DFSMS V1.10 and EAV Technical Guide

Learn the function and features of z/OS V1R9 and V1R10

Plan for EAV volumes and migrate to them

Includes worked implementation examples

Mary Lovelace
Edson Buturi
Anthony Fletcher
Gert Laumann
Eduard Pavlik
Norbert Schlumberger

ibm.com/redbooks
Note: Before using this information and the product it supports, read the information in “Notices” on page xi.

First Edition (September 2009)
This edition applies to Version 1, Release 9 and Version 1, Release 10 of z/OS (product number 5694-A01).
Contents

Notices ......................................................................................................................... xi
Trademarks ................................................................................................................... xii

Preface ........................................................................................................................... xiii
The team who wrote this book ....................................................................................... xiii
Become a published author ......................................................................................... xv
Comments welcome .................................................................................................... xvi

Part 1. z/OS V1.9 DFSMS function summary ................................................................. 1

Chapter 1. DFSMSdfp V1.9 enhancements ................................................................. 3
  1.1 Basic Access Method performance enhancements ................................................. 4
    1.1.1 Long-term page fixing for BSAM data buffers .............................................. 4
    1.1.2 BSAM and QSAM support for MULTACC ..................................................... 5
    1.1.3 QSAM support for MULTSDN parameter ................................................... 6
  1.2 VSAM system-managed buffering enhancements ................................................. 7
    1.2.1 SMB overview .............................................................................................. 7
    1.2.2 Installation considerations ............................................................................. 8
  1.3 System page data sets .......................................................................................... 9
    1.3.1 Page data set usage summary: Pre-DFSMS V1.9 changes .............................. 9
    1.3.2 IDCAMS LISTCAT ....................................................................................... 10
    1.3.3 TSO RECEIVE error ..................................................................................... 16
    1.3.4 Large format data set support: Additional TSO support .............................. 16
    1.3.5 Large format data set support: Additional z/OS FTP support ..................... 17

Chapter 2. DFSMSdss Fast Replication use of new SFI support .................................. 19
  2.1 DFSMSdss Fast Replication and SFI: Overview ................................................. 20
  2.2 Storage class attribute influence ......................................................................... 20
  2.3 Storage Facility Image: Level of support ............................................................. 20
  2.4 DFSMS volume allocation selection with SFI attribute ....................................... 21

Chapter 3. DFSMShsm V1.9 enhancements ................................................................ 23
  3.1 DFSMShsm enhancements ................................................................................... 24
    3.1.1 New cross-memory service invoking DSS .................................................... 24
    3.1.2 Functional statistics record (FSR) improvements ......................................... 26
    3.1.3 Return priority exit ARCRPEXT changes ..................................................... 27
  3.2 System Logger handling of DFSMShsm recalls ..................................................... 28

Chapter 4. ISPF enhancements in z/OS V1.9 ............................................................... 33
  4.1 ISPF profile sharing ............................................................................................ 34
  4.2 Browsing and viewing z/OS UNIX files .............................................................. 34
  4.3 System symbolics in data set name and volume .................................................. 35
  4.4 ISPF Data Set utility list totalling ....................................................................... 35

Chapter 5. PDS and PDSE enhancements in z/OS V1.9 ............................................. 37
  5.1 PDS and PDSE changes ....................................................................................... 38
  5.2 Migration: PDSE ............................................................................................... 38
  5.3 Migration: PDS ................................................................................................. 38
  5.4 PDS/PDSE Volcount restriction ........................................................................ 38
## Chapter 8. DFSMSrmm V1.9 enhancements

8.1 Task management support ........................................... 50
  8.1.1 New support in z/OS V1.9 ..................................... 51
  8.1.2 Long running tasks ............................................ 51
  8.1.3 Operator commands ........................................... 51
  8.1.4 New operator commands ..................................... 53
  8.1.5 Restarting the DFSMSrmm subsystem ......................... 55
  8.1.6 Stopping the DFSMSrmm subsystem .......................... 55
  8.1.7 QUIESCE the DFSMSrmm subsystem ......................... 57

8.2 Multitasking of utilities ........................................ 59
  8.2.1 EDGHSKP EXPROC ........................................... 59
  8.2.2 EDSUTIL ...................................................... 62
  8.2.3 Using EDGSPLCS ............................................ 68

8.3 Control data set (CDS) serialization .......................... 78
  8.3.1 CDS serialization change .................................... 78
  8.3.2 CDSID PARMLIB option ..................................... 79
  8.3.3 Creating a CDSID to a new RMM system .................... 79
  8.3.4 Providing a CDSID to an existing RMM system ............ 80
  8.3.5 DFSMSrmm messages ......................................... 81
  8.3.6 Global resource serialization (GRS) ......................... 83
  8.3.7 Migration and coexistence considerations ................. 84

8.4 JCL data set names ............................................ 84
  8.4.1 Quoted data set name rules .................................. 84
  8.4.2 Un-quoted data set name rules .............................. 85
  8.4.3 Implementation ............................................... 85
  8.4.4 RMM subcommands ........................................... 86
  8.4.5 Creating a CDS record ....................................... 87
  8.4.6 Data set names in RMM subcommands ....................... 88
  8.4.7 Examples of RMM subcommands ............................... 88
  8.4.8 Changes in vital record specification (VRSEL) processing . 92
  8.4.9 Security considerations .................................... 94

8.5 Shared PARMLIB support ..................................... 95
  8.5.1 Visual overview .............................................. 95
  8.5.2 OPTION MEMBER(xx) ....................................... 95
  8.5.3 LISTCONTROL OPTION ................................... 96
  8.5.4 Restarting DFSMSrmm ....................................... 97
  8.5.5 DFSMSrmm installation wizard ............................. 97

8.6 TSO subcommands ............................................... 98
  8.6.1 Changes in subcommand parsing ............................ 98
  8.6.2 NOSECLEVEL parameter ................................... 99
  8.6.3 CLIST enhancements ......................................... 99
  8.6.4 Sample JCL for SEARCH subcommand ...................... 100
  8.6.5 User defined format of CLIST data set .................... 101
8.6.6 Search command enhancements ........................................ 103
8.6.7 STORAGEGROUP parameter ............................................. 114
8.7 REPORT17 of EDGRRPTE REXX exec ....................................... 114
8.8 3592 Model E05 software support .......................................... 115
  8.8.1 Change summary ..................................................... 116
  8.8.2 Migration and coexistence considerations ................................. 116
  8.8.3 Label anomaly processing ............................................ 117
8.9 Common Information Model (CIM) provider ................................. 117
  8.9.1 RMM CIM classes .................................................. 117
  8.9.2 CIM association classes ........................................... 118
  8.9.3 Supported CIM operations ........................................... 119
  8.9.4 Search operands .................................................... 119
  8.9.5 Software dependencies ............................................. 120
  8.9.6 CIM installation ................................................... 121
  8.9.7 Migration and coexistence considerations ................................. 122

Chapter 9. NFS enhancements in DFSMS V1.9 .......................... 123
  9.1 Network File Systems (NFS) enhancements ............................ 124
  9.2 24-bit addressing relief ................................................ 124
  9.3 Multi TCP/IP stack support ............................................ 124
  9.4 Usage and invocation .................................................. 124
  9.5 Addds operator command .............................................. 125
  9.6 RACF data labeling .................................................... 126
  9.7 NFS v4 client support ................................................ 127
  9.8 Client attribute syntax .............................................. 128
  9.9 Server Ctrace upgrade ................................................ 128
  9.10 Terminal ID based restricted MVSLOGIN ................................ 130

Part 2. z/OS V1.10 DFSMS enhancements .................................. 131

Chapter 10. DFSMSdfp V1.10 enhancements ............................... 133
  10.1 DFSMS data class space override ..................................... 134
  10.2 System determined block size set by data class ......................... 135
  10.3 New read-only variables in ACS routines ............................... 136
  10.4 Message enhancements ................................................ 137

Chapter 11. PDSE enhancements in z/OS V1.0 .......................... 139
  11.1 BMF DISPLAY HSPSTATS command ................................... 140
     11.1.1 General DISPLAY HSPSTATS commands .............................. 140
     11.1.2 Tailored display HSPSTATS commands .............................. 142
  11.2 BMF Display VSTOR command .......................................... 147
  11.3 Significance of the HSPSTATS ......................................... 148
  11.4 Migration and coexistence considerations ............................... 148

Chapter 12. DFSMSShsm V1.10 enhancements ............................. 149
  12.1 BACKDS enhancements: NEWNAME and ARCHBACK .......................... 150
     12.1.1 Security considerations .......................................... 153
     12.1.2 Backup version cleanup .......................................... 154
  12.2 CDS backup improvements ............................................. 154
  12.3 DFSMSShsm usage of virtual concurrent copy ........................... 155
  12.4 EAV: Extended Address Volume support ................................ 156
  12.5 RAS: TAPE COPY enhancement ......................................... 156
Chapter 13. DFSMSdss V1.10 enhancements ................................. 157
13.1 DUMP using NEWNAMEUNCONDITIONAL. .......................... 158
   13.1.1 Keyword syntax ............................................ 159
   13.1.2 Restrictions ................................................ 159
13.2 DEFRAg performance enhancement ................................... 160
13.3 CONSOLIDATE command .............................................. 160
   13.3.1 RACF protection ............................................ 161
   13.3.2 Coexistence APAR A22900 .................................. 162

Chapter 14. Virtual Concurrent Copy in DFSMS V1.10 ......................... 163
14.1 Virtual Concurrent Copy overview ................................... 164
14.2 Setting up the Virtual Concurrent Copy environment ................. 164
   14.2.1 ANTMAIN PARMLIB support ................................ 164
   14.2.2 XRC PARMLIB support ...................................... 165
   14.2.3 Working space data sets ................................... 165
   14.2.4 WSDS considerations ........................................ 167
   14.2.5 Running VCC in production ................................. 168
   14.2.6 DSS data movement method preference ..................... 169
   14.2.7 DFSMShsm backup processing ............................... 170
   14.2.8 Displaying WSDS usage ..................................... 171

Chapter 15. RLS Secondary Lock Structure support .......................... 173
15.1 VSAM RLS secondary lock structure overview ........................ 174
   15.1.1 Basic record level sharing setup .......................... 174
   15.1.2 SMSVSAM address space ..................................... 175
   15.1.3 CF structures .............................................. 175
15.2 Preparing for VSAM RLS Secondary Lock Structures ................ 176
15.3 Definition of RLS Secondary Lock Structures ........................ 176
15.4 Migrating to multiple lock structures ................................ 177
   15.4.1 Enabling multiple lock structure support .................. 177
   15.4.2 Displaying secondary lock structures ..................... 178
   15.4.3 Monitoring multiple lock structures using SMF ............ 180
   15.4.4 SMF Type 42 record ........................................ 180
15.5 Operating secondary lock structures ................................ 181
15.6 Fallback from using Multiple Lock Structures ...................... 182

Chapter 16. DFSMSdfp OAM V1.10 enhancements ............................ 183
16.1 DFSMSdfp OAM enhancement summary ................................ 184
   16.1.1 List of enhancements ....................................... 184
   16.1.2 Migration considerations .................................. 185
16.2 OAM enhancement implementation .................................. 185
   16.2.1 IEFSNxx PARMLIB changes ................................. 185
   16.2.2 Changes to CBROAMxx PARMLIB ............................ 187
   16.2.3 Changes to F OAM commands ............................... 188

