11	DC	A(AIOPT11)	Write WTO Message Exit
VEC12	DC	A(AIOPT12)	Write WTOR Message Exit
VEC13	DC	A(AIOPT13)	Present ADRUFO Record Exit
VEC14	DC	A(AIOPT14)	Function Ending Exit
VEC15	DC	A(AIOPT15)	Present WTOR Response
VEC16	DC	A(AIOPT16)	OPEN/EOF Tape Sec/Ver Exit
VEC17	DC	A(AIOPT17)	OPEN/EOF NonSpec Tape Mount
VEC18	DC	A(AIOPT18)	Insert log VSAM Rcd -Restore
VEC19	DC	A(AIOPT19)	Output Tape I/O Error Exit
VEC20	DC	A(AIOPT20)	Volume Information Exit
VEC21	DC	A(AIOPT21)	Data Set Verification Exit
VEC22	DC	A(AIOPT22)	Bypass Verification Exit
VEC23	DC	A(AIOPT23)	Data Set Processed Exit
VEC24	DC	A(AIOPT24)	Concurrent Copy Init Complete
*			
VEC25	DC	A(AIOPT25)	Backspace tape input
VEC26	DC	A(AIOPT26)	Volume open for output
*			
UIMWTO	DC	C'>>> USRUIM is active <<<'	
WTOLEN	EQU	*-UIMWTO	Show that we are active
*			
UIMRESP	DC	C'T'	Reply T to ADR369
RESPLEN	EQU	*-UIMRESP	
*			
ADR369D	DC	CL8'ADR369D'	DFSMSdss Authorization WTOR
ADR012I	DC	CL8'ADR012I'	Last DFSMSdss message id
*			
MSGLEN	DC	H'133'	Length of messages
INSERTCT	DC	F'0'	Count of msgs given to DFSMSdss
MSGCOUNT	DC	F'0'	Count of messages in table
*			
RTBCNT	DC	H'0'	Count of Tape Blocks Read
WTBCNT	DC	H'0'	Count of Tape Blocks Written
*			
RLTBCNT	DC	H'0'	Count Log Tape Blocks Read
WLTBCNT	DC	H'0'	Count Log Tape Blocks Written
*			
RDTCNT	DC	H'0'	Count Disk Tracks Read
WDTCNT	DC	H'0'	Count Disk Tracks Written
*			
WTOCT	DC	H'0'	Count of WTOs
*			
PARM	DC	X'0000'	Parm call test for Opt 13
*			
TAPE01	DC	CL6'TAPE01'	Tape volser for Opt 17
*			
*			
HEADER	DC	C' **** Begin USRUIM Messages '	
	DC	CL(133-(*-HEADER))'*****'	
		*****'	

Figure 12. User Interaction Module Example (Part 10 of 11)

```

*
TRAILER DC   C' **** End USRUIM Messages '
DC   CL(133-(*-TRAILER))'*****'
*****'

*
BLANKS  DC   CL133' '           Blank message

*
READMSG DC   C'    Disk Tracks Read:   '
RDTINS  DC   C'XXXXXXX'
          DC   C'    Tape Blocks Read:   '
RTBINS  DC   C'XXXXXXX'
          DC   C'    Logical Tape Blocks Read:   '
RLTBINS DC   C'XXXXXXX'
          DC   CL(133-(*-READMSG))' '

*
WRITEMSG DC   C'    Disk Tracks Written: '
WDTINS  DC   C'XXXXXXX'
          DC   C'    Tape Blocks Written: '
WTBINS  DC   C'XXXXXXX'
          DC   C'    Logical Tape Blocks Written: '
WLTBINS DC   C'XXXXXXX'
          DC   CL(133-(*-WRITEMSG))' '

*
OPT00MSG DC   C'    Messages follow for task: '
OPT00TSK DC   C'XX'
          DC   CL(133-(*-OPT00MSG))' '

*
OPT19MSG DC   C'    Permanent Tape Error for DDNAME: '
OPT19DD  DC   C'XXXXXXX'
          DC   C' VOLSER: '
OPT19VS  DC   C'XXXXXX'
          DC   C' Return Code: '
OPT19RC  DC   C'XX'
          DC   CL(133-(*-OPT19MSG))' '

*
OPT20MSG DC   C'    EI20DSN = ''
OPT20DSN DC   C'XXXXXXX.XXXXXXX.XXXXXXX.XXXXXXX.XXXXXXX'
          DC   C' '''
          DC   CL(133-(*-OPT20MSG))' '

*
CVDWORK  DS   D           Workarea for decimal conversion

*
          ADREID0          Macro to map EIDB

*
          ADRUFO           Macro to map UFO

*
USRUIM   CSECT
MSGTABLE DS   CL13300      Room in table for 100 msgs

*
TABLEMAP DSECT
TABENTRY DS   CL133        DSECT to insert msgs into table

*
SYSPMAP  DSECT
          DS   CL1        DSECT to find MSGID in DFSMSdss
MSGID   DS   CL7         messages

*
          END

```

Figure 12. User Interaction Module Example (Part 11 of 11)

```
PAGE 0001 5695-DF175 DFSMSDSS V2R10.0 DATA SET SERVICES 1999.211 14:56
ADR010I (SCH)-PRIME(01), SIZE VALUE OF 4096K WILL BE USED FOR GETMAIN
WTO 'USRIMPDS INVOKING DFSMSdss'
ADR101I (R1)-R101 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'WTO '
DUMP DS (EXC(SYSL+*)) LOGIND(DASD) OUTDD(TAPE)
ADR101I (R1)-R101 (01), TASKID 002 HAS BEEN ASSIGNED TO COMMAND 'DUMP '
IF LASTCC=0 -
ADR101I (R1)-R101 (01), TASKID 003 HAS BEEN ASSIGNED TO COMMAND 'IF '
THEN DEFRAF DDN(DASD)
ADR101I (R1)-R101 (01), TASKID 004 HAS BEEN ASSIGNED TO COMMAND 'DEFRAF '
ADR101I (R1)-R101 (01), 1999.211 14:56:05 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED.
ADR050I (002)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE
ADR016I (002)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR006I (002)-STEND(01), 1999.211 14:56:05 EXECUTION BEGINS
ADR788I (002)-DIVSM(03), PROCESSING COMPLETED FOR CLUSTER PUBSEXMP.KSDS.S01, 100 RECORD(S) PROCESSED, REASON 0
ADR801I (002)-DTOSC(01), DATA SET SERIALIZATION IS COMPLETE. 4 OF 4 DATA SETS WERE SELECTED: 0 FAILED SERIALIZATION
AND 0 FAILED FOR OTHER REASONS.
ADR454I (002)-DTOSC(01), THE FOLLOWING DATA SETS WERE SUCCESSFULLY PROCESSED
PUBSEXMP.SAM.S01
PUBSEXMP.POS.S01
CLUSTER NAME PUBSEXMP.ESDS.S01
CATALOG NAME SYSL.MVRES.MASTCAT
COMPONENT NAME PUBSEXMP.ESDS.S01.DATA
CLUSTER NAME PUBSEXMP.KSDS.S01
CATALOG NAME SYSL.MVRES.MASTCAT
COMPONENT NAME PUBSEXMP.KSDS.S01.DATA
COMPONENT NAME PUBSEXMP.KSDS.S01.INDEX
ADR006I (002)-STEND(02), 1999.211 14:56:06 EXECUTION ENDS
ADR013I (002)-CLTSK(01), 1999.211 14:56:06 TASK COMPLETED WITH RETURN CODE 0000
ADR050I (004)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE
ADR016I (004)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR006I (004)-STEND(01), 1999.211 14:56:07 EXECUTION BEGINS
ADR208I (004)-EANAL(01), 1999.211 14:56:07 BEGINNING STATISTICS ON D9S060:
FREE CYLINDERS 000058
FREE TRACKS 000003
FREE EXTENTS 000002
LARGEST FREE EXTENT (CYL,TRK) 000053,0003
```

Figure 13. Output Resulting from Use of the UIM Exits (Part 1 of 2)

```
PAGE 0002 5695-DF175 DFSMSDSS V2R10.0 DATA SET SERVICES 1999.211 14:56
FRAGMENTATION INDEX 0.050
PERCENT FREE SPACE 97
ADR220I (004)-EANAL(01), INTERVAL BEGINS AT CC:HH 000001:0000 AND ENDS AT CC:HH 000006:0000
ADR209I (004)-EFRAG(01), 1999.211 14:56:07 MOVED EXTENT 001 FROM 00006:0000-00006:0004 TO 00001:0000-00001:0004 FOR
PUBSEXMP.ESDS.S01.DATA
ADR209I (004)-EFRAG(01), 1999.211 14:56:07 MOVED EXTENT 001 FROM 00006:0005-00006:0006 TO 00001:0005-00001:0006 FOR
PUBSEXMP.KSDS.S01.DATA
ADR209I (004)-EFRAG(01), 1999.211 14:56:07 MOVED EXTENT 001 FROM 00006:0007-00006:0008 TO 00001:0007-00001:0008 FOR
PUBSEXMP.SAM.S01
ADR209I (004)-EFRAG(01), 1999.211 14:56:07 MOVED EXTENT 001 FROM 00006:0009-00006:000A TO 00001:0009-00001:000A FOR
PUBSEXMP.PDS.S01
ADR209I (004)-EFRAG(01), 1999.211 14:56:08 MOVED EXTENT 002 FROM 00006:000B-00006:000B TO 00001:000B-00001:000B FOR
PUBSEXMP.PDS.S01
ADR213I (004)-EANAL(01), 1999.211 14:56:08 ENDING STATISTICS ON D9S060:
DATA SET EXTENTS RELOCATED 000005
TRACKS RELOCATED 000012
FREE CYLINDERS 000058
FREE TRACKS 000003
FREE EXTENTS 000001
LARGEST FREE EXTENT (CYL,TRK) 000058,0003
FRAGMENTATION INDEX 0.000
PAGE 0003 5695-DF175 DFSMSDSS V2R10.0 DATA SET SERVICES 1999.211 14:56
ADR212I (004)-EANAL(01), EXTENT DISTRIBUTION MAP FOR D9S060:
EXTENT *FREE SPACE BEFORE* *FREE SPACE AFTER* * ALLOCATED *
SIZE
IN NO. CUM. NO. CUM. NO. CUM.
TRACKS EXTS PCT/100 EXTS PCT/100 EXTS PCT/100
1 4 0.148
2 4 0.444
5 1 0.629
10 1 1.000
75 1 0.085
>499 1 1.000 1 1.000
ADR006I (004)-STEND(02), 1999.211 14:56:08 EXECUTION ENDS
ADR013I (004)-CLTSK(01), 1999.211 14:56:08 TASK COMPLETED WITH RETURN CODE 0000
***** Begin USRIM Messages *****
Messages follow for task: 00
Messages follow for task: 02
Disk Tracks Read: 00000014 Tape Blocks Read: 00000000 Logical Tape Blocks Read: 00000000
Disk Tracks Written: 00000000 Tape Blocks Written: 00000036 Logical Tape Blocks Written: 00000012
Messages follow for task: 04
Disk Tracks Read: 00000052 Tape Blocks Read: 00000000 Logical Tape Blocks Read: 00000000
Disk Tracks Written: 00000012 Tape Blocks Written: 00000000 Logical Tape Blocks Written: 00000000
***** End USRIM Messages *****
ADR012I (SCH)-DSSU (01), 1999.211 14:56:08 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0000
```

Figure 13. Output Resulting from Use of the UIM Exits (Part 2 of 2)

Appendix E. Data Set Attributes

This section lists various attributes that DFSMSdss can set or change for a given data set, and identifies where DFSMSdss gets the attribute information from.

Locate the data set attribute of interest in the first column of Table 19. The remaining columns indicate where and under which conditions DFSMSdss finds the attribute information. For example, the data set size is usually determined by the source data set, unless the preallocated target data set is larger. The ALLDATA(x) and ALLEXCP keywords have an effect on the data set size.

Table 19. Data Set Attributes and How They Are Determined.

Attribute	Is the Attribute Determined by...			
	...the Source Data Set?	...the Pre-Allocated Target?	...a Keyword?	...another Factor?
Data set name	Yes, if no RENAME or RENAMEU	No	RENAME or RENAMEU	No
Data set size	Yes, unless the preallocated target data set is larger	Yes, if it is big enough	ALLDATA(x) or ALLEXCP	No
Volumes	Yes, if doing a RESTORE and no output volumes were specified, no target exists, and not SMS-managed	Yes	Yes, if not SMS, OUTDDNAME(x,...) or OUTDYNAM(x,...)	If SMS-managed, ACS routines and SMS allocation choose volumes with most available space; if not SMS-managed, DFSMSdss chooses volumes with most available space
Data set location on volume	Yes, if no target and either ABSTR, PSU, POU, or DAU	Yes	FORCE can override ABSTR, PSU, POU, and DAU	DFSMSdss locates wherever space is available; DEFrag may move extents
PDS directory size (blocks)	Yes	No	No	No
PDSE directory size (blocks)	Yes	No	No	No
SMS Storage Class or Management Class	Yes, if no target and BYPASSACS is specified	Yes	Yes, if STORCLAS(x) or MGMTCLAS(x), or both, are specified with BYPASSACS	ACS routines if no target and BYPASSACS is not specified
SMS Data Class	Yes, if no target	Yes	No	No
BLKSZ	Yes, if REBLOCK is not specified and data set is not system reblockable	No	If REBLOCK keyword is specified, DFSMSdss chooses a new optimal blocksize	If system reblockable, DFSMSdss chooses a new optimal blocksize, or else the user can change blocksize with the installation reblock exit and can specify REBLOCK with the installation options exit
LRECL	Yes	No	No	No
RECFM	Yes	No	No	No
DSORG	Yes	No	No	No
Number of stripes	Yes, source must be striped	Yes for non-VSAM. No for VSAM	No	For nonguaranteed-space, determined by sustained data rate (SDR) in STORCLAS. For guaranteed-space, must have a nonzero SDR, then determined by number of output volumes supplied

Data Set Attributes

Table 19. Data Set Attributes and How They Are Determined. (continued)

Attribute	Is the Attribute Determined by...			
	...the Source Data Set?	...the Pre-Allocated Target?	...a Keyword?	...another Factor?
Number of volumes (VOLCOUNT)	Yes	Yes	VOLCOUNT can make a single volume source into a multivolume target, or change the number of volumes for a multivolume data set	Yes (see the COPY and RESTORE command VOLCOUNT parameter descriptions for specific information)
Number of extents	Yes, for imbedded extended KSDSs during physical data set restore	Yes	No	DFSMSdss always tries to consolidate during COPY/RESTORE. RELEASE may reduce the number of extents
PDS/PDSE	Yes, if no target or if doing a RESTORE	Yes, if doing a COPY	CONVERT(PDS(x)) or CONVERT(PDSE(x))	No
Cataloged	Yes, if RECATALOG(*) is specified, no target, and the user is not doing a physical data set restore	Yes	RECATALOG(x), CATALOG, UNCATALOG (applies to source only)	If SMS or VSAM, cataloged by default; if physical data set restore, only single volume non-VSAM is cataloged (if CATALOG is specified)
Allocation unit	Yes, if no target and TGTALLOC(SOURCE) specified or defaulted	Yes	TGTALLOC(x)	No
Free space in VSAM	Yes, if going to like device, nonVALIDATE, no dummy blocks, and CI and CA sizes do not change	Yes, if doing VSAM I/O and uses values in target catalog entry	VALIDATE, NOVALIDATE, FREESPACE	No
Security (RACF)	If no target and source was generic or discrete, and no applicable profile protecting new target, a new discrete will be defined	Yes	MENTITY, MVOLSER	Full RACF profile information (access lists) are not preserved
AIX data sets on VSAM clusters	If SPHERE is specified during COPY or DUMP and RESTORE, sphere and connections are preserved	No	SPHERE	No
GDS state	Yes, if no target and TGTGDS(source) specified	Yes	TGTGDS(x)	No
RLS BWO field	Yes, if no preallocated target and no UIM input	Yes, if no UIM input	No	UIM can pass a value in Exit 22 during logical RESTORE
RLS timestamps	Yes, if not a logical restore	No	No	For logical restore, if dumped using RLS access, timestamps reflect the time of the dump; otherwise the timestamps are zero
RLS recovery required	Yes, if no UIM input	No	No	UIM can pass a value in Exit 22 during logical RESTORE
RLS log parameter	Yes, if no preallocated target and no UIM input	Yes, if no UIM input	No	UIM can pass a value in Exit 22 during logical RESTORE
RLS log stream ID	Yes, if no preallocated target and no UIM input	Yes, if no UIM input	No	UIM can pass a value in Exit 22 during logical RESTORE

Appendix F. Customizing ISMF

There may be times when you wish to customize the interactive storage management facility (ISMF). This section lists the libraries that you can customize and some restrictions. For more information, refer to *z/OS DFSMS Using the Interactive Storage Management Facility*, which describes ISMF restrictions and customization with DFSMSdss.

ISMF Libraries That Can Be Customized

You can customize the following ISMF libraries:

Panel Library

ISMF allows you to make the following changes:

- Change the initial priming values that ISMF ships
- Change the default values for data entry panels
- Provide additional restrictions to values that are entered for certain fields on panels
- Remove fields from functional panels
- Change highlighting and color
- Change the format of the panel
- Modify existing functional panel text and help text
- Add new fields to panels
- Add new panels.

Message Library

You can modify existing messages and add new messages.

Skeleton Library

You can modify the job skeletons for ISMF commands and line operators.

Table Library

You can modify the Interactive System Productivity Facility (ISPF) command tables.

Load Library

You can modify the ISMF command and line operator tables. The tables are contained in nonexecutable CSECTs in the load library.

CLIST Library

You can modify the options on the CLIST CONTROL statement.

Restrictions to Customizing ISMF

General restrictions for ISMF customization include the following:

1. Before changing anything, you should make a backup copy of ISMF. Keep this unmodified version of the product for diagnostic purposes. IBM support and maintenance is provided only for the unmodified version of ISMF.
2. Do not delete or rename any of the parts of ISMF. Deleting or renaming severely affects processing and can cause ISMF to fail.
3. ISMF is copyrighted. Under the IBM licensing agreement you may modify ISMF for your own use. You may not, however, modify it for commercial resale.

Appendix G. Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer/Library Server versions of z/OS books in the Internet library at:

www.ibm.com/servers/eserver/zseries/zos/bkserv/

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

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

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

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

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA

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

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Programming interface information

This publication primarily documents information that is NOT intended to be used as a Programming Interface of DFSMSdss.

This publication also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of DFSMSdss. This information is identified where it occurs either by an introductory statement to a section by the following marking:

_____**Programming Interface information**_____

_____**End of Programming Interface information**_____

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

IBM Corporation
Information Enabling Requests
Dept. DZWA
5600 Cottle Road
San Jose, CA 95193
U.S.A.

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

Trademarks

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

AIX	IMS
CICS	MVS/ESA
DFSMSdfp	MVS/XA
DFSMSdss	OS/390
DFSMShsm	RACF
DFSMS/MVS	RAMAC
ES/9000	System/370
Enterprise Storage Server	System/390
FlashCopy	S/370
Hiperspace	S/390
IBM	z/OS
IBMLink	zSeries

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

Glossary

This glossary defines technical terms and abbreviations used in DFSMSdss documentation. If you do not find the term you are looking for, refer to the index of the appropriate DFSMSdss manual or view *IBM Dictionary of Computing*, located at

<http://www.ibm.com/networking/nsg/nsgmain.htm>

A

ABARS. Aggregate backup and recovery support.

ABEND. Abnormal end of task. End of a task, a job, or a subsystem because of an error condition that cannot be resolved by recovery facilities while the task is performed.

ABENDxxx. The keyword that identifies the abnormal end of DFSMSdss because of a system-detected error.

ABSTR. A subparameter of the SPACE parameter in a DD statement. It indicates that specified tracks be assigned to a data set.

ACCEPT processing. An SMP/E process necessary for installing the FMIDs. SMP/E ACCEPT processing uses JCL to accept the modules and macros necessary to run the FMIDs. The FMIDs are accepted into the DLIBs from the temporary data sets.

access method services. A multifunction service program that is used to manage both VSAM and non-VSAM data sets and integrated catalog facility or the ICF catalog. It is used to define data sets and allocate space for them; convert indexed-sequential data sets to key-sequenced data sets; modify data set attributes in the catalog; reorganize data sets; facilitate data portability between operating systems; create backup copies of data sets, data set records, and catalog entries; help make inaccessible data sets accessible; list the records of data sets and catalogs; define and build alternate indexes; and convert OS CVOLs and the ICF catalog to integrated catalog facility catalogs.

ACDS. Active control data set.

ACS. Automatic class selection.

ADSP. Automatic data set protection.

alias. An alternate name for a member of a partitioned data set.

ALLOC. A space allocation parameter that indicates type, such as cylinders or tracks.

alternate index. In systems with VSAM, a key-sequenced data set containing index entries organized by the alternate keys of its associated base data records. It provides an alternate means of locating records in the data component of a cluster on which the alternate index is based.

alternate index cluster. In VSAM, the data and index components of an alternate index.

APAR. Authorized program analysis report.

APF. Authorized program facility.

application interface. An interface used to invoke DFSMSdss from another program.

apply processing. In SMP and SMP/E, the process, initiated by the APPLY command, that places system modifications (SYSMODS) into the target system libraries.

attach. In programming, to create a task that can be performed asynchronously with the performance of the mainline code.

authorization. (1) The right granted to a user to communicate with or make use of a computer system. (2) The process of giving a user either complete or restricted access to an object, resource, or function.

authorized program analysis report (APAR). A request for correction of a problem caused by a suspected defect in a current unaltered release of a program.

automatic class selection (ACS). A mechanism for assigning SMS classes and storage groups.

automatic data set protection (ADSP). A system function, enabled by the SETROPTS ADSP specification and the assignment of the ADSP attribute to a user with ADDUSER or ALTUSER, that causes all permanent data sets created by the user to be automatically defined to RACF with a discrete RACF profile..

B

backout. The CICSVR function that you can use if CICS fails in the attempt to back out uncommitted changes on a VSAM sphere. Using information from the RCDS, CICSVR constructs a job to back out uncommitted changes on a VSAM data set as indicated on the log.

backup. The process of creating a copy of a data set to ensure against accidental loss.

backup while open. DFSMSdss can perform backup of data sets that are open for update for a long period of time (like CICS). DFSMSdss can perform a logical data set dump of these data sets even if another application has them serialized.

base cluster. In systems with VSAM, a key-sequenced or entry-sequenced data set over which one or more alternate indexes are built.

basic catalog structure (BCS). The name of the catalog structure in the integrated catalog facility environment. An integrated catalog facility catalog consists of a BCS and its related VSAM volume data sets (VVDSs).

basic direct access method (BDAM). An access method used to directly retrieve or update particular blocks of a data set on a direct access device.

basic partitioned access method (BPAM). An access method that can be applied to create program libraries in direct access storage for convenient storage and retrieval of programs.

basic sequential access method (BSAM). An access method for storing or retrieving data blocks in a continuous sequence, using either a sequential access or a direct access device.

BCS. Basic catalog structure.

BDAM. Basic direct access method.

BLK. A subparameter of the SPACE parameter in a DD statement. It specifies that space is allocated by blocks.

block length. Synonym for block size.

block size. (1) The number of data elements in a block. (2) A measure of the size of a block, usually specified in units such as records, words, computer words, or characters. (3) Synonymous with block length. (4) Synonymous with physical record size.

BPAM. Basic partitioned access method.

bpi. Bits per inch.

Broken data set. Data sets which do not conform to IBM data set standards are referred to as *broken*. Broken data sets are either missing catalog entries, VTOC entries, or VVDS entries; or, have invalid catalog entries, VTOC entries, or VVDS entries.

BSAM. Basic sequential access method.