Chapter 17. DFSMSrmm V1.10 enhancements ................................ 195
17.1 Enabling use of DFSMSdss COPY Services .......................... 196
   17.1.1 Backup of DFSMSrmm control data set using DSS COPY. .. 196
   17.1.2 Restore the DFSMSrmm control data set .................... 200
17.2 SMF record changes ................................................ 202
17.3 Audit controls for release processing .............................. 204
   17.3.1 EXPDRTDROP PARMLIB option ............................ 205
   17.3.2 VRSRMDROP PARMLIB option ............................... 207
   17.3.3 VRSRETAIN PARMLIB option ............................... 208
Chapter 19. Planning to use EAV ................................................. 327
  19.1 Software requirements and compatibility .............................. 328
  19.2 IBM System Storage DS8000 support .................................. 328
    19.2.1 Dynamic Volume Expansion .................................. 328
    19.2.2 Copy services considerations ................................. 329
  19.3 Supported data set types .............................................. 329
  19.4 Application migration .................................................. 329
    19.4.1 z/OS tracking facility ....................................... 330
    19.4.2 TRKADDR macro ............................................. 333
  19.5 VTOC changes .......................................................... 333
    19.5.1 VTOC size considerations .................................. 333
    19.5.2 LSPACE macro changes .................................... 334
    19.5.3 VVDS size .................................................... 334
    19.5.4 Extent descriptors .......................................... 335
    19.5.5 VTOC/VVDS EAV creation scenario .......................... 335
  19.6 Allocation considerations ............................................ 335
    19.6.1 Volume size considerations ................................ 337
    19.6.2 SMS storage group free space considerations ............... 338
    19.6.3 SMS volume selection ...................................... 338
  19.7 Analyzing your environment before implementing EAV .............. 340
  19.8 Defragmentation of EAVs .............................................. 341
    19.8.1 DEFRAG Version 1 .......................................... 341
    19.8.2 DEFRAG Version 2 .......................................... 341
    19.8.3 RACF considerations ....................................... 342
  19.9 DFSMSshm space management ......................................... 343
  19.10 Copy Services considerations ...................................... 345
  19.11 SMF changes .......................................................... 345
    19.11.1 SMF Type 14 and 15 ....................................... 345
    19.11.2 SMF Type 19 .............................................. 346
    19.11.3 SMF Type 60, 61, 64, 65, 66 .............................. 346
    19.11.4 SMF Type 74, subtype 1 .................................. 346
  19.12 DFSMSdss considerations ............................................ 346
  19.13 DB2 considerations .................................................. 346
  19.14 Out of space abends .................................................. 347
  19.15 Migration positioning ............................................... 347
    19.15.1 Expanding existing volumes with DVE ...................... 347
    19.15.2 Creating new EAVs ....................................... 348
    19.15.3 Planning for testing and implementing EAV .............. 348
    19.15.4 Fallback from EAV processing ............................ 349
    19.15.5 User education ............................................ 349
    19.15.6 Miscellaneous changes ................................... 349
    19.15.7 DEVSERV display .......................................... 350

Chapter 20. DS8000 EAV implementation .................................... 351
  20.1 Implementing EAV for the DS8000 .................................. 352
    20.1.1 Base installation ........................................... 352
    20.1.2 Alias considerations ....................................... 353
    20.1.3 HyperPAV and PAV features ................................ 353
  20.2 Creating EAVs ........................................................ 354
    20.2.1 Creating a new EAV ......................................... 354
    20.2.2 ICKDSF initialization ....................................... 360
    20.2.3 Expanding existing volumes ................................ 360
    20.2.4 Reformating VTOC .......................................... 365
SMF85TA Program execution results ........................................... 475
SMF Record 85 Subtype 32-35 Records for OAM OSMC usage .......... 483
SMF Record 85 Subtype 40 Records for OAM OSMC Command Recycle .... 491

Appendix C. Code samples DFSMS V1.10 ........................................ 495
C.1 SMF Record type 85 subtype 1-10 data display program .......... 496
C.2 VTOC management: Enhancement for EAV ............................ 496
C.3 VTOC management macros extended for EAV - LSPACE .......... 497
  C.3.1 EAV Migration Assistance Tracker considerations: LSPACE. 498
  C.3.2 LSPACE macro use examples ..................................... 498
C.4 VTOC management macros extended for EAV: OBTAIN .......... 510
  C.4.1 EAV Migration Assistance Tracker considerations: OBTAIN  510
  C.4.2 OBTAIN macro use examples ..................................... 511
C.5 VTOC management macros extended for EAV: REALLOC .......... 530
C.6 VTOC management macros extended for EAV: CVAFDIR ............ 530
  C.6.1 EAV Migration Assistance Tracker considerations: CVAFDIR 535
C.7 VTOC management macros extended for EAV: CVAFDSM .......... 560
  C.7.1 EAV Migration Assistance Tracker considerations: CVAFDSM 562
C.8 VTOC management macros extended for EAV - CVAFFILT ......... 572
  C.8.1 EAV Migration Assistance Tracker considerations: CVAFFILT 580
C.9 VTOC management macros extended for EAV - CVAFSEQ ........... 599
  C.9.1 EAV Migration Assistance Tracker considerations: CVAFSEQ 602
C.10 DCB OPEN of VTOC extension for EAV ................................. 619
  C.10.1 EAV Migration Assistance Tracker considerations: DCB OPEN of VTOC 619
  C.10.2 OPEN/DCB/READ macro use to read the VTOC. ................. 619
C.11 DCB OPEN of EAS eligible data set extension for EAV .......... 630
  C.11.1 EAV Migration Assistance Tracker Consideration: DCB OPEN of an EAS Eligible data set. 630
C.12 Supporting macros for sample programs .............................. 630
C.13 Supporting subroutine for sample programs ......................... 636
  C.13.1 Subroutine CVSTCHK ........................................... 636
C.14 Migration Assistance Tracker .......................................... 641
  DFSMS instances tracked by the EAV migration assistance tracker  642
  Exclusion list .......................................................... 658

Related publications ............................................................. 661
IBM Redbooks publications ...................................................... 661
Other publications .............................................................. 661
Online resources ............................................................... 661
How to get Redbooks publications ........................................... 661
Help from IBM ................................................................. 662

Index ................................................................. 663
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.

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.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.
Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (© or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

CICS®

DB2®

DS6000™

DS8000®

ECKD™

eServer™

FlashCopy®

GDPS®

HyperSwap®

IBM®

IMS™

OS/390®

Parallel Sysplex®

RACF®

Redbooks®

Redbooks (logo)®

Service Director™

System Storage™

TDMF®

Tivoli®

TotalStorage®

WebSphere®

z/Architecture®

z/OS®

zSeries®

The following terms are trademarks of other companies:

Disk Magic, and the IntelliMagic logo are trademarks of IntelliMagic BV in the United States, other countries, or both.

ACS, and the Shadowman logo are trademarks or registered trademarks of Red Hat, Inc. in the U.S. and other countries.

Java, JDK, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

Each release of DFSMS builds upon the previous version to provide enhanced storage management, data access, device support, program management, and distributed data access for the z/OS® platform in a system-managed storage environment.

This IBM® Redbooks® publication provides a summary of the functions and enhancements in z/OS V1R9 DFSMS. It then provides an in-depth technical description of the functions and enhancements in z/OS V1R10 DFSMS. It provides you with the information that you need to understand and evaluate the content of this DFSMS release, along with practical implementation hints and tips. Also included are enhancements that were made available through an enabling PTF that has been integrated into z/OS DFSMS V1R10.

z/OS V1R10 provides Extended Address Volumes (EAV), a capability that enables support for over 54 GB of addressable storage, removing a limitation in place today. Initially EAV is planned to support up to 262,668 cylinders (up to 223 GB of addressable storage) per volume, allowing you to simplify storage management by providing the ability to manage fewer, larger volumes as opposed to many small volumes. This book provides detailed information about planning, implementing, and migrating to EAVs.

This book was written for storage professionals and system programmers who have experience with the components of DFSMS. It provides sufficient information so that you can start prioritizing the implementation of new functions and evaluating their applicability in your DFSMS environment.

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center, as shown in the following photograph.
Mary Lovelace is a Consulting IT Specialist at the International Technical Support Organization. She has more than 20 years of experience with IBM in large systems, storage and storage networking product education, system engineering and consultancy, and systems support. She has written many IBM Redbooks publications about z/OS storage products and TotalStorage® Productivity Center.

Edson Buturi has been an IT Storage Specialist in Hortolândia, São Paulo, Brazil for two years. He is working for International Commercial Accounts. Before that, he worked 25 years in the Bank of Brazil, in the mainframe environment (operation, production, storage support, and auditor of IT). His areas of expertise include DFSMSdfp, DFSMSHsm, and tapes. He has a Lawyer degree from the Faculdade Curitiba, of Curitiba, Paraná, Brazil.

Anthony Fletcher is an IT Specialist working with the Global Services Delivery (GSD) z/OS software platform of IBM Global Services Ondemand Infrastructure Services, based in New Zealand, but working for New Zealand and Australia. He has over 35 years of experience in z/OS, OS/390®, and their predecessors and related components, both as a customer of IBM and with IBM Global Services. He is a team leader for the mainframe operations of four diverse clients in the banking, airline, investment, and telecommunications industries. He holds a degree in Electrical Engineering from SALFORD University, Lancashire, UK. His main areas of expertise include DFSMS, DFSMSrmm, and DFSMSHsm, and has a working knowledge of RACF®. He has been involved with writing several IBM Redbooks publications. He also has experience in installing non-IBM products for the GSD platform.

Gert Laumann is an IT Specialist in Integrated Technology Delivery, Server Systems Operations / Storage Management in Denmark. He has 23 years of experience in z/OS, working with storage management since 1989. He is the team leader for the Danish mainframe storage team. The experience is mostly founded on IBM products (DFSMSdfp, DFSMSdss, DFSMSHsm, and DFSMSrmm, but has at the same also worked with OEM software and hardware products. His focus has mainly been automation and standardization across mainframe customer platforms, currently working on helping customers share IBM DASD and tape hardware.
Eduard Pavlik is a Senior System Programmer working for DATEV eG in Nürnberg, Germany. He has over 20 years of mainframe experience, and for the past 17 years has focused on Storage Management. His main areas of expertise include DFSMS, DFSMShsm, DFSMSrmm, DFSMSoam, TSM, ICF Catalogs and ATLs.

Norbert Schlumberger is an IT Architect with IBM Germany. He has 31 years of experience in storage software and storage management for IBM and customer systems, including 19 years of experience in DFSMSrmm. He also has experience in DFSMShsm and possesses a good knowledge of RACF. Norbert's areas of expertise include performing conversions from vendor tape management products to DFSMSrmm, new DFSMSrmm implementations, and marketing support for DFSMSrmm, including IBM 3494 and IBM 3495 ATLs, VTSs, and vendor robotics. He wrote a tool called “Tape Copy Tool: Enhancing DFSMSrmm” to copy data sets residing on tape from one media to another media, including the update of the ICF user catalog and DFSMSrmm after all data is successfully copied. He has worked at IBM for 34 years.

Thanks to the following people for their contributions to this project:

Rich Conway
Robert Haimowitz
Paul Rogers
International Technical Support Organization

Arthur Bariska
Stephen Branch
James Cammarata
Jerry Codde
Lisa Ingall
Mike Koester
Lyle Merithew
Glenn Wilcock
Mike Wood

Book reviewers and technical support

Special thanks to Cecilia Carranza Lewis and Wayne Rhoten for their time and support in the education of the team and their extensive review of the materials.

The EAV part of this book is dedicated in memory of Ken Kapulka and Marc Duquette, who were members of the EAV core design team.

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at: ibm.com/redbooks/residencies.html
Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online Contact us review Redbooks publications form found at:
  ibm.com/redbooks
- Send your comments in an e-mail to:
  redbooks@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400
z/OS V1.9 DFSMS function summary

This part of the book provides an overview of the function provided in z/OS V1.9 DFSMS.
Chapter 1. DFSMSdfp V1.9 enhancements

In this chapter we discuss new and changed functions in DFSMSdfp.

We cover the following topics:

- Basic Access Method performance enhancements
- VSAM system-managed buffering enhancements
- System page data sets
- Page data set usage summary: Pre-DFSMS V1.9 changes
- IDCAMS LISTCAT
- TSO RECEIVE error
- Large format data set support: Additional TSO support
- Large format data set support: Additional z/OS FTP support
1.1 Basic Access Method performance enhancements

In z/OS V1.9 DFSMS provides the following enhancements to BSAM and QSAM, as follows:

- Long-term page fixing for BSAM data buffers with the FIXED=USER parameter
- BSAM and QSAM support for the MULTACC parameter
- QSAM support for the MULTSDN parameter

The DCB extension (DCBE) macro must be coded for these enhancements. The format of the DCBE macro is shown in Figure 1-1.

```
[label] DCBE [,BLKSIZE=n]
    [,BLOCKTOKENSIZE={LARGE|SMALL}]
    [,CAPACITYMODE=XCAP]
    [,EODAD=relexp]
    [,FIXED=USER]
    [,GETSIZE={YES|NO}]
    [,NOVER={YES|NO}]
    [,PASTEOD={YES|NO}]
    [,RMODE31={BUFF|NONE}]
    [,SYNAD=relexp]
    [,SYNC={SYSTEM|NONE}]
```

*Figure 1-1  DCBE macro format*

Where:

**FIXED=USER**  With this DCBE option, the caller ensures that the data areas will remain fixed from the time the READ and WRITE macro instruction is issued through the completion of the CHECK or WAIT macro instructions. Failure to keep them fixed can result in a system integrity exposure because the channel program is using the real addresses associated with the data areas. This keyword is new with z/OS V1.9.

1.1.1 Long-term page fixing for BSAM data buffers

To improve performance, in z/OS V1.9 BSAM allows certain calling programs to specify that all their BSAM data buffers have been page fixed. This specification frees BSAM from the CPU-time intensive work of fixing and freeing the data buffers itself. The only restrictions are:

- The calling program must be APF authorized, or be in system key or supervisor state.
- The format of the data set must be either basic format, large format, PDS, or extended format.

*Note: Compressed format data sets are not supported.*

The DCBE macro option “FIXED=USER” must be coded to specify that the calling program has done its own page fixing and indicates that the user has page fixed all BSAM data buffers.
A bit in the DCBE control block (DCBEBENEFIX) is set on to indicate that a program can benefit from specifying FIXED=USER on the DCBE macro. In addition, a bit in the data extent block (DEB2XUPF) is set on to indicate that FIXED=USER is in effect.

The caller can ensure that the data areas are fixed by doing either of these actions:

- Issuing the PGSER FIX macro.
- Using the GETMAIN or STORAGE macro for a page fixed subpool.

To avoid duplicate page fixing, the user program can test whether it is running on the appropriate release. To do that, the user program can test whether DFARELS is equal to or greater than X’03010900’.

### 1.1.2 BSAM and QSAM support for MULTACC

In z/OS V1.9, the MULTACC parameter of the DCBE macro is expanded. This is done to optimize performance for tape data sets with BSAM, and to support QSAM with optimized performance for both tape and DASD data sets. The calculations that are used to optimize the performance for BSAM with DASD data sets are also enhanced.

When dealing with a tape data set, OPEN supports MULTACC for BSAM and QSAM.

#### BSAM support

For BSAM in V1.9, if you code a nonzero MULTACC value, OPEN calculates a default number of READ or WRITE requests that you are suggesting the system queue more efficiently. OPEN calculates the number of BLKSIZE-length blocks that can fit within 64 KB, then multiplies that value by the MULTACC value. If the block size exceeds 32 KB, then OPEN uses the MULTACC value without modification (this can happen only if you are using LBI, the large block interface). The system then tries to defer starting I/O requests until you have issued this number of READ or WRITE requests for the DCB. BSAM will never queue (defer) more READ or WRITE requests than the NCP value set in OPEN.

For BSAM, it will work as documented for DASD:

- If you code a nonzero MULTACC value, OPEN will calculate a default number of read or write requests that you are suggesting the system queue more efficiently.
- The system will try to defer starting I/O requests until you have issued this many read or write requests for the DCB.

**Note:** BSAM will never queue or defer more read or write requests than the number of channel programs (NCP) value set in OPEN.

#### QSAM support

For QSAM in z/OS V1.9, if you code a nonzero MULTACC value, OPEN calculates a default number of buffers that you are suggesting the system queue more efficiently. OPEN calculates the number of BLKSIZE-length blocks that can fit within 64 KB, then multiplies that value by the MULTACC value. If the block size exceeds 32 KB, then OPEN uses the MULTACC value without modification (this can happen only if you are using LBI, the large block interface). The system then tries to defer starting I/O requests until that number of buffers has been accumulated for the DCB. QSAM will never queue (defer) more buffers than the BUFNO value that is in effect.

If you code a nonzero MULTACC value, OPEN will calculate a default number of buffers that you are suggesting the system queue more efficiently.
The system will try to defer starting I/O requests until that many buffers have been accumulated for the DCB.

**Note:** QSAM will never queue (defer) more buffers than the BUFNO value that is in effect. IBM recommends setting MULTACC to one half of the MULTSDN value. If you code a MULTACC value that is too large for the system to use, the system ignores the excess amount. However, the absolute upper limit for MULTACC is 255.

### 1.1.3 QSAM support for MULTSDN parameter

In z/OS V1.9, you can use the MULTSDN parameter of the DCBE macro with QSAM. In previous releases, QSAM ignored the MULTSDN parameter. This new support for MULTSDN allows the system to calculate a more efficient default value for DCB's BUFNO parameter, and reduces the situations where you need to specify a BUFNO value.

The user can use MULTSDN to give a hint to OPEN so it can calculate a better default value for QSAM BUFNO instead of 1, 2 or 5. The user will not have to be dependent on device information such as blocks per track or number of stripes.

QSAM accepts a MULTSDN value for the following data sets:

- Tape data sets
- DASD data sets of the following types:
  - Basic format
  - Large format
  - Extended format (non-compressed)
  - PDS

For these supported data set types, the system uses MULTSDN to calculate a more efficient value for BUFNO when the following conditions are true:

- The MULTSDN value is not zero.
- DCBBUFNO has a value of zero after completion of the DCB OPEN exit routine.
- The data set block size is available.

When MULTSDN is specified, note that the default number of buffers might be less than what would have been derived without MULTSDN, as shown in Figure 1-2.

<table>
<thead>
<tr>
<th>Data Set Type</th>
<th>DCBBUFNO default without MULTSDN</th>
<th>DCBBUFNO default with MULTSDN</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDSE Member</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Extended format data set in the compressed format</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>UNIX file</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Extended format data set (not in the compressed format)</td>
<td>2 * number of stripes * number of blocks per track</td>
<td>MULTSDN * number of stripes * number of blocks per track</td>
</tr>
<tr>
<td>Block size equal to or greater than 32 KB (tape)</td>
<td>2</td>
<td>MULTSDN</td>
</tr>
<tr>
<td>Block size less than 32 KB (tape)</td>
<td>5</td>
<td>MULTSDN * number of blocks in 64 KB</td>
</tr>
<tr>
<td>IBM 2540 card reader or card punch</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>PS, PDS</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Others (including Dummy)</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

*Figure 1-2  Default buffer numbers for QSAM (with and without MULTSDN)*
1.2 VSAM system-managed buffering enhancements

The JCL AMP parameter SMBVSP keyword lets you limit the amount of virtual buffer space to acquire for direct optimized processing when opening a data set. Before z/OS V1.9, changing that value required editing the JCL statement, which was not practical when running a batch job. In z/OS V1.9, VSAM provides a simpler, more efficient way of modifying the SMBVSP value, by specifying it for a data class using ISMF. The system-managed buffering (SMB) field on the ISMF DATA CLASS DEFINE/ALTER panel lets you specify the value in kilobytes or megabytes. SMB then uses the specified value for any data set defined to that data class. With this method, the effect of modifying the SMBVSP keyword is no longer limited to one single job step, and no longer requires editing individual JCL statements.

In addition, a new JCL AMP keyword, MSG=SMBBIAS, lets you request a message that displays the record access bias that is specified on the ACCBIAS keyword or chosen by SMB in the absence of a user selection. The IEC161I message is issued for each data set that is opened. The new keyword is optional and default is to not issue a message. You should avoid the keyword when a large number of data sets are opened in quick succession.

1.2.1 SMB overview

System-managed buffering (SMB), a feature of DFSMSdfp, supports batch application processing. SMB uses formulas to calculate the storage and buffer numbers needed for a specific access type. Each algorithm is called an access bias. SMB takes the following actions:

- It changes the defaults for processing VSAM data sets. This enables the system to take better advantage of current and future hardware technology.
- It initiates a buffering technique to improve application performance. The technique is one that the application program does not specify. You can choose or specify any of the four processing techniques that SMB implements:
  
  **Direct Optimized (DO)** The DO processing technique optimizes for totally random record access. This is appropriate for applications that access records in a data set in totally random order. This technique overrides the user specification for nonshared resources (NSR) buffering with a local shared resources (LSR) implementation of buffering.

  **Sequential Optimized (SO)** The SO technique optimizes processing for record access that is in sequential order. This is appropriate for backup and for applications that read the entire data set or a large percentage of the records in sequential order.

  **Direct Weighted (DW)** The majority is direct processing, some is sequential. DW processing provides the minimum read-ahead buffers for sequential retrieval and the maximum index buffers for direct requests.

  **Sequential Weighted (SW)** The majority is sequential processing, some is direct. This technique uses read-ahead buffers for sequential requests and provides additional index buffers for direct requests. The read-ahead will not be as large as the amount of data transferred with SO.
1.2.2 Installation considerations

Using the new SMB enhancement, you can:

- Set the storage limit used by SMB DO from the ISMF DFSMS data class panel as shown in Figure 1-3.

![Figure 1-3 SMB value set in the data class]

An ISMF list of the data class, showing the resulting SMBVSP value is shown in Figure 1-4.

![Figure 1-4 SMBVSP value in data class list]

- Code MSG=SMBBIAS in your JCL to request a VSAM open message indicating what SMB access bias actually is used for a particular component being opened, as shown in Figure 1-5.
The IEC161I message format is shown in Figure 1-6.

```
IEC161I 001 (Actual Access Bias being used)-255
For Non-DO, extra message indicating buffer numbers:
IEC161I 001 (AAAAAABB CCCCCDDD DDDDDDD)-255
  WHERE AAAAAA = HEX VALUE OF BASE BUFND
  BBBBBBBB = HEX VALUE OF BASE BUFNI
  CCCCCCCC = HEX VALUE OF PATH BUFND
  DDDDDDDD = HEX VALUE OF PATH BUFNI
```

Figure 1-6  IEC161I message format

The IEC161I message will only appear if requested for diagnostic, or analytical purposes. Avoid flooding the console by having the message issued for every open of a component.

Using the new SMB enhancement, you can provide a way to limit the storage SMB DO uses for a large amount of data sets at once, without changing the JCL for each job step.