C

CA. Control area.

call. (ISO) The action of bringing a computer program, a routine, or a subroutine into effect, usually by specifying the entry conditions and jumping to an entry point.

card image. A one-to-one representation of the hole patterns of a punched card; for example, a matrix in which a one represents a punch and a zero represents the absence of a punch.

CC-compatible SnapShot. See *virtual concurrent copy*.

CCHHR. Cylinder, cylinder, head, head, record.

CCW. Channel command word.

CDE. Contents directory entry.

CDS. Control data set.

channel command word (CCW). A doubleword at the location in main storage specified by the channel address word. One or more CCWs make up the channel program that directs data channel operations.

CI. Control interval.

CICS. Customer Information Control System.

CICSVR. CICS VSAM Recovery.

CICS VSAM Recovery (CICSVR). CICS VSAM Recovery is an IBM product that recovers your lost or damaged VSAM data. CICSVR V3.1 recovers VSAM data in the following environments:

- CICSVR VSAM batch logging (when the VSAM data sets are not accessed in record level sharing mode)
- CICS TS
- CICS V4

CLIST. Command list.

complete recovery. The CICSVR function that consists of forward recovery followed by backout, if needed. In CICSVR complete recovery, CICSVR restores a DFSMSHsm or DFSMSdss backup for you.

component identification keyword. The first keyword, represented as a number, in a set of keywords used to describe a DFSMSdss program failure.

compress. (1) To reduce the amount of storage required for a given data set by having the system replace identical words or phrases with a shorter token associated with the word or phrase. (2) To reclaim the unused and unavailable space in a partitioned data set that results from deleting or modifying members by moving all unused space to the end of the data set.

COMPRESS command. The DFSMSdss function that reduces partitioned data sets by taking unused space and consolidating it at the end of the data set.

compressed format. A particular type of extended-format data set specified with the (COMPACT) parameter of data class. VSAM can compress individual records in a compressed-format data set. SAM can compress individual locks in a compressed-format data set. See *compress*.

concatenation. An operation that joins two characters or strings in the order specified, forming one string whose length is equal to the sum of the lengths of the two characters or strings.

concurrent copy. A function to increase the accessibility of data by letting you make a consistent backup or copy of data concurrent with normal application program processing.

concurrent copy-compatible (CC-compatible) SnapShot. See *virtual concurrent copy*.

conditioned volume.. The target volume from a previous FULL volume COPY operation which specified DUMPCONDITONING.

control area (CA). A group of control intervals used as a unit for formatting a data set before adding records to it. Also, in a key-sequenced data set, the set of control intervals, pointed to by a sequence-set index record, that is used by VSAM for distributing freespace and for placing a sequence-set index record adjacent to its data.

control interval (CI). A fixed-length area of auxiliary storage space in which VSAM stores records. It is the unit of information transmitted to or from auxiliary storage by VSAM.

control volume (CVOL). A volume that contains one or more indexes of the catalog.

constructs. A collective name for data class, storage class, management class, and storage group.

CONVERTV command. The DFSMSdss function that converts volumes to and from Storage Management Subsystem management without data movement.

COPY command. The DFSMSdss function that performs data set, volume, and track movement.

CP. Control program.

CREDT. Creation date.

CSW. Channel status word.

CVAF. Common VTOC access facility.

CVOL. Control volume.

CVT. Communication vector table.

CYL. A subparameter of the SPACE parameter in a DD statement. It specifies that space is allocated by cylinders.

D

DADSM. The direct access space management program that maintains the VTOC, VTOCIX, and space on a volume.

DAM. Direct access method.

DASD. Direct access storage device.

DASD ERP. DASD error recovery procedure.

DASD volume. A DASD space identified by a common label and accessed by a set of related addresses.

data class. A list of data set allocation parameters and the values that are used when allocating a new SMS-managed data set.

data compression (run-length). A method of encoding repetitive series of identical characters so that they occupy less space on a dump tape. Data compression is supported by both physical dump and logical dump processing.

Data Facility Storage Management Subsystem/MVS (DFSMS/MVS). The complementary functions of DFSMSdfp, DFSMSdss, DFSMShsm, and DFSMSrmm which, together with RACF provide a system-managed, administrator-controlled storage environment.

data set backup. Backup to protect against the loss of individual data sets.

data set change indicator. A bit that is set by OPEN when the data set is opened for processing other than input. This flag is supported on MVS systems that have data-set-changed flag support installed.

data set FlashCopy. One of the FlashCopy Version 2 functions. See also *FlashCopy Version 2*.

DAU. Direct access unmovable.

DB2. IBM DATABASE 2.

DCB. Data control block.

DEFRAG command. The DFSMSdss function that consolidates the free space on a volume to help prevent out-of-space abends on new allocations.

DEQ. An assembler language macro instruction used to remove control of one or more serially reusable resources from the active task.

DFSMS. Data Facility Storage Management Subsystem.

DFSMS environment. An environment that helps automate and centralize the management of storage. This is achieved through a combination of hardware, software, and policies. See also *system-managed storage*.

DFSMSdfp. A DFSMS/MVS functional component that provides functions for storage management, data management, program management, device management, and distributed data access.

DFSMSdss. A DFSMS/MVS functional component used to copy, move, dump, and restore data sets and volumes. DFSMSdss is the primary data mover of DFSMS/MVS.

DFSMShsm. A DFSMS/MVS functional component used for backing up and recovering data, and managing space on volumes in the storage hierarchy.

DFSMS/MVS. Data Facility Storage Management Subsystem/MVS.

DFSORT. Data Facility Sort.

DIAGNOSE. An access method services command that scans an integrated catalog facility basic catalog structure (BCS) or a VSAM volume data set (VVDS) to validate the data structure.

DIRF. DADSM interrupt recording facility. If a system fails, or a permanent I/O error occurs during allocation of space or during performance of a routine that updates the VTOC, the VTOC may be in error. To ensure that an error is recorded, the DADSM routines turn on a bit in the VTOC upon entry to a DADSM function, and, if no errors occur during processing, turn off that bit upon exiting from that function.

distribution libraries. IBM-supplied partitioned data sets on tape containing one or more components that the user restores to disk for subsequent inclusion in a new system.

DLIB. Distribution library.

DOC. In diagnosing program failures, the keyword that identifies an error in the documentation of a program.

DOS. Disk Operating System.

DOS bit. On a volume without an indexed VTOC, a bit that indicates that the free space map is invalid.

DOS/VSE. DOS/Virtual Storage Extended.

DSCB. Data set control block.

DSCHA. A DFSMSdss keyword that is used in BY filtering. It indicates that the data set is to be selected if the data set has been changed.

dsname. Data set name.

DSORG. Data set organization. It is specified in the JCL as "DSORG=".

DUMP command. The DFSMSdss function used to back up data sets, tracks, and volumes.

dynamic allocation. Assignment of system resources to a program when the program is performed rather than when it is loaded main storage.

E

early warning system (EWS). A microfiche copy of the information contained in the software support facility (SSF), organized by component identification number, and indexed by APAR symptom code. EWS is published monthly and available to customers of IBM licensed programs.

ECB. Event control block.

EC mode. Engineering change mode.

empty data set. A data set in which the pointer to the last-used block is 0.

ENQ. An assembler language macro instruction that requests the control program to assign control of one or more serially reusable resources to the active task. It is also used to determine the status of a resource; that is, whether it is immediately available or in use, and whether control has been previously requested for the active task in another ENQ macro instruction.

entry-sequenced data set (ESDS). In VSAM, a data set whose records are loaded without respect to their contents and whose RBAs cannot change. Records are retrieved and stored by addressed access, and new records are added at the end of the data set.

EOF. End-of-file.

EOJ. End of job.

erase-on-scratch. The physical erasure of data on a DASD data set when the data set is deleted (scratched).

ESA. Enterprise Systems Architecture.

ESS. Enterprise Storage Server.

ESDS. Entry-sequenced data set.

ESTAE. Extended specify task abnormal exit.

EQ. Equal to.

EWS. Early warning system.

EXCP. Execute channel program.

execute channel program (EXCP). A macro used to access a data set without specifying the organization.

EXPDT. Expiration date.

extended format. The format of a data set that has a data set name type (DSNTYPE) of EXTENDED. The data set is structured logically the same as a data set that is not in extended format but the physical format is different. Data sets in extended format can be striped or compressed. Data in an extended format VSAM KSDS can be compressed. See also *striped data set* and *compressed format*.

extended specify task abnormal exit (ESTAE). A task recovery routine that provides recovery for those programs that run enabled, unlocked, and in task mode.

extent. A continuous space on a DASD volume occupied by a data set or portion of a data set. An extent of a data set contains a whole number of control areas.

F

FC. CVAF function code.

FCEC. CVAF function-error code.

filtering. The process of selecting data sets based on specified criteria. These criteria consist of fully- or partially-qualified data set names, or of certain data set characteristics, or of both.

FlashCopy. A function of the Enterprise Storage Server (ESS) and DFSMSdss that provides instant data copying. When resources allow, DFSMSdss automatically selects FlashCopy.

FlashCopy V1. FlashCopy Version 1.

FlashCopy V2. FlashCopy Version 2.

FlashCopy Version 1. The initial FlashCopy feature provided by ESS. FlashCopy Version 1 is supported at the volume level. Both the source and target volumes must reside on the same logical subsystem (LSS). Each volume can be in one FlashCopy relationship.

FlashCopy Version 2. FlashCopy Version 2 provides enhancements to the existing FlashCopy Version 1 feature of ESS. These enhancements include data set FlashCopy, multiple relationship FlashCopy, incremental FlashCopy, improvement in FlashCopy establish time, elimination of LSS constraint, and consistency group support. The source and target volumes must reside in the same ESS. DFSMS exploits data set FlashCopy.

FMID. Function modification identifier.

forward recovery. The CICSVR function that reapplies all changes to the VSAM sphere since the last backup. CICSVR gets the information it needs to construct the recovery job from the RCDS. The contents of the logs

are applied to the VSAM sphere to return it to its exact state before the data was lost. With CICSVR forward recovery, CICSVR restores a DFMSHsm or DFSMSdss backup for you.

fragmentation index. The qualitative measure of the scattered free space on a volume.

fully-qualified data set name. A data set in which all the qualifiers are completely spelled out.

function modification identifier (FMID). A code that identifies the release levels of a program product.

FVL. Function vector list.

G

GDG. Generation data group.

GDS. Generation data set.

generalized trace facility (GTF). An optional OS/VS service program that records significant systems events, such as supervisor calls and start I/O operations, for the purpose of problem determination.

generation data group (GDG). A collection of historically related non-VSAM data sets that are arranged in chronological order; each data set is a generation data set.

generation data set. One generation of a generation data group.

global resource serialization (GRS). A component of z/OS used for serializing use of system resources and for converting hardware reserves on DASD volumes to data set enqueues.

GT. Greater than.

GTF. Generalized trace facility.

H

HFS. Hierarchical file system.

I

ICKDSE. Device Support Facilities.

IDCAMS. Access Method Services.

IDRC. Improved data recording capability.

IMS/VS. Information Management System/Virtual Storage.

INCORROUT. In diagnosing program failures, the keyword that identifies incorrect or missing program output.

incremental backup. A process in which data sets are backed up only if they have changed since their last backup.

installation exit. The means specifically described in an IBM software product's documentation by which an IBM software product may be modified by a customer's system programmers to change or extend the functions of the IBM software product. Such modifications consist of exit routines written to replace one or more existing modules of an IBM software product, or to add one or more modules or subroutines to an IBM software product, for the purpose of modifying (including extending) the functions of the IBM software.

integrated catalog facility. A facility by which VSAM data set volume-related fields are separated from the catalog and maintained in the VVDS on the volume on which the data set resides.

integrated catalog facility catalog. A catalog that is composed of a basic catalog structure (BCS) and its related volume table of contents (VTOC) and VSAM volume data sets (VVDSs).

Interactive Problem Control System (IPCS). A component of MVS that permits online problem management, interactive problem diagnosis, problem tracking, and problem reporting.

Interactive Storage Management Facility (ISMF). An interactive interface of DFSMS/MVS that allows users and storage administrators access to the storage management functions.