## 1.3 System page data sets

There has been no change in z/OS V1.9 relating to the system page data set size, but the restriction on the usable part of a page data set remains with z/OS V1.9. For users migrating from z/OS V1.7 to V1.9, this restriction might not be recognized.

In z/OS V1.8 DFSMSdfp enhanced the IDCAMS DEFINE PAGESPACE to allow larger data sets to be defined

The Auxiliary Storage Manager component was not enhanced to handle these larger data sets. Any PAGE data set that is defined with more slots than the supported number can be used in z/OS V1.8 and z/OS V1.9, but only the supported maximum will be used.

The long standing recommendation that page data set usage not be allowed to go above 30% remains, and this is applied to the part of the data set that is being used. Therefore if larger data sets are allocated, the unusable percentage becomes progressively more than 70%.

### 1.3.1 Page data set usage summary: Pre-DFSMS V1.9 changes

z/OS V1.8 DFSMS provided support to allow a page space to have a maximum size equal to 16,777,215 slots, which is 16 GB. Support in z/OS has not been provided to allow a local page data set to be greater than 1,048,576 slots, which is 4 GB.

The z/OS restriction of 4GB of page space applies to all page data sets; PLPA, COMMON, and LOCAL.
1.3.2 IDCAMS LISTCAT

There are no new LISTCAT functions introduced in z/OS V1.9, but significant changes were made in z/OS V1.8 that will affect users migrating from z/OS V1.9 from V1.7. z/OS V1.8 introduced improvements in LISTCAT processing performance, resulting in changes in the SYSPRINT output format.

LISTCAT

IDCAMS LISTCAT processing in z/OS V1.8 was enhanced to provide better performance, especially for large catalogs. The performance improvement is automatic in z/OS V1.8 and no action needs to be taken in order to exploit the new function. The output from LISTCAT command has also changed, which might affect products that process the output. The system service used to achieve improved performance is Generic File Locate (GFL).

As a result of difficulties some installations had in accommodating the output format changes during conversion to z/OS V1.8, an alias called IDCNOGFL has been defined to IDCAMS, which, when used, results in output that is in the same format as on z/OS V1.7, but does not include the performance improvements.

The IDCNOGFL alias is made available for z/OS V1.8 and z/OS V1.9 through PTFs.

If an installation determines that they will need to use this new alias when converting to z/OS 1.8 or z/OS V1.9, the same alias is made available for z/OS V1.7. In the case of z/OS V1.7, the alias provides exactly the same function as IDCAMS. The alias is provided to allow JCL changes to be made while still running on z/OS V1.7 so that JCL changes do not have to be made during the tarnation to z/OS V1.8 or z/OS V1.9.

LISTCAT LEVEL examples

A new header line appears on each page of LISTCAT output. Additionally, there have been changes in the way LISTCAT LEVEL processing works for GDGs and ALIASes. Figure 1-7 shows an example of pre-z/OS V1.8 LISTCAT LEVEL output, specifying the ALIAS name in the LISTCAT LEVEL command.

```
LISTCAT LEVEL(SYSDOC)
GDG BASE ------ SYSDOC.CIMN.SYSLOGD
   IN-CAT --- COMCIC.ICFCAT
NONVSAM ------ SYSDOC.CIMN.SYSLOGD.G0832V00
   IN-CAT --- COMCIC.ICFCAT
   .
   .
   .
IDCAMS SYSTEM SERVICES
   TIME: 17:06:24
   THE NUMBER OF ENTRIES PROCESSED WAS:
   AIX -------------------0
   ALIAS -----------------0
   CLUSTER ---------------0
```

Figure 1-7 Pre z/OS V1.8 LISTCAT output
Figure 1-8 shows an example of z/OS V1.8 LISTCAT LEVEL output illustrating the header and ALIAS changes.

```
LISTCAT LEVEL(SYSDOC)  LISTING FROM CATALOG -- COMCIC.ICFCAT
GDG BASE -------- SYSDOC.CIMN.SYSLOGD
IN-CAT --- COMCIC.ICFCAT
NONVSAM -------- SYSDOC.CIMN.SYSLOGD.G0832V00
  IN-CAT --- COMCIC.ICFCAT
.
.
.
IDCAMS SYSTEM SERVICES                                           TIME: 17:04:40
LISTING FROM CATALOG -- SCI18A.DASDPLEX.CATALOG
ALIAS --------- SYSDOC
  IN-CAT --- SCI18A.DASDPLEX.CATALOG
IDCAMS SYSTEM SERVICES                                           TIME: 17:04:40
LISTING FROM SELECTED CATALOGS
THE NUMBER OF ENTRIES PROCESSED WAS:
  AIX -------------------0
  ALIAS -----------------1
  CLUSTER --------------0
```

Figure 1-8  z/OS V1.8 LISTCAT output

**PGM=IDCAMS and PGM=IDCNOGFL examples**

In the following examples, we illustrate the differences in the output when using PGM=IDCAMS and then using PGM=IDCNOGFL.

**Note:** These examples are not the complete output from the two modes of execution. The intent is to illustrate that there are changes, so you can recognize that changes might be required if post processing of the IDCAMS output is implemented.

In Figure 1-9 on page 12 and Figure 1-10 on page 13, we show output when using the standard PGM=IDCAMS invocation to issue the LISTCAT LVL(MHLRES1) command.

In these examples, the differences are highlighted where the difference is in the text. There are also numerous changes in the output carriage control content that is not highlighted unless part of other text changes.
In Figure 1-9, the changes to the output are:

- The addition of an extra page eject and title line between the echo of the command line and the output
- A different order on the GDG detailed information

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Page</th>
<th>Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>02/14/08</td>
<td>19:49:42</td>
<td>1</td>
<td>IDCAMS SYSTEM SERVICES</td>
<td>- LISTC LVL(MHLRES1) ALL</td>
</tr>
<tr>
<td>02/14/08</td>
<td>19:49:42</td>
<td>2</td>
<td>IDCAMS SYSTEM SERVICES</td>
<td>- LISTING FROM CATALOG -- UCAT.VSBOX01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ONONSVSAM -------- MHLRES1.ABARS.SELECT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>IN-CAT --- UCAT.VSBOX01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HISTORY</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DATASET-OWNER-----(NULL) CREATION----------2007.051</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RELEASE------------------2 LAST ALTER--------2007.075</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ATTRIBUTES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LIMIT------------------7 SCRATCH NOEMPTY</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ASSOCIATIONS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NONSVSAM--MHLRES1.BCAT.CATALOG.LISTING.G0001V00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NONSVSAM--MHLRES1.BCAT.CATALOG.LISTING.G0002V00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NONSVSAM--MHLRES1.BCAT.CATALOG.LISTING.G0003V00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NONSVSAM--MHLRES1.BCAT.CATALOG.LISTING.G0004V00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NONSVSAM--MHLRES1.BCAT.CATALOG.LISTING.G0005V00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NONSVSAM--MHLRES1.BCAT.CATALOG.LISTING.G0006V00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ONONSVSAM -------- MHLRES1.BCAT.CATALOG.LISTING.G0002V00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>IN-CAT --- UCAT.VSBOX01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HISTORY</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DATASET-OWNER-----(NULL) CREATION----------2007.051</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>EXPIRATION------0000.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ACCOUNT-INFO--</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>STATUS--------ACTIVE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SMSDATA</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>STORAGECLASS ---STANDARD MANAGEMENTCLASS---MCDB22</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DATACLASS --------(NULL) LBACKUP ---0000.000.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>VOLUMES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>VOLSER--------------MLD30C DEVTYPEx-3010200F'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FSEQN------------------0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ASSOCIATIONS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>IDCAMS SYSTEM SERVICES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>02/14/08 PAGE 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- LISTING FROM CATALOG -- UCAT.VSBOX01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 GDG------MHLRES1.BCAT.CATALOG.LISTING</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Attributes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ONONSVSAM -------- MHLRES1.BCAT.CATALOG.LISTING.G0001V00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>IN-CAT --- UCAT.VSBOX01</td>
</tr>
</tbody>
</table>

Figure 1-9  z/OS V1.9 LISTCAT output extract using PGM=IDCAMS (1 of 2)
Figure 1-10 shows the changes to the output, including:

- Addition of text concerning the ALIAS entry
- Addition of the ALIAS count in the summary
- An increase in the number of entries in the summary

```
OALIAS --------- MHLRES1
  IN-CAT --- MCAT.SANDBOX.Z19.SBOX00
  HISTORY
  RELEASE----------------2
  ASSOCIATIONS
  USERCAT--UCAT.VSBOX01
1IDCAMS   SYSTEM SERVICES
  TIME: 19:49:42 02/14/08 PAGE 75
- LISTING FROM SELECTED CATALOGS
  THE NUMBER OF ENTRIES PROCESSED WAS:
    AIX ------------------------0
    ALIAS ----------------------1
    CLUSTER ---------------------7
    DATA ------------------------6
    GDG -------------------------3
    INDEX -----------------------6
    NONVSAM --------------------215
    PAGESPACE ------------------0
    PATH ------------------------0
    SPACE -----------------------0
    USRCATALOG -----------------0
    TAPELIBRARY ----------------0
    TAPEVOLUME ------------------0
    TOTAL ----------------------238
```

Figure 1-10  z/OS V1.9 LISTCAT output extract using PGM=IDCAMS (2 of 2)
Figure 1-11 and Figure 1-12 show a sample from output using the alias PGM=IDCNOGFL invocation and the same LISTCAT command that produced the output in Figure 1-9 on page 12 and Figure 1-10 on page 13.

![Image of z/OS V1.9 LISTCAT output extract using PGM=IDCNOGFL (1 of 2)]
Migration and coexistence

Information APAR II14250 contains details about the differences in IDCAMS LISTCAT processing in z/OS V1.8 and z/OS V1.9 as well as the current recommendation on maintenance to be applied.

Maintenance

We recommend that you have the following PTFs installed prior to using the new support:

- UA38132 for z/OS V1.7 - for compatibility with the later releases
- UA38133 for z/OS V1.8 - provides alias IDCNOGFL as well as fixes
- UA38134 for z/OS V1.9 - provides alias IDCNOGFL as well as fixes
- Fix for APAR OA22903 for z/OS V1.8 and z/OS V1.9 when using PGM=IDCNOGFL
- Fix for APAR OA22625 for z/OS V1.8 and z/OS V1.9 when using PGM=IDCNOGFL
- Fix for APAR OA22627 for z/OS V1.8 and z/OS V1.9 when using PGM=IDCAMS
- Fix for APAR OA22632 for z/OS V1.8 and z/OS V1.9 when using PGM=IDCAMS
- Fix for APAR OA22542 for z/OS V1.8 and V1.9 when using PGM=IDCAMS (use PGM=IDCNOGFL as a bypass for this problem)
- Fix for APAR OA22078 for z/OS V1.6 and V1.7 to get alias IDCNOGFL

OA16912 and OA18720 represent the roll-up APARs for this new support. OA20169 is the APAR resulting from the PTF for OA18184 being marked PE. It is important because it fixes the PE introduced with OA18184 as well as the issue that OA18184 was intended to fix, where LISTCAT was not issuing error messages for offline volumes.
1.3.3 TSO RECEIVE error

A change was made in z/OS V1.9 to the TSO RECEIVE command as part of the implementation of support for DSNTYPE=LARGE data sets. This had unexpected consequences in some cases.