Interactive System Productivity Facility (ISPF). An IBM licensed program used to develop, test, and run application programs interactively. ISPF is the interactive interface for all storage management functions.

I/O. Input/output.

IPCS. Interactive Problem Control System.

IPL. Initial program load.

ISMF. Interactive Storage Management Facility.

ISAM. Indexed sequential access method.

ISMF. Interactive Storage Management Facility.

ISPF. Interactive Systems Productivity Facility.

ISPF/PDF. Interactive Systems Productivity Facility/Program Development Facility.

J

JCL. Job control language.

JES. Job entry subsystem.

JES2. An MVS subsystem that receives jobs into the system, converts them to internal format, selects them for operation, processes their output, and purges them from the system. In an installation site with more than one processor, each JES2 processor independently controls its job input, scheduling, and output processing.

JES3. An MVS subsystem that receives jobs into the system, converts them to internal format, selects them for operation, processes their output, and purges them from the system. In complexes that have several loosely coupled processing units, the JES3 program manages processors so that the global processor exercises centralized control over the local processors and distributes jobs to them via a common job queue.

JFCB. Job file control block.

job control language (JCL). A problem-oriented language used to identify the job or describe its requirements to an operating system.

job entry subsystem (JES). A system facility for spooling, job queuing, and managing I/O.

JSCB. Job step control block.

K

K. Kilobyte: 1 024 bytes.

key-sequenced data set. A VSAM file or data set whose records are loaded in ascending key sequence and controlled by an index. Records are retrieved and stored by keyed access or by addressed access, and new records are inserted in key sequence by means of distributed free space. Relative byte addresses can change because of control interval or control area splits.

keyword. A symptom that describes one aspect of a program failure.

KRDS. Keyrange data set. Also known as a key-sequenced data set with key ranges.

KSDS. Key-sequenced data set.

L

LASTCC. Last condition code.

LDS. Linear data set.

like devices. Devices that have the same track capacity and number of tracks per cylinder (for example, 3380 Model D, Model E, and Model K).

LINK. An assembler language macro instruction that causes control to be passed to a specified entry point.

The linkage relationship established is the same as that created by a basic assembler language (BAL) instruction.

link-pack area (LPA). An area of virtual storage that contains reentrant routines that are loaded at IPL (initial program load) time and can be used concurrently by all tasks in the system.

load module. A computer program in a form suitable for loading into main storage for operation.

load module library. A partitioned data set used to store and retrieve load modules.

logical DUMP operation (data set). A DUMP operation in which logical processing is performed.

logical processing (data set). Processing that treats each data set and its associated information as a logical entity. As an example, DFSMSdss processes an entire data set before beginning with the next one.

logical storage subsystem (LSS). Used internally by ESS to manage a set of logical volumes which are associated with an individual device adapter, e.g., a physical ESS subsystem may be partitioned into multiple logical storage subsystems.

logical RESTORE operation (data set). A RESTORE operation that uses as input a data set produced by a logical DUMP operation.

logical volume. The output produced from a physical DUMP operation, for which all data is derived from a single DASD volume.

LOOP. In diagnosing program failures, the keyword that identifies a program failure in which some part of the program repeats endlessly.

LPA. Link-pack area.

LSS.. Logical storage subsystem.

LT. Less than.

LRECL. Logical record length.

LVOL. Logical volume.

M

Mb. Megabit; 1 048 576 bits.

MB. Megabyte; 1 048 576 bytes.

maintenance-level keyword. In diagnosing program failures, a keyword that identifies the maintenance level of DFSMSdss.

management class. A list of the migration, backup, and retention parameters and the values for an SMS-managed data set.

map record. The record that maps the tracks dumped by DFSMSdss.

MAXCC. Maximum condition code.

MCS. Multiple console support.

MENTITY. Model entity.

minivolume. In an MVS system running on VM/370, an OS/VS-formatted VM/370 minidisk whose size is equal to or less than that of the real volume. DFSMSdss uses the device size specified in the VTOC. Minivolumes are supported only by the system version of DFSMSdss.

MSGADRnnnt. In diagnosing program failures, the DFSMSdss message keyword that tells of an error, or seems itself to be in error.

MVS. Multiple Virtual Storage.

N

NVR. Non-VSAM volume record.

O

operating system (OS). Software that controls the execution of programs; an operating system may provide services such resource allocation, scheduling, input/output control, and data management.

OS. Operating system.

P

pageable link-pack area (PLPA). Link-pack area.

PAM. Partitioned access method.

partially qualified data set name. A data set name in which the qualifiers are not spelled out. Asterisks and percent signs are used in place of the undefined qualifiers.

partitioned data set (PDS). A data set in direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

partitioned data set extended (PDSE). A system-managed, page-formatted data set on direct access storage. A PDSE contains an indexed directory and members similar to the directory and members of partitioned data sets. A PDSE can be used instead of a partitioned data set.

PDS. Partitioned data set.

PDSE. Partitioned data set extended.

PERFM. In diagnosing program failures, the keyword that identifies degradation in program performance.

physical DUMP operation (data set). A DUMP operation in which physical processing is performed.

physical processing (data set). Processing that moves data at the track-image level and can operate against volumes, tracks, and data sets. As an example, DFSMSdss may only process one volume of a multivolume data set.

PLPA. Pageable link-pack area.

POU. Partitioned organization unmovable.

PRB. Program request block.

private library. A user-owned library that is separate and distinct from the system library.

PSU. Physical sequential unmovable.

PSW. Program status word.

PTF. Program temporary fix.

Q

QSAM. Queued sequential access method.

qualified name. A data set name consisting of a string of names separated by periods; for example, “TREE.FRUIT.APPLE” is a qualified name.

qualifier. Each component name in a qualified name other than the rightmost name. For example, “TREE” and “FRUIT” are qualifiers in “TREE.FRUIT.APPLE.”

queued sequential access method (QSAM). An extended version of the basic sequential access method (BSAM). Input data blocks awaiting processing or output data blocks awaiting transfer to auxiliary storage are queued on the system to minimize delays in I/O operations.

R

RACE. Resource Access Control Facility.

RAMAC Virtual Array (RVA). A DASD that uses a virtual array architecture.

RB. Request block.

RBA. Relative byte address.

RCDS. Recovery Control Data Set.

RDJFCB. Read job file control block.

RECEIVE processing. An SMP/E process necessary to install new product libraries. During this process, the

code, organized as unloaded partition data sets, is loaded into temporary SMPTLIB data sets. SMP/E RECEIVE processing automatically allocates the temporary partitioned data sets that correspond to the files on the tape, and loads them from the tape.

RECFM. Record format.

recovery. The process of rebuilding data after it has been damaged or destroyed, often by restoring a backup version of the data or by reapplying transactions recorded in a log.

REFDT. A DFSMSdss keyword used in BY filtering. It indicates the last-referenced date.

relative byte address (RBA). The displacement (expressed as a fullword binary integer) of a data record or a control interval from the beginning of the data set to which it belongs, independent of the manner in which the data set is stored.

relative record data set (RRDS). A VSAM data set whose records are loaded into fixed-length slots.

RELEASE command. The DFSMSdss function that releases the unused space in sequential and partitioned data sets for use by other data sets.

RESERVE. A method of serializing DADSM update accesses to the VTOC. It is also a method of serializing processor accesses to a shared DASD volume.

Resource Access Control Facility (RACF). An IBM program product that provides for access control by identifying and verifying users to the system, authorizing access to DASD data sets, logging the detected unauthorized attempts to enter the system, and logging the detected accesses to protected data sets.

RESTORE command. The DFSMSdss function used to recover data sets, tracks, and volumes.

RMID. Replacement module identifier.

RNL. Resource name list.

RRDS. Relative record data set.

RVA. RAMAC Virtual Array.

run-length data compression. Data compression (run-length).

S

SAF. System authorization facility.

SAM. Sequential access method.

scheduler task. A DFSMSdss subtask that interprets and schedules commands.

SCP. System control program.

SEQ. Sequential or sequential processing.

sequential data striping. A software implementation of a disk array that distributes data sets across multiple volumes to improve performance.

SEREP. System environmental recording, editing, and printing

SMF. System management facilities.

SML. MVS Storage Management Library.

SMP. System Modification Program.

SMP/E. System Modification Program/Extended.

SMPE. A cataloged procedure that includes the required DD statements for running SMP/E and is used in the RECEIVE, APPLY, and ACCEPT steps of SMP/E processing.

SMS. Storage Management Subsystem.

SnapShot. A function of the RAMAC Virtual Array (RVA) that allows an instantaneous copy to be made of data sets using DFSMS software.

software support facility (SSF). An IBM online database that allows for storage and retrieval of information about all current APARs and PTFs.

SP. System Product.

sphere. A VSAM cluster with one or more associated alternate indexes and paths. The VSAM cluster (sometimes called the base cluster), alternate indexes, and paths are sometimes referred to as sphere components.

SSE. Software support facility.

Stand-Alone restore. One of two DFSMSdss programs. The Stand-Alone restore program runs independently of the MVS system environment and is limited to one function—a full or partial (tracks) RESTORE from a dump tape.

storage class. A named list of data set storage attributes that identify performance goals and availability requirements, defined by the storage administrator, used to select a device that can meet those goals and requirements.

storage constructs. The group of predefined models (data class, management class, storage class, and storage group) that are used to classify storage management needs and procedures for data sets under the Storage Management Subsystem. Each data set has construct names associated with it by explicit specification or defaulting.

storage group. A named collection of DASD volumes that have been grouped to meet a defined service strategy.

storage management. The task of managing auxiliary storage resources for an installation.

Storage Management Subsystem (SMS). An MVS subsystem that helps automate and centralize the management of storage. To manage storage, the storage management subsystem provides the storage administrator with control over data class, storage class, management class, storage group, and automatic class selection routine definitions.

stripe. In DFSMS, the portion of a striped data set, such as an extended format data set, that resides on one volume. The records in that portion are not always logically consecutive. The system distributes records among the stripes such that the volumes can be read from or written to simultaneously to gain better performance. Whether it is striped is not apparent to the application program.

striped data set. An extended format data set that occupies multiple volumes. A software implementation of sequential data striping.

striping. A software implementation of a disk array that distributes a data set across multiple volumes to improve performance.

subtask. A task initiated and ended by a higher order task.

SVC. Supervisor call instruction.

SVRB. Supervisor request block.

SYSRES. System residence disk

system data. The data sets required by MVS or its subsystems for initialization.

system-managed data set. A data set that has been assigned a storage class.

system-managed storage. Storage managed by the Storage Management Subsystem. SMS attempts to deliver required services for availability, performance, space, and security to applications.

system library. A collection of data sets or files in which the parts of an operating system are stored.

system link library. System library.

System Modification Program (SMP). A program used to install software and software changes on the MVS system.

System Modification Program Extended (SMP/E). An IBM licensed program used to install software and software changes on the MVS system. In addition to

providing the services of SMP, SMP/E consolidates installation data, allows more flexibility in selecting changes to be installed, provides a dialog interface, and supports dynamic allocation of data sets.

T

TCB. Task control block.

Time sharing option (TSO). An option on the operating system for a System/370 that provides interactive time sharing from remote terminals.

TIOT. Task input/output table.

TLIB. Target library.

track packing. A technique used by DFSMSdss that builds target tracks for any DASD device using input physical record information.

TRK. A subparameter of the SPACE parameter in a DD statement. It specifies that space is to be allocated by tracks.

TSO. Time sharing option.

TSO/E. TSO/Extensions.

TTR. Track-track-record.

type-of-failure keyword. In diagnosing program failures, a keyword that identifies the type of program failure that has occurred in DFSMSdss.

U

UACC. Universal access authority.

UCB. Unit control block.

UIM. User interaction module.

unlike devices. Devices that have different track capacities or a different number of tracks per cylinder.

used tracks. Tracks from the beginning of data sets to the last-used track.

user exit. A programming service provided by an IBM software product that may be requested by an application program for the service of transferring control back to the application program upon the later occurrence of a user-specified event.

V

VDRL. Volume restore limits.