The RECEIVE command allocates a work area as part of its processing, specifically requesting the use of DSNTYPE=BASIC as part of the additional support for large format data sets. If an installation has SMS ACS™ specifications that allocate data sets to VIO (typically small data sets), the RECEIVE command can fail when it allocates the work area and it is assigned to VIO.

If this happens, the message does not clearly indicate that it is related to the use of VIO for the work data set.

A typical set of messages that might occur is shown in Figure 1-13. In this case, the data set being received is a PDS, so IEBCOPY is being used internally by the RECEIVE command.

```
12:24:09 Receiving XTDMF.TDM410.CNTL
12:24:10 INMR901I Dataset TDMS1.TDM410.CNTL from ALT00 on NODENAME 0 on NODENAME
12:24:10 INMR906A Enter restore parameters or 'DELETE' or 'END' + ' or 'END' +
12:24:10 INMR908A The input file attributes are: DSORG=PARTITIONED,
12:24:10 RECFM=FB, BLKSIZE=6160, LRECL=80, File size=58K bytes + G=PARTITIONED,
12:24:10 RECFM=FB, BL
12:24:10 INMR909A You may enter DSNAME, SPACE, UNIT, VOL, OLD/New, or
12:24:10 RESTORE/COPY/DELETE/END VOL, OLD/New, or RESTORE/CO
12:24:10 INMR070I RECEIVE command terminated. Failure in partitioned dataset reloading process. + e in partitioned dataset re
12:24:10 IKJ56894I UTILITY DATA SET NOT ALLOCATED+
12:24:10 IKJ56894I STORAGE MANAGEMENT SUBSYSTEM DETECTED AN ERROR, EITHER MESSAGES WERE NOT REQUESTED OR A SYSTEM ERROR OCCURRED TED AN ERROR, EITHER MESSAG
12:24:10 RECEIVE command terminated with RC=12
12:24:10 END
```

Figure 1-13 Messages if TSO RECEIVE work data set is allocated to VIO before OA23595 is applied

This has been determined to be a problem in SMS as described by APAR OA23595.

The problem can be resolved by applying the fix to APAR OA23595, or by adjusting the SMS ACS routines to avoid allocation to VIO.

1.3.4 Large format data set support: Additional TSO support

z/OS V1.7 introduced large format data set support. Large format data sets are sequential data sets. In z/OS v1.9 support is added to TSO functions that did not initially support large format data sets.

A large format data set is one of the three sequential data set types. This data set might not actually contain a large amount of data. But the fact that it is large format means that it has the capability to grow large in size, beyond 64 K (65535) tracks.
There are three types of sequential data sets:

- Basic format: A traditional data set, as existed prior to z/OS V1.7. These data sets cannot grow beyond 64 K tracks per volume.

- Large format: A data set (introduced in z/OS v1.7) that has the capability to grow beyond 64 K tracks but can be very small. The significance is that after being defined as a large format data set, it can grow to over 64K tracks without further intervention. The maximum size is x’FFFFFFE’ or approximately 16M tracks per volume.

- Extended format: An extended format data set that must be DFSMS-managed. This means that it must have a storage class. These data sets can be striped, and can grow up to x’FFFFFFFE’ tracks per volume.

Updates have been made to the following commands and service to ensure that each can handle large format data sets:

- TSO TRANSMIT, RECEIVE
- PRINTDS
- REXX LISTDSI function
- CLIST LISTDSI statement
- REXX EXECIO command
- CLIST OPENFILE/GETFILE/PUTFILE I/O processing

These enhancements are internal, and remove the restriction in z/OS V1.7 and z/OS V1.8 that prevented use of large format data sets.

<table>
<thead>
<tr>
<th>Restriction reminder: The following types of data sets cannot be allocated as large format data sets:</th>
</tr>
</thead>
<tbody>
<tr>
<td>- PDS, PDSE, and direct data sets</td>
</tr>
<tr>
<td>- Virtual I/O data sets, password data sets, and system dump data sets</td>
</tr>
</tbody>
</table>

**1.3.5 Large format data set support: Additional z/OS FTP support**

The z/OS FTP function has been enhanced by APAR PK48337 for large format data set support. APAR PK48337 is available for z/OS V1.9 and for z/OS V1.8 and z/OS V1.7.
Chapter 2. DFSMSdss Fast Replication use of new SFI support

In this chapter we discuss the Storage Facility Image (SFI) as it relates to DFSMSdss Fast Replication.

We cover the following topics:

- DFSMSdss Fast Replication and SFI: Overview
- Storage class attribute influence
- Storage Facility Image: Level of support
- DFSMS volume allocation selection with SFI attribute
2.1 DFSMSdss Fast Replication and SFI: Overview

The Storage Facility Image (SFI) is currently the boundary on which DFSMSdss Fast Replication works. An SFI corresponds to a physical control unit. A source volume of a DFSMSdss request located in one SFI cannot have a target volume in a different SFI.

DFSMSdss Fast Replication function requires that all pieces of a data set reside in the same SFI. A multi-volume data set that was allocated across multiple SFIs is not eligible for the Data Set Fast Replication operation.

In order to take advantage of the Data Set Fast Replication function, SMS Volume Selection has been enhanced to prefer candidate volumes that are in the same SFI, when allocating or extending an SMS managed multi-volume data set that has point-in-time copy volumes requested using the ACCESSIBILITY parameter in the storage class. The point-in-time copy volumes are requested, when the ACCESSIBILITY parameter equals CONTINUOUS or CONTINUOUS PREFERRED.

In this section we discuss how it works and how to make the optimum usage of the change.

2.2 Storage class attribute influence

Storage class settings influence SMS Volume selection as described in DFSMS Storage Administrator Reference, SG24-7402. To have the new SFI enhancement taken into consideration, the storage class ACCESSIBILITY parameter has to have Continuous specified. The settings shown below are possible on the storage class ACCESSIBILITY parameter:

- **Continuous**: Specifies that DFSMS must select point-in-time copy volumes and reject non-point-in-time copy volumes
- **Continuous Preferred**: Specifies that DFSMS prefers point-in-time copy volumes over non-point-in-time copy volumes
- **Standard**: Specifies that DFSMS prefers non-point-in-time copy volumes over point-in-time copy volumes
- **Nopref**: Specifies that DFSMS ignores point-in-time copy capability and treats both point-in-time and non-point-in-time copy volumes equally.

You can further specify the values for the Versioning and Backup subparameter in the storage class to control which volumes are preferred for the copy. This enhancement will prefer point-in-time volumes in the same SFI when allocating or extending an SMS managed multi-volume data set regardless of the Versioning and Backup settings.

2.3 Storage Facility Image: Level of support

Even if Storage Class parameters are set to Continuous, the SFI support will not always be possible due to specific conditions in the individual allocation or extent situations. Next we discuss the conditions involved.

**Note**: The DFSMS storage group might have volumes in one or more SFIs.
When SFI is considered
SFI is considered under the following circumstances:

- SFI is considered when allocating a multi-volume data set with ACCESSIBILITY
  Continuous or Continuous Preferred.
- If the data set already exists, it must be limited to one SFI.
- A primary volume meets all volume selection criteria.
  A volume that does not meet the SFI criteria is not a primary volume. Secondary volumes,
  however, that meet the criteria are ranked higher, when more important preference
  attributes are equal

When SFI is not considered
SFI is not considered under the following circumstances:

- ACCESSIBILITY is set to STANDARD or NOPREF, where non-point-in-time copy
  volumes are preferred or there is no preference at all.
- A data set is a single volume data set.
- Guaranteed space allocation points to different SFIs.
- A data set already exists on different SFIs.
- No storage groups have sufficient volumes within the requested SFI.
- No storage groups have sufficient unique controllers in the same SFI to meet the stripe
  count for a striped allocation.
- Space Constraint Relief is in effect.

DFSMSdss issues the message IGD17395I to indicate why a multi-volume data set is not
allocated in the same SFI.

2.4 DFSMS volume allocation selection with SFI attribute

When an allocation is eligible for SFI affinity, DFSMS volume selection takes the SFI attribute
into consideration on all types of allocation. The effect of SFI consideration on each of the
allocation types is discussed here.

Normal allocation
SMS volume selection for normal allocation will select the volumes with the highest
cumulative preference values. Volumes in the same SFI will have the highest preference.
After having selected the first volume, DFSMS will select the remaining volumes from the
same storage group and continue to prefer volumes within the same SFI.

Striping allocation
With the new SFI enhancement, DFSMS will regard a storage group as primary, when it
contains a sufficient number of unique control units to meet the requested stripe count. If the
storage group is unable to meet the stripe count, it is regarded as secondary. DFSMS will
prefer a primary storage group. If no primary storage group is available, DFSMS will use the
existing approach not to take SFI into consideration.
**Best-fit allocation**

Best-fit allocation is used by DFSMSdss during restore or recall. When best-fit is used, DFSMS selects the storage group that has the most available space within the same SFI. The selected volume will be the highest ranked in the volume list and the remaining volumes will be selected from the same SFI.

If there is an insufficient number of volumes in the same SFI to meet the need in the allocation, the remaining volumes will be selected from the highest ranked volumes from a different SFI.

**EOV: New volume extend**

When DFSMS is called to extend an SMS managed data set to a new volume, SMS Volume Selection will prefer a volume in the same SFI, when the following conditions are met:

- ACCESSIBILITY is set to Continuous or Continuous Preferred in the storage class.
- A data set only resides on volumes in one SFI. In the case of keyed VSAM data sets, both data and index must reside in the same SFI.

**Migration considerations**

This enhancement allocates or extends multi-volume data sets to the volumes that are in the same SFI when possible.

Volumes and storage groups that meet the SFI requirement might see more multi-volume data sets in the part of a storage group belonging to one SFI than in a part belonging to another SFI, making it more difficult to balance allocations in a storage group.

The SFI support is transparent to the user as soon as they have specified the ACCESSIBILITY parameter as CONTINUOUS. Extended use of SFI allocation might require attention to having sufficient capacity in the Storage Group on the SFI level.

**Note:** There is no way you can guarantee that DFSMSdss Fast Replication allocations and extents will happen in the same SFI, if SFI requirements are not met, unless you keep your storage groups at a SFI level. Traditional I/O methods, which are much slower, will instead be used by DFSMSdss for the copy operation if a multi-volume data set is allocated across multiple SFIs.
DFSMSShsm V1.9 enhancements

In this chapter we discuss new and changed functions in DFSMSShsm V1.9.

We cover the following topics:

- DFSMSShsm enhancements
- New cross-memory service invoking DSS
- Functional statistics record (FSR) improvements
- Return priority exit ARCRPEXT changes
- System Logger handling of DFSMSShsm recalls
3.1 DFSMSShsm enhancements

In this section, we introduce the following enhancements:

► New cross-memory service invoking DSS
► Functional statistics record (FSR) improvements
► Return priority exit ARCRPEXT changes

3.1.1 New cross-memory service invoking DSS

DFSMShsm captures UCBs in storage below the 16 MB line because various invoked functions require it. Installations that heavily use DFSMShsm, by running many simultaneous tasks, have experienced 878 abends. These abends are the result of exhausting below the line storage in the DFSMShsm private area.

With z/OS V1.9, DFSMShsm will invoke DFSMSdss through its cross-memory application interface (ADRXMAIA) for all functions except RECALL.

Invoking DFSMSdss in a separate address space reduces the amount of storage used in the DFSMShsm address space. DFSMShsm will request a unique DFSMSdss address space for each function and Host ID. One of these DFSMSdss address spaces is started automatically by DFSMShsm whenever a dump, restore, migration, backup, recover, or CDS backup function is invoked. A DFSMSdss address space is not started for RECALL tasks.

Details on new DFSMSdss address spaces created

The address space identifier for each non-Fast Replication function will be in the format of ARnXXXX, where:

► n is the unique DFSMShsm Host ID.
► XXXX represents the function, as follows:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUMP</td>
<td>DFSMShsm Dump</td>
</tr>
<tr>
<td>REST</td>
<td>DFSMShsm Restore</td>
</tr>
<tr>
<td>MIGR</td>
<td>DFSMShsm Migration</td>
</tr>
<tr>
<td>BKUP</td>
<td>DFSMShsm Backup</td>
</tr>
<tr>
<td>RCVR</td>
<td>DFSMShsm Recover</td>
</tr>
<tr>
<td>CDSB</td>
<td>DFSMShsm CDS Backup</td>
</tr>
</tbody>
</table>

For example, AR1MIGR is the address space name for migration tasks on Host 1.

Note: The address spaces are started automatically when the functions are invoked on the DFSMShsm host. The address spaces exist until DFSMShsm is shut down. The address spaces automatically terminate when DFSMShsm is shut down.
Displaying the address spaces
To see the spawned address spaces, issue a DISPLAY command as shown in Figure 3-1.

<table>
<thead>
<tr>
<th>DFSMSdss</th>
<th>ARC3MIGR</th>
<th>IEFPROC</th>
<th>NSW *</th>
<th>A=00A8 PER=NO SMC=000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PGN=N/A DMN=N/A AFF=NONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CT=000.022S ET=01.05.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>WUID=STC03360 USERID=STC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>WKL=STCTASKS SCL=STC P=1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RGP=N/A SRVR=NO QSC=NO</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ADDR SPACE ASTE=78B1FA00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DFSMSdss</th>
<th>ARC3CDSB</th>
<th>IEFPROC</th>
<th>NSW *</th>
<th>A=0074 PER=NO SMC=000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PGN=N/A DMN=N/A AFF=NONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CT=000.071S ET=104.695S</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>WUID=STC03361 USERID=STC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>WKL=STCTASKS SCL=STC P=1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>RGP=N/A SRVR=NO QSC=NO</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ADDR SPACE ASTE=78B20D00</td>
</tr>
</tbody>
</table>

**Figure 3-1** Display DFSMShsm address spaces

**Note:** These address spaces were created when issuing a DFSMShsm MIGRATE and CDS BACKUP command on a LPAR whose DFSMShsm HOST ID is “3”.

You can also do a SDSF display to see the HSM DFSMSdss address spaces as shown in Figure 3-2.

**Figure 3-2** SDSF display active

Large block interface considerations
When z/OS V1.9 DFSMShsm invokes DFSMSdss through the cross-memory interface (ADRXMAIA) so that DFSMSdss can run in its own address space, the DFSMSdss large block interface (LBI) function, introduced by APAR OA13742, is not supported. If you prefer to have the DFSMSdss LBI function rather than the cross-memory function, you can disable DFSMShsm's use of the cross-memory function with the following patch in the ARCCMDxx member:

```
PATCH .MCVT.+433 X'00'
```

If ARCCMDxx includes the patch PATCH .MCVT.+432 BITS(XX......), where X is a 0 or 1, remove the patch.
Migration considerations
The new cross-memory service creating the DFSMSdss address spaces will be invoked by default when you install z/OS V1.9. If you choose not to enable the service immediately after installing z/OS V1.9, you can disable it by using the patch mentioned under “Large block interface considerations” on page 25. If you choose to let the service be enabled by default, make sure the following tasks are done in preparation:

- Have the started task names created, known to RACF and connected to a user ID.
- Consider having WLM differ on the priority of the individual DFSMSdss tasks. We recommend setting the priority on the backup task the highest, due to system wide exclusive enqueue held on DFSMShsm resources during backup.

3.1.2 Functional statistics record (FSR) improvements

The function statistics record (FSR) is a control block that contains statistics for a particular function that is performed on a data set. It is maintained in the DFSMShsm work space until the data set processing has completed. Upon completion of the function, the record is written to the DFSMShsm log and accumulated by category into daily statistics records (DSR) and volume statistics records (VSR) in the migration control data set.

Function statistics records are key to performing problem analysis with DFSMShsm. This enhancement includes a number of additions to the information recorded in the FSR:

- Indicate in the FSR when a recall caused a tape takeaway.
- Record in the recall FSR the number of times a migrated data set had been recycled before it was eventually recalled.
- Add the recycle source volume volser to the FSR.
- Provide the CPU time used for a partial release (FSR type 18) and the expiration of a backup version (FSR type 19).
- Record the number of tracks needed when an error occurred due to not enough ML1 space.
- The FSRSTAT program was updated to analyze the new FSR field for recall tape takeaway.

Details of the FSR enhancement
Each FSR contains values for some fields defined in the ARCFSR macro. Some fields have multiple uses. The same field means different things based on the FSRTYPE or function. One new bit called FSRF_RECALL_TAKEAWAY is added although the size of the FSR record remains the same. DFSMShsm modules have been changed to calculate the new field and save it in the FSR record.
Table 3-1 lists the new FSR field name, type, and information provided.

<table>
<thead>
<tr>
<th>FSR field name</th>
<th>FSR offset # ('X')</th>
<th>Type</th>
<th>Length</th>
<th>FSRTYPE</th>
<th>Information provided</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSRF_RECALL_TAKEAWAY</td>
<td>222 (DE)</td>
<td>Bitstring</td>
<td>5</td>
<td></td>
<td>RECALL caused a tape takeaway when set to 1</td>
</tr>
<tr>
<td>FSRF_RECYCLE_COUNTER</td>
<td>191 (BF)</td>
<td>Integer</td>
<td>1</td>
<td>5</td>
<td>Number of times the data set being recalled was recycled on ML2</td>
</tr>
<tr>
<td>FSRF_RECYCLE_COUNTER</td>
<td>191 (BF)</td>
<td>Integer</td>
<td>1</td>
<td>12</td>
<td>Number of times the data set has been recycled on ML2 since its last migration</td>
</tr>
<tr>
<td>FSR_RECYCLE_SOURCE_VOLSER</td>
<td>98 (62)</td>
<td>Character</td>
<td>6</td>
<td>10 and 12</td>
<td>Source volume for recycle</td>
</tr>
<tr>
<td>FSR_CPU</td>
<td>180 (B4)</td>
<td>Integer</td>
<td>4</td>
<td>18</td>
<td>CPU time used for partial release processing</td>
</tr>
<tr>
<td>FSR_CPU</td>
<td>180 (B4)</td>
<td>Integer</td>
<td>4</td>
<td>19</td>
<td>CPU time used for expire incremental backup version function</td>
</tr>
<tr>
<td>FSR_PSQTY</td>
<td>292 (124)</td>
<td>Integer</td>
<td>4</td>
<td>1 and 2</td>
<td>The number of tracks still needed when an error occurred due to &quot;NOT ENOUGH SPACE ON ML1 VOLUME&quot;</td>
</tr>
</tbody>
</table>

You can use the DFSMSHsm REPORT command or SMF records to view the FSR information.

### 3.1.3 Return priority exit ARCRPEXT changes

The ARCRPEXT return-priority exit is taken as each DELETE, RECALL, or RECOVER request, in the form of a management work element (MWE), is about to be queued on one of DFSMSHsm’s functional subtask queues for processing. For a RECALL request, this exit is invoked by the host initiating the RECALL.

#### Extent reduction

Extent reduction is part of HSM’s primary space management (PSM). The extent threshold is defined by SETSYS MAXEXTENTS(x) command. During PSM, if DFSMSHsm finds a data set that is not eligible for migration (for example, too young) but has exceeded the MAXEXTENTS value, HSM will schedule a MIGRATE and immediate RECALL just for extent reduction.

The problem was that DFSMSHsm did not invoke the ACS routines for RECALL processing and always recalled the data set back to its original volume.
The ARCRPEXT exit has been enhanced to give you a choice for data set RECALL during extent reduction processing. A field has been added to the exit to enable you to indicate that ACS routines should be invoked for extent reduction.

**Enhancement details**

A new bit field “EXTRDCTN” is added to the output parameter list for exit ARCRPEXT.

On return from exit ARCRPEXT, this field is set to either 0 or 1:

- If the bit field is 0, the data set will be recalled to its original volume.
- If the bit field is 1, MWEFEXT_SAMEVOL = OFF is set.

The existing algorithm for ACS routines is then followed.

**Note:** The bit field added to the output parameter list is initialized to 0. This is the default.

### 3.2 System Logger handling of DFSMShsm recalls

The implementation of System Logger recalls of migrated log stream data sets is single threaded and synchronous prior to z/OS V1.9.

This means that every RECALL has be to completed before the next RECALL request can happen. Even if the RECALL request can happen in two tasks (for example, production and test), this causes limited or slow access to log stream resources during offload or application reads. In fact, a RECALL request for one logstream data set can hold up a request for another logstream request.

IBM strongly recommends not migrating System Logger staging data sets and avoiding migrating offload data sets whenever possible. If migration is necessary because of DASD space constraints, migrate the low priority offload data sets for which associated applications can handle delays in accessing the data sets.

Enhancements in z/OS V1.9 allow for multiple concurrent asynchronous recalls, providing the ability to display data sets being currently recalled by System Logger. This enables you to stop System Logger from waiting for recalls causing problems.

**Managing System Logger recall issues**

System Logger allows both log stream offload and staging data sets to be migrated. System Logger will automatically recall data sets when they are needed. You can migrate and recall data sets using DFSMShsm or similar protocol.

System Logger now allows up to 24 outstanding asynchronous recalls for Production Group and 8 for Test Group. New messages will report progress of the recall process:

- Message IXG271I is issued, when a System Logger migrated task is not making progress. The data set name for the oldest outstanding request will be displayed in the message text.
- Message IXG281I will notify you of outstanding recalls caused by:
  - Maximum number of recalls for the group is pending
  - One single asynchronous recall is pending for more than 30 seconds.
System Logger monitors log-stream-offload activity as well as specific log stream data set allocation requests and notifies the installation if an offload appears to be hung or is taking too long.

**Enhanced DISPLAY command**

If System Logger is delayed due to data set recalls, consider using the new DISPLAY command to diagnose recall delays. Operational messages indicating recall activity are IXG278i, IXG279i, IXG280i and IXG281i. The format of the command is:

```
DISPLAY LOGGER,STATUS[,RECALLS]
```

STATUS (or ST) is the default, when no parameters are specified. RECALLS (or REC) displays outstanding asynchronous recall requests System Logger has made to DFSMShsm through the ARCHRCAL service.

Figure 3-3 shows the output of the DISPLAY command. There is one outstanding RECALL request. In case of any outstanding requests, the DISPLAY output will show the data set names by group and the number of seconds, each have been in process.

```
Display LOGGER,ST,REC
IXG601I hh.mm.ss LOGGER DISPLAY
SYSTEM LOGGER STATUS
SYSTEM SYSTEM LOGGER STATUS
SY1 ACTIVE
   LOGGER DATA SET RECALLS
   GROUP: PRODUCTION
   SECONDS DATA SET NAME
   00000033 IXGLOGR.PROD.STREAM01.A0000025
GROUP: TEST NO DATA SET RECALLS WAITING
```

*Figure 3-3  DISPLAY LOGGER command output*

If services on any group have been disabled, message IXG601I will reflect this, as indicated by the group name and the message *Logger no longer recalling data sets asynchronously.*

**SETLOGR FORCE,NORECALL command**

If System Logger activity is delayed by RECALL activity, consider using the new command:

```
F SETLOGR FORCE[,NORECALL,DSName=data set name]
```

The SETLOGR FORCE,NORECALL command causes System Logger to stop waiting for a particular data set to be recalled. If System Logger message IXG272E is displayed, reply to this message only after you have attempted to correct any general allocation or recall related messages.