VDSS. VTOC/Data Set Services.

virtual concurrent copy. An operation that uses SnapShot to provide a concurrent copy-like function when the source volume supports SnapShot, but not concurrent copy. (Also called CC-compatible SnapShot.)

virtual storage access method (VSAM). An access method for direct or sequential processing of fixed and variable-length records on direct access devices. The records in a VSAM data set or file can be organized in logical sequence by a key field (key sequence), in the physical sequence in which they are written on the data set or file (entry-sequence), or by the relative-record number.

VM. Virtual machine.

VOLID. Volume ID.

VOLSER. Volume serial number.

volume. The storage space on DASD, tape or optical devices, which is identified by a volume label.

volume backup. Backup of an entire volume to protect against the loss of the volume.

volume header record. The record in the DFSMSdss dump tape that identifies and contains data pertinent to the whole volume, and identifies the type of operation that created a dump.

volume trailer record. The record in the DFSMSdss dump tape that identifies the end of the data for a DASD volume.

VRRDS. A VSAM variable record RRDS.

VSAM. Virtual storage access method.

VSAM volume data set (VVDS). A data set that describes the VSAM and SMS-managed non-VSAM data sets on a volume. The name of the data set is SYS1.VVDS.Vvolser.

VSE. Virtual storage extended.

VTOC. Volume table of contents.

VTOCIX. The data set on which the location of the data set VTOC entries are kept in an index for quick access by DADSM. The name of the data set is SYS1.VTOCIX.Vvolser.

VVDS. VSAM volume data set.

VVR. VSAM volume record.

W

WAIT. In diagnosing program failures, the keyword that identifies DFSMSdss suspended activity, while waiting for some condition to be satisfied. DFSMSdss does not issue a message to tell why it is waiting.

WTO. Write to operator.

X

XA. Extended Architecture.

Z

zFS. See *zSeries File System*.

zSeries File System (zFS). A z/OS UNIX file system that can be used in addition to the hierarchical file system (HFS). zFS stores files in VSAM linear data sets. z/OS provides support for zFS in its Distributed File Service element.

Index

Special characters

- * (single asterisk) used in partially qualified data set names 12
- ** (double asterisk) used in partially qualified data set names 12
- % (percent sign) used in partially qualified data set names 12

Numerics

- 16 megabytes virtual storage
 - buffers above 8
- 31-bit addressing mode 314
- 3494 Tape Library 239
- 3495 Tape Library 239
- 64-bit addressing mode 8

A

- ABEND keyword 6
- ABOVE16 keyword 8
- access authorization for DFSMSdss
 - commands 271
- accessibility 365
- ACL (automatic cartridge loader) 242
- ACTIVE subkeyword of TGTGDS
 - COPY command 90
 - RESTORE command 212
- ADMINISTRATOR keyword 259, 265
 - COMPRESS command 32
 - COPY command 48
 - DEFrag command 108
 - DUMP command 128
 - FACILITY-class profiles for 266
 - PRINT command 159
 - RELEASE command 169
 - RESTORE command 185
- ADMINISTRATOR parameter
 - (Stand-Alone BUILDsa command) 25
- ADRDSU 303
- ADREID0 data area 331
- ADRMCLVL macro 343
- ADRUFO, presenting record - eioption 13 320
- ADRUIM module 306
- ADRXMAIA 307
- ADSP (Automatic Data Set Protection) 259, 261
- AIX (alternate index), recataloging 53
- ALLDATA keyword
 - COPY command 49
 - copying data set to like device 103
 - copying data set to unlike device 104
- DUMP command 128
- PRINT command 159
- ALLEXCP keyword
 - COPY command 50
 - copying data set to like device 103
 - copying data set to unlike device 104
- DUMP command 129

- ALLMULTI Keyword 285
- ALLOC keyword 14
- allocation space 89, 211
- alternate index, recataloging 53
- always-call 259, 262
- AMDSADM (SADM) service aid 246
- AMSGCNT keyword 7
- application interface
 - blocks 311
 - example 346
 - structure 305
 - summary
 - data set processing 330
 - record processing 330
- application program process 345
- ASIDPTR 305
- assignment of class names using the RESTORE and COPY commands 219
- asterisks, in a data set qualifier 12
- authorization levels for BUILDsa command 253
- automatic cartridge loader 242
- Automatic Data Set Protection
 - See ADSP
- AUTORELBLOCKADDRESS keyword
 - COPY command 50
 - RESTORE command 186
- auxiliary commands 229
 - continuation errors 235
 - EOJ 236
 - IF-THEN-ELSE 233
 - PARALLEL 230
 - SERIAL 230
 - SET 232
 - WTO 229
 - WTOR 229
- avoiding lockout 288

B

- backing up (dumping) data 121
- backspace physical tape record - eioption 25 329
- backup copies, multiple 105
- backup-while-open serialization
 - CICS 297
 - concurrent copy 300
 - DFSMStvs 297
 - general processing 299
 - IMS data sets 301
 - overview 297
 - status definitions 298
- BELOW16 keyword 8
- BLK subkeyword of TGTALLOC
 - COPY command 89
 - RESTORE command 211
- BLKSIZE and LRECL keywords, specifying in a print operation 161
- block size (DFSMSdss dump data set)
 - controlling at dump time 141

- block size (DFSMSdss dump data set)
 - (continued)
 - default when dumping to tape or DASD 141
 - with COPYDUMP command 105
- block structure, calling 303
- blocks, application interface 311
- broken data sets 19
- buffers above 16 megabytes 8
- buffers below 2 gigabyte real storage 8
- building a Stand-Alone Services IPL-able core image 24
- BY keyword
 - COMPRESS command 33
 - COPY command 51
 - criteria 11
 - DEFrag command 110
 - DUMP command 129
 - filtering criteria 19
 - RELEASE command 169
 - RESTORE command 186
- bypass verification exit - eioption 22 323
- BYPASSACS keyword
 - COPY command 51
 - RESTORE command 187

C

- calling block structure 303
- CANCELERROR keyword
 - COPY command 52
 - DUMP command 130
 - RESTORE command 187
- catalog
 - access authority 271
 - standard search order 19
- CATALOG keyword
 - CONVERTV command 38
 - COPY command 52
 - RESTORE command 188
- catalog management services 256
- CATLG keyword 14
- characteristics
 - data sets 11
 - filtering by 14
- CHECKVTOC keyword
 - COPY command 53
 - DUMP command 130
- CICSVRBACKUP
 - COPY command 54
 - DUMP command 130
- class selection, process diagram 219
- codes, condition
 - See condition codes
- command
 - IF-THEN-ELSE 234
 - null 234
 - syntax, general instructions 3
- commands
 - auxiliary 229

commands (*continued*)
 function 21
 overview 21
 compaction 131
 compatibility between DFSMSdss
 versions 285
 COMPRESS
 access authorization for 272
 subkeyword (DUMP command) 131, 135
 COMPRESS command
 ADMINISTRATOR keyword 32
 BY keyword 33
 DDNAME keyword 33
 DYNALLOC keyword 33
 DYNAM keyword 33, 34
 EXCLUDE keyword 34
 FILTERDD keyword 34
 INCLUDE keyword 35
 PASSWORD keyword 36
 sample operations 37
 WAIT keyword 36
 concurrent copy 83, 157
 dynamic pacing 83
 full-volume dump 152
 initialization 157
 initialization complete – eioption 24 328
 CONCURRENT keyword
 COPY command 54
 DUMP command 131
 condition codes
 general description 231
 LASTCC 232
 MAXCC 232
 setting 231
 conditioned volume 62
 CONSOLIDATE keyword (DEFrag command) 110
 continuation errors in auxiliary commands 235
 control area 72, 194
 control interval 72, 194
 controlling task processing 231
 CONVERT keyword (COPY command) 55
 CONVERTV command
 CATALOG keyword 38
 DDNAME keyword 39
 DYNAM keyword 39
 explanation 37
 FORCECP keyword 39
 INCAT keyword 38
 NONSMS keyword 40
 PREPARE keyword 40
 REDETERMINE keyword 40
 sample operations 41
 SELECTMULTI keyword 40
 serialization 295
 SMS keyword 41
 syntax diagram 38
 TEST keyword 41
 COPY command
 ACTIVE subkeyword of TGTGDS 90
 ADMINISTRATOR keyword 48
 ALLDATA keyword 49
 ALLEXCP keyword 50

COPY command (*continued*)
 AUTORELBLOCKADDRESS keyword 50
 BLK subkeyword of TGTALLOC 89
 BY keyword 51
 BYPASSACS keyword 51
 CANCELERROR keyword 52
 CATALOG keyword 52
 CHECKVTAC keyword 53
 CICVRBACKUP keyword 54
 CONCURRENT keyword 54
 CONVERT keyword
 partitioned data set 55
 partitioned data set extended 55
 COPYVOLID keyword 56
 CPVOLUME keyword 56
 CYL keyword 89
 data set to like device 103
 data set to unlike device 104
 DATASET keyword 56
 DEFERRED subkeyword of TGTGDS 90
 DELETE keyword 61
 DUMPCONDITONING keyword 62
 DYNALLOC keyword 63
 ENQFAILURE subkeyword 90
 example 9
 EXCLUDE keyword 63
 explanation 42
 FASTREPLICATION keyword 64
 FCNOCOPY keyword 65, 68, 69, 70
 FILTERDD keyword 64
 FORCE keyword 71
 FORCECP keyword 71
 FREESPACE keyword 72
 FULL keyword 72
 INCAT keyword 73
 INCLUDE keyword 73
 INDDNAME keyword 73
 INDYDAM keyword 74
 IOERROR subkeyword 90
 LOGINDDNAME keyword 74
 LOGINDYDAM keyword 75
 MAKEMULTI keyword 76
 MENTITY keyword 77
 MGMTCLAS keyword 78
 MVOLSER keyword 77
 NOPACKING command 78
 NULLMGMTCLAS keyword 78
 NULLSTORCLAS keyword 88
 OUTDDNAME keyword 79
 OUTDYDAM keyword 79
 OUTTRACKS keyword 80
 PASSWORD keyword 80
 PDS subkeyword 55
 PDSE subkeyword 55
 PERCENTUTILIZED keyword 81
 PROCESS keyword 81
 PURGE keyword 83
 READIOPACING keyword 83
 REBLOCK keyword 83
 RECATALOG keyword 52
 RELBLOCKADDRESS keyword 84
 RENAMEUNCONDITIONAL keyword 85
 REPLACE keyword 86

COPY command (*continued*)
 REPLACEUNCONDITIONAL keyword 87
 sample operations 96
 SHARE keyword 87
 SOURCE subkeyword of TGTALLOC 57, 89, 90, 111
 special considerations 43
 SPHERE keyword 88
 STORCLAS keyword 88
 STORGRP keyword 89
 syntax diagram 43
 SYS1 subkeyword 81
 TGTALLOC keyword 89
 TGTGDS keyword 90
 TOLERATE keyword 90
 TRACKS keyword 91
 TRK subkeyword of TGTALLOC 89
 TTRADDRESS keyword 92
 UNCATALOG keyword 92
 UNDEFINDSORG keyword 81
 VOLCOUNT keyword 92
 WAIT keyword 94
 WRITECHECK keyword 95
 copy operation for logical data set 43
 COPY, access authorization for 273, 277
 COPYDUMP command
 explanation 105
 INDDNAME keyword 105, 136
 LOGICALVOLUME keyword 105
 OUTDDNAME keyword 106
 sample operations 106
 syntax diagram 105
 COPYDUMP, access authorization for 277
 copying the Stand-Alone Services core image (example) 28
 COPYVOLID keyword
 COPY command 56
 RESTORE command 189
 CPVOLUME keyword
 COPY command 56
 DUMP command 132
 PRINT command 159
 RESTORE command 189
 CPYV keyword
 See COPYVOLID keyword
 CREDIT keyword 15
 criteria for filtering 11
 Cross-Memory Application Interface
 controlling DFSMSdss
 ASPACE parameter 309
 SNAPX parameter 310
 SRVRTIME parameter 308
 module main entry point 307
 restrictions 314
 usage guidelines 314
 customizing ISMF 363
 CYL subkeyword of TGTALLOC
 COPY command 89
 RESTORE command 211