Other keywords on the SETLOGR command are DISCONnect and DELete.

More than one RECALL request can be outstanding for the same data set name. The SETLOGR command will affect all these recalls for the named data set.
Figure 3-4 shows the output from the command entered to stop System Logger from waiting on the current recall request for the data set XGLOGR.PROD.STREAM01.A0000025.

```
SY1 setlogr force,norec,dsn=IXGLOGR.PROD.STREAM01.A0000025
SY1 IXG651I SETLOGR FORCE NORECALL COMMAND ACCEPTED FOR DSN=IXGLOGR.PROD.STREAM01.A0000025
SY1 IXG280I IXGLOGR RECALL REQUEST STOPPED BY SETLOGR COMMAND FOR DSN=IXGLOGR.PROD.STREAM01.A0000025
SY1 IXG661I SETLOGR FORCE NORECALL PROCESSED SUCCESSFULLY FOR DSN=IXGLOGR.PROD.STREAM01.A0000025
```

**Figure 3-4  SETLOGR FORCE,NORECALL command output**

## Preventing recalls at offload writing

The offload OFFLOADRECALL(NO) parameter in the System Logger setup indicates whether offload processing is to recall the current offload data set. If you specify OFFLOADRECALL(YES), offload processing should recall the current offload data set. If you specify OFFLOADRECALL(NO), offload processing should skip recalling the current offload data set and allocate a new one.

Note that this option might cause any or all of the current offload data set to be wasted space on DASD after it is recalled. When using this option, care should be taken to size the data sets appropriately.

With OFFLOADRECALL(NO), System Logger will request that allocation does not wait on any ENQ serialization contention to be resolved, and will receive a class two type error (unavailable system resource), as described in the chapter, "Interpreting Error Reason Codes from DYNALLOC," in the manual *z/OS MVS Programming: Authorized Assembler Services Guide*, SA22-7608.

With OFFLOADRECALL(YES), System Logger will not be able to offload until the current offload data set can be recalled. This happens if DFSMShsm is not available, and either:

- The current offload data set for one or more of the logstreams residing in the structure is migrated, while the structure is being rebuilt.
- Logger is going through recovery for one of the logstreams whose current offload data set is migrated.

**Migration**

z/OS will automatically make the new support available.

Consider using the offload parameter OFFLOADRECALL(NO) to prevent System Logger from waiting for a data set recall at offload writing.

This keyword can be updated even when the log stream is actively connected. The change is immediately reflected in the log stream definition. It will take effect on the subsequent first connection to the log stream in the sysplex. For a structure-based log stream, the change will also take effect during the next structure rebuild. However, for a DASD-only log stream, the change will take effect upon the next offload data set switch.
When the above cases are encountered, System Logger allocates a new current offload data set to continue offloading the log data from the primary or interim storage. When the OFFLOADRECALL(NO) parameter is specified, this enhancement can aid in reducing the impact of situations where the current offload data set is not immediately available due to ENQ contention issues by allowing offload processing to continue writing to the newly allocated current offload data set.

You can control the offload situation better in relation to DFSMShsm by using the new display and action commands when you are in a hang situation, to stop System Logger from waiting for a DFSMShsm recall:

- DISPLAY LOGGER,STATUS,RECALLS
- F SETLOGR FORCE,NORECALL

For additional information, refer to the chapter, “Planning for System Logger Applications,” in the manual, *MVS Setting Up a Sysplex*, SA22-7625.
ISPF enhancements in z/OS V1.9

In this chapter, we discuss z/OS V1.9 ISPF enhancements, which can help the storage administrator do their daily tasks:

- Instead of having LPAR-specific profile data sets, ISPF V1.9 now enables support for concurrent sharing of ISPF profile data sets across a sysplex.
- ISPF V1.9 enables editing, browsing, and viewing of z/OS UNIX® files.
- System symbols can be specified within data set names and volumes entered on ISPF panels.
- The ISPF Data Set List Utility now totals the number of tracks and selected data sets on top of a list.
4.1 ISPF profile sharing

Profile sharing is intended to be used by the same user logging concurrently on to multiple systems. The contents of the ISPF profile are often user-specific.

Customizing for profile sharing shows how you can customize shared profiles within ISPF.

The ISPF Configuration utility provides an option to specify an additional qualifier for ISPF temporary data sets, including Log, List, and temporary control and work data sets. The Multi-Logon Profile Sharing Settings panel shows your settings, after the support is enabled. To display this panel, perform one of these actions:

- Select the **Environ** choice on the ISPF Settings panel action bar, then select option 2 **Shared Profile settings...**.
- Issue the ISPF system command **SHRPROF** from any ISPF command line.

The panel text provides an overview of the command and its parameters.

4.2 Browsing and viewing z/OS UNIX files

The ISPF Edit, Browse, and View functions have been enhanced to support the processing of z/OS UNIX files. The ISPF edit, browse, and view panels have been modified to indicate the new support with the text:

**Other Partitioned, Sequential or VSAM Data Set, or z/OS UNIX file**

When viewing z/OS UNIX files, you might want to use the record length specification. The numeric value entered in this field is used by ISPF to display the data in the file as fixed-length records, rather than using the newline character to delimit each record. This is useful for browsing files that would otherwise have very large records if the newline character is used as the record delimiter.

The ISPF Edit option (option 2) allows you to create, display, and change data stored in ISPF libraries, other partitioned or single-volume or multivolume sequential data sets, or z/OS UNIX files with these characteristics:

- **Record Format (RECFM):**
  - Fixed or variable (non-spanned)
  - Blocked or unblocked
  - With or without printer control characters
  - Logical Record Length (LRECL)
  - From 1 to 32 760, inclusive, for fixed-length records
  - From 5 to 32 756, inclusive, for variable-length records.

- **VSAM data:**
  - VSAM data can be edited if the ISPF configuration table has been customized to enable VSAM support (that is, **VSAM>Edit_ENABLED** is set to **YES**).

- **z/OS UNIX files**
4.3 System symbolics in data set name and volume

System symbols can now be specified within data set names and volumes entered on ISPF panels.

4.4 ISPF Data Set utility list totalling

The ISPF Data Set Utility View now shows a totalling of data sets selected and sums the number of tracks in total for the data sets listed for the Space and Total view (see Figure 4-1). The available views are:

- Volume: This view shows a data set list that contains data set names and the volumes on which they reside.
- Space: The Space view shows a data set list that contains data set names, tracks, percentages used, extents, and devices. An additional header line, displayed above the column headings and showing the total tracks of all data sets, the total tracks of all non-excluded data sets, the number of data sets listed and the number of non-excluded data sets listed, is displayed if the Display Total Tracks option is selected.
- Display total tracks: Use this option to display an additional header line on the Space or the Total view, showing the total tracks of all data sets, the total tracks of all non-excluded data sets, the number of data sets listed, and the number of non-excluded data sets listed. Depending on the size of the data set list, processing time increases because information about the tracks for all data sets has to be collected before the list is displayed.

![Figure 4-1 Example of ISPF Data Set Utility List using Total](image)

When the list comprises 50 data sets or more, a pop-up panel is displayed, indicating the progress of the data collection.
PDS and PDSE enhancements in z/OS V1.9

In this chapter we discuss the z/OS V1.9 support for PDS and PDSE.

We cover the following topics:
- PDS and PDSE changes
- Migration: PDSE
- Migration: PDS
- PDS/PDSE Volcount restriction
5.1 PDS and PDSE changes

Here we discuss items that have changed and might need your attention:

- Migration: PDSE
- Migration: PDS
- Removal of Dynamic Volume Count restriction for PDS/PDSE

5.2 Migration: PDSE

Migration from z/OS 1.7 requires the PTF for the following APAR to be applied on z/OS 1.7 to allow access to objects created by the z/OS V1.8 or z/OS V1.9 Program Binder:

- OA13294
- OA13525

5.3 Migration: PDS

The program binder in z/OS V1.9 has been enhanced to prevent access to an output PDS that does not have RECFM=U. This is to protect PDS data sets that are not intended to contain load modules. This change makes the behavior of the program binder when using a PDS for output the same as it is when the output is a PDSE.

If it is necessary to use the program binder to store output in an existing PDS that does not have RECFM=U, the check can be overridden by explicitly coding RECFM=U on the output DD statement. The result of doing this might be unpredictable.

5.4 PDS/PDSE Volcount restriction

z/OS V1.8 introduced a check to prevent a PDS from being allocated if the JCL or SMS parameters requested a volume count of greater than 1. A PDS/PDSE cannot span volumes. Although this is a long standing restriction, it was only with z/OS V1.8 that the check was implemented so that an allocation would fail.

Users are exposed to this issue if, for example, they code JCL that specifies a dynamic volume count greater than 1, or if they use the JCL LIKE= (dsname) option and the specified data set has a dynamic volume count greater than 1.

In the first case, the JCL can be changed relatively easily. In the second case, there might be some difficulty finding a suitable replacement reference data set.

APAR OA22738 has been created to address this dynamic volume count issue (by removing message IGD17295I and support for determining if data sets are partitioned or sequential). The correct data class and volume count have to be controlled by JCL or the ACS routines.
DFSMSdfp OAM V1.9 enhancements

In this chapter we discuss the OAM enhanced support in z/OS V1.9.

We cover the following topics:
- DFSMSdfp OAM enhancements
- Migration considerations
- Implementing the OAM enhancements
- OAM-DB2 table changes
6.1 DFSMSdfp OAM enhancements

The Object Access Method (OAM) storage hierarchy consists of three levels (disk, optical, and tape). To offer further granularity of the tape level, two new sublevels within the tape level of the OAM storage hierarchy are introduced. This effectively expands OAM’s storage hierarchy into four levels:

- Disk
- Optical
- Tape sublevel 1 (TSL1)
- Tape sublevel 2 (TSL2)

In addition to enabling the ability to write and read object data directly to and from a given tape sublevel, OAM provides the ability to transition object data within the tape family (for example: from VTS to native tape). An installation will be able to move data freely in and out of all four hierarchy levels via the OSREQ macro and the OAM storage management component (OSMC) functions.

6.2 Migration considerations

Regardless of whether or not your installation intends to exploit the new function, you must modify and run the OAM DB2® migration job "CBRSMSR19 from SYS1.SAMPLIB to add the new TSL column to the DB2 TAPEVOL table, and prime it with:

- 1 for grouped volumes
- Blank for scratch and backup volumes

If the customer is running in an OAMplex, and is sharing data across systems, the system administrator must ensure that all systems are capable of tape sublevel support, prior to enabling the new support. To ensure that all systems are capable of tape sublevel support, all systems in the OAMplex must have the OAM, ISMF, and SMS maintenance and toleration PTFs for tape sublevel support installed.

For more information about the software requirements for DFSMS, see z/OS Migration, GA22-7499.

For additional information about z/OS coexistence and release migration information, refer to z/OS Planning for Installation, GA22-7504.

6.3 Implementing the OAM enhancements

Next we discuss the following topics, leading you through the implementation process:

- Creating the new SUBLEVEL entry in the TAPEVOL table
- Updating DFSMS storage class constructs
- Defining tape sublevel parameters to OAM
- Modifying the SETOAM keywords
- Displaying the new OAM tape level settings
Creating the new SUBLEVEL entry in the TAPEVOL table

A new job, CBRSMR19 in the SAMPLIB, performs migration from the z/OS V1R8 version of the OAM Configuration Database to the z/OS V1R9 version, which supports tape sublevels.