D

DASD (direct access storage device) space fragmentation 107

DASDVOL
 access authority 259, 265, 278
 authority 256, 269

data integrity
 considerations
 COPY command 95
 DUMP command 151
 RESTORE command 179
 duplicate volser considerations 179

DYNALLOC keyword 292

ENQ keyword 289

RESERVE macro 287

RESERVE-ENQUEUE processing 296

serialization 287

WAIT option 289, 295

data set
 attribute sources 361
 broken 19
 BY filtering 19
 changed flag
 DUMP command 143
 characteristics (BY criteria) 11
 dummy 264
 filtering 11
 full and tracks restore 218
 moving 9
 name
 fully qualified 12
 partially qualified 12
 non-VSAM enqueue option 290
 processed notification exit - eoption 23 326
 profiles
 discrete 260
 generic 260
 qualifier, rules regarding asterisks in 12
 RESTORE 10
 SMS-managed data sets
 non-VSAM 217
 physical restore actions 217
 VSAM 217
 temporary 263
 to like device, ALLDATA and ALLEXCP interactions when copying 103
 to unlike device, ALLDATA and ALLEXCP interactions when copying 104
 verification - eoption 21 323

DATACLAS keyword 15

DATALENGTH keyword (PRINT command) 160

DATASET keyword
 COPY command 56
 DUMP command 132
 PRINT command 160
 RESTORE command 190

DCB keywords LRECL and BLKSIZE, specifying in a print operation 161

DDNAME keyword
 COMPRESS command 33
 CONVERTV command 39
 DEFrag command 110, 113
 RELEASE command 169

DDNAME list 303

DDNAME/VOLID Pointer 314

DDPTR parameter 303

DEBUG keyword 7

DEFERRED subkeyword of TGTGDS
 COPY command 90
 RESTORE command 212

DEFINE function of SAF (system authorization facility) 256

DEFrag
 data set 262

DEFrag command
 access authorization for 278
 ADMINISTRATOR keyword 108
 BY keyword 110
 CONSOLIDATE keyword 110
 DDNAME keyword 110
 DDNAME subkeyword 110
 DYNALLOC keyword 112
 DYNAM keyword 113
 EXCLUDE keyword 113
 explanation 107
 FASTREPLICATION keyword 114
 FORCECP keyword 115
 FRAGMENTATIONINDEX keyword 115
 LIST subkeyword 110
 MAXMOVE keyword 115
 PASSDELAY subkeyword 115
 PASSWORD keyword 117
 sample operations 118
 syntax diagram 107
 WAIT keyword 117
 WRITECHECK keyword 118

DELETE keyword
 COPY command 61
 DUMP command 132

DELETECATALOGENTRY keyword (RESTORE command) 190

determining version, release, modification level using the ADRMCLVL macro 343

DEVTYPE parameter (Stand-Alone TAPECNTL command) 252

DFM 62

DFP segment 264

DFSMSdss
 access authorization for commands 271
 compatibility with older versions 285
 module, main entry point 303
 pointer to address-space-identifier list 305
 processing option 311

DFSMStvs
 backup-while-open data sets 297
 backup-while-open status definitions 298

direct access data set
 copying 84
 restoring 204

disability 365

disk track
 reading - eoption 07 319
 writing - eoption 08 319

DO-END group of commands
 condition for processing 233
 syntax 233

DSCHA keyword 15

DSORG keyword 15

dummy data set 264

DUMP command
 access authorization for 278
 ADMINISTRATOR keyword 128
 ALldata keyword 128
 ALLEXCP keyword 129
 BY keyword 129
 CANCELERROR keyword 130
 CHECKVTOC keyword 130
 CICSVRBACKUP keyword 130
 COMPRESS command 131, 135
 CONCURRENT keyword 131
 CPVOLUME keyword 132
 data set changed flag 143
 data set example 10
 DATASET keyword 132
 DELETE keyword 132
 DYNALLOC keyword 133
 ENQFAILURE subkeyword 147
 EXCLUDE keyword 133
 explanation 121
 FCWITHDRAW keyword 134
 FILTERDD keyword 134
 FORCECP keyword 135
 FULL keyword 135
 INCAT keyword 136
 INCLUDE keyword 136
 INDDNAME keyword 136
 INDYNA keyword 137
 IOERROR subkeyword 147
 LOGINDNAME keyword 138
 LOGINDYNA keyword 139
 NOVALIDATE keyword 140, 149
 ONLYINCAT keyword 136, 140
 OPTIMIZE keyword 140
 OUTDDNAME keyword 141
 PASSWORD keyword 141
 PROCESS keyword 143
 PURGE keyword 143
 READIOPACING keyword 143
 RESET keyword 143
 sample operations 151
 SELECTMULTI keyword 75, 139
 SHARE keyword 145
 special considerations 121
 SPHERE keyword 146
 STORGRP keyword 146
 syntax diagram 121
 SYS1 subkeyword 143
 TOLERATE keyword 147
 TRACKS keyword 148
 UNCATALOG keyword 149
 VALIDATE keyword 149
 WAIT keyword 150
 with HFS data sets 290

dump data, multiple copies 105

DUMPCONDITONING keyword
 conditioned volume 62
 COPY command 62

DYNALLOC keyword 292
 COMPRESS command 33
 COPY command 63
 DEFrag command 112
 DUMP command 133
 PRINT command 160
 RELEASE command 170
 RESTORE command 192

DYNAM keyword
 COMPRESS command 34
 CONVERTV command 39
 DEFRAG command 113
 RELEASE command 170
 dynamic allocation 292

E

eioption 00 – function startup 315
 eioption 01 – reading SYSIN record 317
 eioption 02 – printing SYSPRINT record 317
 eioption 03 – reading physical tape record 317
 eioption 04 – reading logical tape record 318
 eioption 05 – writing logical tape record 318
 eioption 06 – writing physical tape record 318
 eioption 07 – reading disk track 319
 eioption 08 – writing disk track 319
 eioption 09 – reading utility SYSPRINT 319
 eioption 10 – writing SYSPRINT record 319
 eioption 11 – writing WTO message 320
 eioption 12 – writing WTOR message 320
 eioption 13 – presenting ADRUFO record 320
 eioption 14 – function ending 321
 eioption 15 – presenting WTOR response 321
 eioption 16 – OPEN/EOV tape volume security and verification exit 321
 eioption 17 – OPEN/EOV nonspecific tape volume mount 321
 eioption 18 – insert logical VSAM record during restore 322
 eioption 19 – output tape I/O error 322
 eioption 20 – volume notification 322
 eioption 21 – data set verification 323
 eioption 22 – bypass verification exit 323
 eioption 23 – data set processed notification exit 326
 eioption 24 – concurrent copy initialization complete 328
 eioption 25 – backspace physical tape record 329
 eioption 26 – IMS volume notification user exit 329
 ELSE command 234
 END command 234
 ending
 function – eioption 14 321
 your DFSMSdss step 236
 ENDTRK parameter (Stand-Alone RESTORE command) 249
 ENQ keyword 289
 ENQFAILURE subkeyword
 COPY command 90
 DUMP command 147
 PRINT command 162
 RESTORE command 212

enqueue
 compared to DYNALLOC 292
 exit routine options for non-VSAM data sets 290
 scheme 289
 EOJ command 236
 EQ operator 15
 erase-on-scratch 264
 ERRORTRACKS keyword (PRINT command) 160
 EXCLUDE keyword
 COMPRESS command 34
 COPY command 63
 criteria 11
 DEFRAG command 113
 DUMP command 133
 RELEASE command 170
 RESTORE command 193
 EXEC statement 5
 EXEC statement PARM information 6
 exit
 identification block 311
 interface, structure 311
 EXPDT keyword 15
 extended-sequential data set 95, 167, 216
 EXTNT keyword 15

F

FACILITY-class profiles for DFSMSdss keywords 257, 265
 FASTREPLICATION keyword
 COPY command 64
 DEFRAG command 114
 FCNOCOPY keyword
 COPY command 65, 68, 69, 70
 FCWITHDRAW keyword
 DUMP command 134
 FILE parameter (Stand-Alone RESTORE command) 250
 filter DD, JCL statement 6
 FILTERDD keyword
 COMPRESS command 34
 COPY command 64
 DUMP command 134
 RELEASE command 171
 RESTORE command 193
 filtering
 BY 19
 characteristics 14
 general description 11
 relative generation 14
 VSAM data sets 11
 FORCE keyword
 COPY command 71
 RESTORE command 193
 FORCECP keyword
 CONVERTV command 39
 COPY command 71
 DEFRAG command 115
 DUMP command 135
 RELEASE command 171
 RESTORE command 194
 FRAGMENTATIONINDEX keyword
 (DEFRAG command) 115
 free-space fragmentation 107

FREESPACE keyword
 COPY command 72
 RESTORE command 194
 FROMADDR parameter (Stand-Alone RESTORE command) 249
 FROMDEV parameter (Stand-Alone RESTORE command) 248
 FSIZE keyword 15
 FULL keyword
 COPY command 72
 DUMP command 135
 RESTORE command 194
 FULL parameter (Stand-Alone RESTORE command) 249
 fully qualified data set names 13
 function
 ending – eioption 14 321
 startup – eioption 00 315

G

GDG (generation data group)
 data set status assignment 90, 212
 filtering 14
 GE operator 15
 generating the Stand-Alone Services program 24
 generation data group
 See GDG
 global access checking table 259, 260
 GT operator 15

H

HFS data set
 coexistence considerations 285
 DUMP command considerations 145
 logical data set COPY 43
 logical dump and serialization 290
 physical dumb and serialization 291
 read/write serialization scheme 293
 RELEASE command consideration 167

I

I/O error, output tape – eioption 19 322
 ICKDSF REFORMAT command 26
 IF command, nesting 235
 IF-THEN-ELSE group of commands
 command sequencing 234
 description 231
 examples 235
 syntax 233
 IMPORT keyword (RESTORE command) 194, 283
 IMS volume notification user exit – eioption 26 329
 INCAT keyword
 CONVERTV command 38, 39
 COPY command 73
 DUMP command 136
 RELEASE command 171
 INCLUDE keyword
 COMPRESS command 35
 COPY command 73

INCLUDE keyword (*continued*)
 criteria 11
 DUMP command 136
 RELEASE command 171
 RESTORE command 196

INDDNAME keyword
 COPY command 73
 COPYDUMP command 105
 DUMP command 136
 PRINT command 160
 RESTORE command 196

INDDNAME parameter (Stand-Alone)
 BUILDSC command 25

INDYDAM keyword
 COPY command 74
 DUMP command 137
 PRINT command 161

initialization
 concurrent copy complete – eioption 24 328
 from concurrent copy failure 152, 157

input error toleration
 COPY command 90
 DUMP command 147
 PRINT command 162

input to DFSMSdss (in DD statement) 5

insert logical VSAM record during restore – eioption 18 322

integrity, data serialization 287

Interactive Storage Management Facility
 See ISMF

invoking DFSMSdss
 JCL examples 9
 via ATTACH, LINK, or CALL macro 306

IOERROR subkeyword
 COPY command 90
 DUMP command 147
 PRINT command 162

IPL
 example 245
 parameter (BUILDSC command) 26
 tape
 rewinding and unloading 252

IPLing Stand-Alone Services on a stand-alone system 243

ISMF (Interactive Storage Management Facility)
 customization 363
 restrictions to customizing 363

J

JCL (job control language)
 concurrent copy 152, 157
 EXEC statement 5
 filter DD statement 6
 input DD statement 5
 invoking DFSMSdss 9
 JOB statement 5
 output DD statement 5
 password DD statement 6
 requirements for DFSMSdss 5
 SYSIN DD statement 5
 SYSPRINT DD statement 5
 volume count subparameter 6

job control language
 See JCL

JOB statement JCL 5

K

keyboard 365

KEYLENGTH keyword (PRINT command) 161

L

LASTCC
 definition 231
 in IF-THEN-ELSE command sequence 233
 in SET command 232

LE operator 15

LINECNT keyword 7

LIST subkeyword
 DEFRAG command 110, 113

lockout, avoiding 288, 289

LOGDDNAME keyword (RELEASE command) 172

logical
 tape record, reading – eioption 04 318
 tape record, writing – eioption 05 318
 VSAM record, insert during restore – eioption 18 322

LOGICALVOLUME keyword
 COPYDUMP command 105
 RESTORE command 197

LOGINDDNAME keyword
 COPY command 74
 DUMP command 138

LOGINDYDAM keyword
 COPY command 75
 DUMP command 139

LookAt message retrieval tool vii

LRECL and BLKSIZE keywords, specifying in a print operation 161

LT operator 15

M

MAKEMULTI keyword
 COPY command 76
 RESTORE command 197

mapping macro, ADREID0 331

MAXCC keyword
 definition 231
 in IF-THEN-ELSE command sequence 233
 in SET command 232

MAXMOVE keyword (DEFRAG command) 115

MENTITY keyword 266, 275, 277, 281, 282
 COPY command 77
 RESTORE command 197

message data set 262

message retrieval tool, LookAt vii

MGMTCLAS keyword
 COPY command 78

MGMTCLAS keyword (*continued*)
 RESTORE command 198

MGMTCLAS profile 15, 264

MINSECQTY keyword (RELEASE command) 173

MINTRACKSUSED keyword (RELEASE command) 174

mode switching 230

moving data sets 9, 61

MULTI keyword 15

multiple backup copies 105, 141

multiple subkeywords 2

multiple tracks read, on dump 140

multivolume data set
 discrete profile 264

MVOLSER keyword
 COPY command 77
 RESTORE command 197

N

NE operator 15

nested IF commands 235

non-SMS authorization 271

non-VSAM data sets, restore actions 218

NONSMS keyword (CONVERTV command) 40

NOPACKING keyword
 COPY command 78
 RESTORE command 198

NOREADCHECK parameter (Stand-Alone RESTORE command) 250

NORUN keyword 8

notification
 data set processed exit – eioption 23 326
 IMS volume user exit – eioption 26 329
 volume – eioption 20 322

NOTIFYCONCURRENT keyword 46, 54, 125, 131

NOVALIDATE keyword (DUMP command) 140, 149

NOVERIFY parameter (Stand-Alone RESTORE command) 249

null command 234

NULLMGMTCLAS keyword
 COPY command 78
 RESTORE command 198

NULLSTORCLAS keyword
 COPY command 88
 RESTORE command 210

O

ONLYINCAT keyword
 COPY command 73
 DUMP command 140
 RELEASE command 171

OPEN/EOV
 nonspecific tape volume mount – eioption 17 321
 tape volume security and verification exit – eioption 16 321

operating in serial or parallel mode 230

OPERATIONS attribute 255, 256
 OPERCNSL parameter (Stand-Alone
 BUILDSEA command)
 limited validation of 239
 purpose of 239
 specifying 27
 OPTIMIZE keyword (DUMP
 command) 140
 option list 303
 options for non-VSAM data sets, enqueue
 exit routine 290
 OPTPTR parameter 303
 original record length 314
 original record pointer 314
 OUTDDNAME keyword
 COPY command 79
 COPYDUMP command 106
 DUMP command 141
 PRINT command 161
 RESTORE command 199
 OUTDYNAM keyword
 COPY command 79
 RESTORE command 200
 OUTDYNAM parameter (Stand-Alone
 BUILDSEA command) 27
 output from DFSMSdss (in DD
 statement) 5
 output tape I/O error – eioption 19 322
 OUTTRACKS keyword
 COPY command 80
 RESTORE command 201
 overview of the Stand-Alone Services
 process 242

P

page ejection, suppressing 7
 page-number list 304
 PAGENO keyword 7
 PAGEPTR parameter 304
 PARALLEL command 230
 PARAM keyword 305
 PARM information in the EXEC
 statement 6
 partially qualified data set names 12
 PASSDELAY subkeyword (DEFrag
 command) 115
 password DD, JCL statement 6
 PASSWORD keyword
 COMPRESS command 36
 COPY command 80
 DEFrag command 117
 DUMP command 141
 PRINT command 161
 RELEASE command 174
 RESTORE command 201
 password protection 258
 PATCH parameter definition 231
 PDS subkeyword (COPY command) 55
 PDSE subkeyword (COPY command) 55
 PERCENTUTILIZED keyword
 COPY command 81
 RESTORE command 202
 performing a restore from dump
 tapes 247
 physical tape record
 backspace – eioption 25 329

physical tape record (*continued*)
 reading – eioption 03 317
 writing – eioption 06 318
 PREPARE keyword (CONVERTV
 command) 40
 preparing for Stand-Alone Services 21,
 253
 presenting
 ADRUFFO record – eioption 13 320
 WTOR response – eioption 15 321
 PRINT command
 ADMINISTRATOR keyword 159
 ALLDATA keyword 159
 CPVOLUME keyword 159
 DATALENGTH keyword 160
 DATASET keyword 160
 DYNALLOC keyword 160
 ENQFAILURE subkeyword 162
 ERRORTRACKS keyword 160
 explanation 157
 INDDNAME keyword 160
 INDYDAM keyword 161
 IOERROR subkeyword 162
 KEYLENGTH keyword 161
 OUTDDNAME keyword 161
 PASSWORD keyword 161
 sample operations 164
 SHARE keyword 162
 syntax diagram 158
 TOLERATE keyword 162
 TRACKS keyword 163
 VTOC keyword 163
 WAIT keyword 164
 PRINT, access authorization for 279
 printing SYSPRINT record – eioption
 02 317
 PROCESS keyword
 COPY command 81
 DUMP command 143
 RELEASE command 175
 RESTORE command 202
 protect-all 262
 protected
 catalogs 270
 group data set 259
 user data set 259
 protecting
 data sets 257
 passwords 258
 resources 257
 SMS-managed data sets 264
 usage of DFSMSdss 257
 PURGE keyword
 COPY command 83
 DUMP command 143
 RESTORE command 202

Q

qualified data set names 12
 RACF (Resource Access Control
 Facility) 255
 authorization checking 77, 198

RACF (Resource Access Control Facility)
 (*continued*)
 DEFINE function 256
 logging 265
 RACF-indicated, meaning of 260
 RACF-protected, meaning of 255
 READCHECK parameter (Stand-Alone
 RESTORE command) 250
 reading
 disk track – eioption 07 319
 logical tape record – eioption 04 318
 physical tape record – eioption
 03 317
 SYSIN record – eioption 01 317
 utility SYSPRINT – eioption 09 319
 READIOPACING keyword 46, 48
 COPY command 83
 DUMP command 143
 REBLOCK keyword
 COPY command 83
 RESTORE command 203
 RECATALOG keyword
 COPY command 52
 RESTORE command 188
 recataloging an alternate index 53
 record area length 313
 REDETERMINE keyword (CONVERTV
 command) 40
 REFDT keyword 15
 relative generation filtering 14
 RELBLOCKADDRESS keyword
 COPY command 84
 RESTORE command 203
 RELEASE command
 ADMINISTRATOR keyword 169
 BY keyword 169
 cross-system serialization 170
 DDNAME keyword 169
 definition 166
 DYNALLOC keyword 170
 DYNAM keyword 170
 EXCLUDE keyword 170
 FILTERDD keyword 171
 FORCECP keyword 171
 INCAT keyword 171
 INCLUDE keyword 171
 LOGDDNAME keyword 172
 MINSECQTY keyword 173
 MINTRACKSUSED keyword 174
 ONLYINCAT keyword 171
 PASSWORD keyword 174
 PROCESS keyword 175
 sample operations 176
 SPHERE keyword 175
 STORGRP keyword 175
 syntax diagram 167
 SYS1 subkeyword 175
 WAIT keyword 176
 RELEASE, access authorization for 279
 RENAME keyword (RESTORE
 command) 204
 RENAMEUNCONDITIONAL keyword
 COPY command 85
 RESTORE command 206
 REPLACE keyword
 COPY command 86
 RESTORE command 207

REPLACEUNCONDITIONAL keyword
 COPY command 87, 208
 RESTORE command 87, 208
 REPLY keyword, in IF-THEN-ELSE
 command 233
 requirements
 real storage for Stand-Alone Services
 program operation 237
 RESERVE macro 287
 RESET keyword (DUMP command) 143
 Resource Access Control Facility
 See RACF
 resource serialization 296
 RESOWNER field 264
 RESTORE command
 access authorization for 279
 actions on non-VSAM data sets 218
 ACTIVE subkeyword of
 TGTGDS 212
 ADMINISTRATOR keyword 185
 AUTORELBLOCKADDRESS
 keyword 186
 BLK subkeyword of TGTLLOC 211
 BY keyword 186
 BYPASSACS keyword 187
 CANCELERROR keyword 187
 CATALOG keyword 188
 COPYVOLID keyword 189
 CPVOLUME keyword 189
 cross-system serialization 192
 CYL subkeyword of TGTLLOC 211
 data set example 10
 DATASET keyword 190
 DEFERRED subkeyword of
 TGTGDS 212
 DELETECATALOGENTRY
 keyword 190
 DYNALLOC keyword 192
 ENQFAILURE subkeyword 212
 EXCLUDE keyword 193
 explanation 177
 FILTERDD keyword 193
 FORCE keyword 193
 FORCECP keyword 194
 FREESPACE keyword 194
 FULL keyword 194
 IMPORT keyword 194, 283
 INCLUDE keyword 196
 INDDNAME keyword 196
 LOGICALVOLUME keyword 197
 MAKEMULTI keyword 197
 MENTITY keyword 197
 MGMTCLAS keyword 198
 MVOLSER keyword 197
 NOPACKING keyword 198
 NULLMGMTCLAS keyword 198
 NULLSTORCLAS keyword 210
 OUTDDNAME keyword 199
 OUTDYNAM keyword 200
 OUTTRACKS keyword 201
 PASSWORD keyword 201
 PERCENTUTILIZED keyword 202
 PROCESS keyword 202
 PURGE keyword 202
 REBLOCK keyword 203
 RECATALOG keyword 188
 RELBLOCKADDRESS keyword 203

RESTORE command (*continued*)
 RENAME keyword 204
 RENAMUNEUNCONDITIONAL
 keyword 206
 REPLACE keyword 207
 REPLACEUNCONDITIONAL
 keyword 87, 208
 ROLLEDOFF subkeyword of
 TGTGDS 212
 sample operations 220
 SHARE keyword 209
 SOURCE subkeyword of
 TGTLLOC 211
 SOURCE subkeyword of
 TGTGDS 212
 special considerations 178
 SPHERE keyword 209
 STORCLAS keyword 210
 syntax diagram 179
 TGTLLOC keyword 211
 TGTGDS keyword 212
 TOLERATE keyword 212
 TRACKS keyword 212
 TRK subkeyword of TGTLLOC 211
 TTRADDRESS keyword 213
 UNDEFINEDSORG subkeyword 202
 VOLCOUNT keyword 214
 WAIT keyword 216
 WRITECHECK keyword 216
 restore operation
 actions on non-VSAM data sets 218
 actions on SMS-managed data
 sets 217
 data set
 key-sequenced with no secondary
 allocation 178
 partitioned with secondary
 allocation of zero 178
 RACF-protected VSAM, logical
 restore of 178
 RENAME,
 RENAMUNEUNCONDITIONAL,
 REPLACE keywords 179
 REWIND parameter (Stand-Alone
 TAPECNTL command) 253
 RIOP keyword 46, 48
 ROLLEDOFF subkeyword of TGTGDS
 COPY command 90
 RESTORE command 212
 running Stand-Alone Services
 in 370 mode 237
 in XA or ESA mode 238
 program 242
 under VM 238
 with a predefined console 239
 RVA 55

S

SADMP service aid 246
 SAF (system authorization facility) 255
 SAM compressed data set, copying 43
 SCAN keyword 8
 SDUMP keyword 7
 search order, standard catalog 19
 SELECTMULTI keyword
 CONVERTV command 40

SELECTMULTI keyword (*continued*)
 DUMP command 75, 139
 separator, definition 1
 SERIAL command 230
 serialization
 avoiding lockout 288
 data integrity 287
 data set 289
 DYNALLOC keyword 292
 ENQ keyword 289
 Read/Write scheme 293
 RESERVE macro 287
 resource 296
 volume 287
 WAIT option 289, 295
 SET command 231, 232, 242
 sample operations 232
 setting
 condition codes 231
 patch bytes with SET PATCH
 command 232
 SHARE keyword
 COPY command 87
 DUMP command 145
 PRINT command 162
 RESTORE command 209
 shortcut keys 365
 SIZE keyword 7
 SMS
 authorization 271
 keyword (CONVERTV command) 41
 managed data sets 264
 MSGCNT keyword 7
 SnapShot 55
 RVA 131
 virtual concurrent copy mode 131
 SOURCE subkeyword of TGTLLOC
 COPY command 90
 RESTORE command 211, 212
 SOURCE subkeyword of TGTGDS
 COPY command 89
 source, definition 5
 space allocation 89, 211
 space fragmentation on DASD 107
 space utilization, percent
 COPY command 81
 RESTORE command 202
 SPECIAL attribute 255, 256
 SPHERE keyword
 COPY command 88
 DUMP command 146
 RELEASE command 175
 RESTORE command 209
 Stand-Alone Services feature 24, 237
 370 mode operation 237
 benefits 237
 building a Stand-Alone Services
 IPL-able core image 24
 BUILDSA command 24
 ADMINISTRATOR parameter 25,
 254
 determining authorization
 level 253
 examples 28
 INDDNAME parameter 25
 IPL parameter 26
 OPERCNSL parameter 27

Stand-Alone Services feature (*continued*)

- BUILDSA command (*continued*)
 - optional parameters 25
 - OUTDDNAME parameter 27
 - OUTDYNAM parameter 27
 - required parameters 25
 - syntax 25
- command sequence processing 242
- commands
 - BUILDSA 24
 - RESTORE 247
 - TAPECNTL 252
- controlling command sequence processing 242
- core image 24
 - definition of 21, 253
 - specifying information for 25
 - specifying the JCL output location for 27
 - specifying the output DASD volume 27
- DASD
 - devices in 370 mode 237
 - devices in XA or ESA mode 238
 - target device address, specifying 249
- determining BUILDSA command authorization level 253
- dump data set
 - device address, specifying 249
 - device type, specifying 248
- duplicate volser considerations 179
- EC mode operation 237
- ESA mode operation 238
- examples
 - BUILDSA command 28
 - copying the Stand-Alone Services core image 28
 - IPL 245
 - RESTORE command 251
 - TAPECNTL command 253
- extended control (EC) mode 237
- IF-THEN-ELSE command 242
- interference from other devices 238
- introduction 237
- IPLing on a stand-alone system 243
- IPLing with a tape library 240
- operator console
 - potential problems 239
 - predefining 237, 238, 239
- optional parameters
 - BUILDSA command 25
 - RESTORE command 249
- predefined console
 - potential problems with 246
 - purpose of 239
- preparing for 21, 253
- procedure to IPL Stand-Alone Services
 - on stand-alone system 243
- procedures to mount and demount
 - tapes using tape libraries 240
- process overview 242
- PSW wait-state codes 245
- purpose 237
- real storage requirements 237
- required parameters
 - BUILDSA command 25

Stand-Alone Services feature (*continued*)

- required parameters (*continued*)
 - RESTORE command 248
 - TAPECNTL command 252
- RESTORE command
 - description 247
 - ENDTRK parameter 249
 - example 251
 - FILE parameter 250
 - FROMADDR parameter 248, 249
 - FULL parameter 249
 - NOREADCHECK parameter 250
 - NOVERIFY parameter 249
 - optional parameters 249
 - READCHECK parameter 250
 - required parameters 248
 - STARTTRK parameter 249
 - syntax 248
 - TAPEVOLSER parameter 250
 - TOADDR parameter 249
 - VERIFY parameter 249
- restoring with a tape library 240
- rewinding a tape with the TAPECNTL command 252
- running in 370 mode 237
- running in XA or ESA mode 238
- running the program 242
- running with a predefined console 239
- security levels for BUILDSA command 253
- selecting the operator console 244
- setup Stand-Alone device feature 240
- specifying
 - information for the Stand-Alone Services core image 25
 - the DASD target device address 249
 - the date and time 244
 - the dump data set device address 249
 - the dump data set device type 248
 - the input device 244
 - the JCL output location for IPL-able core image 27
 - the message output device 244
 - the output DASD volume for IPL-able core image 27
 - where Stand-Alone Services is IPled from 24
- supported platforms 237
- syntax
 - BUILDSA command 25
 - RESTORE command 248
 - TAPECNTL command 252
- SYS1.ADR.SAIPLD.Vvolser data set 24
- SYS1.SADRYLIB target library 24
- System Modification Program (SMP) 24
- System Modification Program Extended (SMP/E) 24
- tape
 - library, setup Stand-Alone feature 240

Stand-Alone Services feature (*continued*)

- tape (*continued*)
 - library, transient mount 241
 - mount and demount
 - procedures 240
- tape library
 - setup Stand-Alone device feature 240
 - transient mount feature 240
- tape mount and demount
 - procedures 240
- TAPECNTL command
 - description 252
 - DEVTYPE parameter 252
 - example 253
 - required parameters 252
 - REWIND parameter 253
 - syntax 252
 - UNITADDR parameter 253
 - UNLOAD parameter 253
- transient mount feature 240
- using
 - tape library 239
 - using automatic cartridge loader 242
- VM, running under 238
- wait-state codes 245
- with a tape library 239
- with an automatic cartridge loader 242
- XA mode operation 238
- standard catalog search order 19
- standard naming conventions 262
- STARTTRK parameter (Stand-Alone RESTORE command) 249
- startup function – eoption 00 315
- status assignment, GDG data sets 90, 212
- storage
 - administrator 265
- STORCLAS keyword
 - COPY command 88
 - RESTORE command 210
- STORCLAS profile 264
- STORGRP keyword 16
 - COPY command 89
 - DUMP command 146
 - RELEASE command 175
- structure, exit interface 311
- subkeywords
 - maximum number 2
 - multiple 2
 - specifying 2
- supported
 - platforms for Stand-Alone Services 237
- suppressing page ejection 7
- switching modes 230
- syntax diagram
 - CONVERTV command 38
 - COPY command 43
 - COPYDUMP command 105
 - DEFrag command 107
 - DUMP command 121
 - PRINT command 158
 - reading a 3
 - RELEASE command 167
 - RESTORE command 179

SYS1 subkeyword
 COPY command 81
 DUMP command 143
 RELEASE command 175

SYSIN
 DD statement, JCL 5
 record, reading – eioption 01 317

SYSPRINT
 DD statement, JCL 5
 reading utility – eioption 09 319
 record printing – eioption 02 317
 writing record – eioption 10 319

system
 authorization facility
 See SAF
 programming information 310

T

tape
 libraries not supported in 370
 mode 237
 mount and demount procedures 240
 output I/O error – eioption 19 322
 physical record backspace – eioption 25 329
 reading logical record – eioption 04 318
 reading physical record – eioption 03 317
 volume mount OPEN/EVOC
 nonspecific – eioption 17 321
 volume security OPEN/EOV – eioption 16 321
 writing logical record – eioption 05 318
 writing physical record – eioption 06 318

tape devices (with Stand-Alone Services)
 in 370 mode 237
 in XA or ESA mode 238

TAPEVOLSER parameter (Stand-Alone RESTORE command) 250

target allocation 211

target class selection 219

target definition 5

task processing, controlling 231

TASK-ID 311

temporary
 data set 263
 data set names 262
 DEFrag data set 262
 message data set 262

TEST keyword (CONVERTV command) 41

TGTALLOC keyword
 COPY command 89
 RESTORE command 211

TGTGDS keyword
 COPY command 90
 RESTORE command 212

THEN command 234

TMPMSGDS keyword 7

TOADDR parameter (Stand-Alone RESTORE command) 249

TOLERATE keyword
 COPY command 90

TOLERATE keyword (*continued*)
 DUMP command 147
 PRINT command 162
 RESTORE command 212

TRACE keyword 8

TRACKS keyword
 COPY command 91
 DUMP command 148
 PRINT command 163
 RESTORE command 212

tracks read, on dump 140

TRK subkeyword of TGTALLOC
 COPY command 89
 RESTORE command 211

TTRADDRESS keyword
 COPY command 92
 RESTORE command 213

TYPRUN keyword 8

U

UAPTR parameter 305

UIMPTR parameter 304

UNCATALOG keyword
 COPY command 92
 DUMP command 149

UNDEFINEDSORG subkeyword
 COPY command 81
 RESTORE command 202

UNITADDR parameter (Stand-Alone TAPECNTL command) 253

UNLOAD parameter (Stand-Alone TAPECNTL command) 253

unloading a tape with the TAPECNTL command 252

User
 area list 305
 Area Pointer 314
 Exit Allowance 311
 Return Code 312

User Interaction Module (UIM) 306
 example 359
 list 304
 sample output 359

utilization, percent
 COPY command 81
 RESTORE command 202

UTILMSG keyword 8

V

VALIDATE keyword (DUMP command) 149

variable length (VL) parameter 305

verification
 bypass exit – eioption 22 323
 data set – eioption 21 323
 exit, OPEN/EOV – eioption 16 321

VERIFY parameter (Stand-Alone RESTORE command) 249

versions of DFSMSdss,
 compatibility 285

virtual concurrent copy 55

VL parameter 305

VM, running Stand-Alone Services under 238

VOLCOUNT keyword
 COPY command 92
 RESTORE command 214

volume
 DUMP with concurrent copy 152
 notification – eioption 20 322
 serialization 287

VSAM (Virtual Storage Access Method)
 data set
 filtering 11
 restore actions on 217

VTOC
 keyword (PRINT command) 143, 163

W

WAIT keyword
 COMPRESS command 36
 COPY command 94
 DEFrag command 117
 DUMP command 150
 PRINT command 164
 RELEASE command 176
 RESTORE command 216

WAIT option 289, 295

wait-state codes for Stand-Alone program 245

WORKUNIT keyword 8

WORKVOL keyword 8

WRITECHECK keyword
 COPY command 95
 DEFrag command 118
 RESTORE command 216

writing
 disk track – eioption 08 319
 logical tape record – eioption 05 318
 physical tape record – eioption 06 318
 SYSPRINT record – eioption 10 319
 to the operator 229
 WTO message – eioption 11 320
 WTOR message – eioption 12 320

WTO
 command 229
 message writing – eioption 11 320

WTOR
 command 229
 message writing – eioption 12 320, 321

X

XABUFF keyword 8

Z

ZBUFF64R keyword 8

zFS data sets
 coexistence considerations 285
 DUMP command considerations 145
 logical data set COPY 43
 logical dump and serialization 291
 physical dump and serialization 291
 read/write serialization scheme 293

RELEASE command
 consideration 167

Readers' Comments — We'd Like to Hear from You

**z/OS
DFSMSSdss Storage Administration
Reference**

Publication No. SC35-0424-06

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	<input type="checkbox"/>				

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	<input type="checkbox"/>				
Complete	<input type="checkbox"/>				
Easy to find	<input type="checkbox"/>				
Easy to understand	<input type="checkbox"/>				
Well organized	<input type="checkbox"/>				
Applicable to your tasks	<input type="checkbox"/>				

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? Yes No

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

Name

Address

Company or Organization

Phone No.

Readers' Comments — We'd Like to Hear from You
SC35-0424-06



Cut or Fold
Along Line

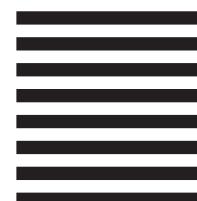
Fold and Tape

Please do not staple

Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department 55JA, Mail Station P384
2455 South Road
Poughkeepsie, NY
United States of America 12601-5400



Fold and Tape

Please do not staple

Fold and Tape

SC35-0424-06

Cut or Fold
Along Line

IBM[®]

Program Number: 5694-A01

Printed in USA

SC35-0424-06