Updating DFSMS storage class constructs

You can set the new OAM Sublevel parameter (OSL) in ISMF to modify existing storage class constructs or create new storage class constructs associated with tape sublevels TSL1 and TSL2 as shown in Figure 6-1. Pre-existing tape storage classes will default to TSL1. These storage class construct changes might necessitate updates to ACS routines. Each level of the storage hierarchy is associated with SMS Storage Class (SC) constructs.

These constructs were previously defined with the Initial Access Response Seconds (IARS) and the Sustained Data Rate (SDR) keywords in the ISMF storage class definition panels. The tape sublevel support adds a new OAM Sublevel (OSL) parameter to the storage class, to indicate the sublevel associated with that storage class. Values are as follows;

- **Initial access response seconds (IARS):**
  - 0 = DASD
  - 1-9999 = removable media
  - Blank will fail the request

- **Sustained data rate (SDR):**
  - 0-2 = Optical
  - 3-999 = Tape volume

- **OAM sublevel (OSL):**
  - 1 = OAM sublevel 1 (default)
  - 2 = OAM sublevel 2

**Note:** A storage class defined with IARS=1, SDR=3 and OSL=2 would equate to TSL2.

![Figure 6-1 OAM Sublevel Storage Class parameter](image)

1 The number within the range is unimportant, because it will only direct the request to removable media.
2 The number within the range is unimportant, because it will only direct the request to tape.
Defining tape sublevel parameters to OAM

The OAM component’s PARMLIB member is CBROAMxx, where xx is the unique suffix specified by the use of OAM=xx parameter in the OAM started procedure, the START command for the OAM started procedure, or in the F OAM,RESTART,OAM=xx command. This PARMLIB member is used by OAM during the OAM address space initialization process to determine parameters and configuration information to be used while the address space is active.

New keywords, L2TAPEUNITNAME and L2DATACLASS, have been added to the SETOAM statement in the CBROAMxx member of PARMLIB in support of the new tape sublevel function.

L2DATACLASS(name) is an optional parameter that specifies the SMS data class to be used when storing objects to TSL2 for object storage groups that do not have their own L2DATACLASS specification on the STORAGEGROUP subparameter of the SETOAM statement. The Tape Sublevel is associated with the OAM Sublevel parameter specified in the DFSMS Storage Class construct.

When using the L2DATACLASS keyword on the SETOAM statement at the global level, you should consider the following effects:

- This specification of L2DATACLASS applies to each of the Object storage groups that do not explicitly specify a L2DATACLASS.
- It allows the installation to modify Tape Device Selection Information and volume expiration date for those Object storage groups that do not have an explicit L2DATACLASS keyword in their STORAGEGROUP subparameter list.

**Note:** L2DATACLASS does not apply to Object Backup storage groups.

L2TAPEUNITNAME(unitname) is a required subparameter of the STORAGEGROUP parameter, if using the TSL2 function. Tape Sublevel is associated with the OAM Sublevel parameter specified in the DFSMS Storage Class construct. For unitname, specify the name of a valid MVS esoteric (group of devices defined to a group name) or a generic unit name.

Valid generic unit names are:

- **3480** A base 3480 device
- **3480x** A 3480 device with the IDRC feature, or a base 3490 device
- **3490** A 3490E device
- **3590-1** A 3590 device (or a device that emulates a 3590-1)

**Note:** The L2TAPEUNITNAME keyword cannot be associated with an Object Backup storage group.
Figure 6-2 shows an example of a SETOAM statement in the CBROAM70 PARMLIB member.

```
EDIT       SYS1.PARMLIB(CBROAM70) - 01.04                  Columns 00001 00072
Command ===>                                                  Scroll ===> CSR
****** **************************** Top of Data ****************************
000001   OAMXCF OAMGROUPNAME(OAMTEST)                   /* OAM XCF         */
000002          OAMMEMBERNAME(OAM70)                    /* MEMBER NAME     */
000003   SETOAM DEMOUNTWAITTIME(90)                     /* GLOBAL LEVEL    */
000004          MAXRECYCLETASKS(1)
000005          MAINTAPEGETasks(2)
000006          MAINTAPESTORETASKS(2)
000007          MOUNTWAITTIME(10)
000008          TAPEDISPATCHERDELAY(40)
000009          TAPEFULLTHRESHOLD(50000)
000010          DSNWTHMSGNAME
000011          OAMSCRATCHSYNCH(ENABLED)
000012          TAPEGRECYCLEMODE(MVSCSCHRATCH)
000013   SETOAM STORAGEGROUP(T0AMBK1                    /* DEFAULT OBJ BKP */
000014          TAPEUNITNAME(ATL3)
000015          SGMAXTAPEGETasks(1)
000016          SGMAXTAPESTORETASKS(1)
000017          TAPEDRIVESTARTUP(5000)
000018          TAPEPERCENTFULL(98))
```

**Figure 6-2  SETOAM statement in a CBROAM00 PARMLIB member**

**Modifying the SETOAM keywords**

You can use the F OAM,UPDATE,SETOAM command to add or change the current L2DATACLASS and L2TAPEUNITNAME specifications for a storage group, or to add or change the L2DATACLASS specification at the global level.

This following command updates the L2DATACLASS value of the SETOAM statement in the CBROAMxx PARMLIB member, with `dataclas` signifying the data class to be used for scope. Specify UPDATE,SETOAM with the L2DATAACL or L2TAPEUN keyword.

```
F OAM,UPDATE,SETOAM,scope,L2DATAACL,dataclas
```

This example updates the L2TAPEUNITNAME value of the SETOAM statement in the CBROAMxx PARMLIB member, with `unitname` signifying the unit name to be used for scope.

```
F OAM,UPDATE,SETOAM,scope,L2TAPEUNIT,unitname
```
OAM operator commands
The following operator commands have been changed to include the new L2TAPEUNITNAME and L2DATACLASS keywords for the SETOAM statement, as shown in Figure 6-3.

```
F OAM,UPDATE,SETOAM
F OAM,DISPLAY,SETOAM
F OAM,DISPLAY,VOL
F OAM,START,RECYCLE
D SMS,STORGRP(grp-name),DETAIL
D SMS,OSMC,TASK(OSMC-task)
F OAM,START,MOVEVOL
F OAM,START,RECOVERY
```

Figure 6-3  OAM operator commands

Displaying the new OAM tape level settings
The F OAM,DISPLAY commands show that the new tape level 2 settings, if any, are in effect at the global and group levels as shown in Figure 6-4.

```
F OAM,D,SETOAM,TOGRP1
CBRI0751 TOGRP1 value for SGMAXTPS is 2
CBRI0751 TOGRP1 value for SGMAXTPR is 1
CBRI0751 TOGRP1 value for EXPDATE is /
CBRI0751 TOGRP1 value for TFULLTHR is 50000
CBRI0751 TOGRP1 value for TFULLPER is 98
CBRI0751 TOGRP1 value for TAPEUNIT is ATL3
CBRI0751 TOGRP1 value for L2TAPEUN is ATL3
CBRI0751 TOGRP1 value for DMNT is 90
CBRI0751 TOGRP1 value for DATACL is
CBRI0751 TOGRP1 value for L2DATACL is
CBRI0751 TOGRP1 value for TCOMP is N
CBRI0751 TOGRP1 value for TDRVSTRT is 5000
CBRI0751 TOGRP1 value for SGMAXREC is 1
```

Figure 6-4  OAM display command

6.4 OAM-DB2 table changes

A new SUBLEVEL column is added to OAM's TAPEVOL table in DB2 to indicate which tape sublevel (TSL1 or TSL2) each tape volume is associated with. Valid values are 1, 2, and blank, with the following meanings:

- 1: The volume is associated with TSL1.
- 1: The volume is associated with TSL2.
- Blank: The volume is not associated with a sublevel. This only applies to OAM scratch or backup volumes.
The ODLOCFL column in OAM’s Object Directory Table which indicates on what storage hierarchy the primary copy of a given object resides, was updated with a new indicator “U” for TSL2. See Figure 6-5.

The values for this column are now:

D = DASD
R = Recalled
Blank = Optical
T = Tape (TSL1)
U = Tape (TSL2)

<table>
<thead>
<tr>
<th>ODLOCFL</th>
<th>ODLSLOC</th>
<th>ODSECLOC</th>
<th>ODBKLOC</th>
<th>ODBKSEC</th>
<th>ODCLID</th>
<th>ODNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>THS008</td>
<td>15</td>
<td>THS011</td>
<td>355</td>
<td>1</td>
<td>PETEST..</td>
</tr>
<tr>
<td>U</td>
<td>THS009</td>
<td>5</td>
<td>THS011</td>
<td>357</td>
<td>2</td>
<td>PETEST..</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>PETEST..</td>
</tr>
</tbody>
</table>

Figure 6-5  Partial display of an object directory select

In Figure 6-6 you can see the correlation between the ODLOCFL information in the object directory table, the volume serial of the tape and the SUBLEVEL in the tapevol table.

<table>
<thead>
<tr>
<th>VOLSER</th>
<th>UNITNAME</th>
<th>MEDIATYP</th>
<th>STORGRP</th>
<th>TYPE</th>
<th>OUNITNAM</th>
<th>DSNFMT</th>
<th>SUBLEVEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>THS011</td>
<td>3590-1</td>
<td>07</td>
<td>TOGRP1B1</td>
<td>B</td>
<td>ATL3</td>
<td>G</td>
<td>1</td>
</tr>
<tr>
<td>THS008</td>
<td>3590-1</td>
<td>07</td>
<td>TOGRP1</td>
<td>G</td>
<td>ATL3</td>
<td>G</td>
<td>1</td>
</tr>
<tr>
<td>THS009</td>
<td>3590-1</td>
<td>07</td>
<td>TOGRP1</td>
<td>G</td>
<td>ATL3</td>
<td>G</td>
<td>2</td>
</tr>
</tbody>
</table>

Figure 6-6  Partial display of the SELECT * FROM CBROAM.TAPEVOL

SMF (type 85)

SMF type 85 records have been enhanced to reflect information regarding tape sublevel information. Several SMF OAM subtypes are modified to report tape sublevel information as follows:

Subtype 2  OSREQ STORE
Subtype 3  OSREQ RETRIEVE
Subtype 6  OSREQ DELETE
Subtype 32 OSMC Storage Group Processing
Subtype 40  Tape RECYCLE Command
Subtype 78  LCS Write Tape Request
Subtype 79  LCS Read Tape Request
Subtype 87  Demount Tape Volume
RLS dump space enhancements in z/OS V1.9

In this section we discuss the z/OS V1.9 enhancements to VSAM Record Level Sharing.

We cover the following topics:

- RLS dump space considerations
- Dump space in a sysplex for VSAM RLS abends
- Migration considerations
7.1 RLS dump space considerations

Before z/OS V1R9, when a VSAM RLS abend occurred, an abend dump was taken only on the originating system where the abend occurred. In z/OS V1R9, new sysplex-wide dumping support causes other systems running VSAM RLS in the sysplex to take dumps as well as the originating system.

7.2 Dump space in a sysplex for VSAM RLS abends

In z/OS V1R9, new sysplex-wide dumping support causes other systems running VSAM RLS in the sysplex to take dumps as well as the originating system. The change is introduced to assure the necessary documentation from the entire sysplex as RLS works on a sysplex level.

This change will be introduced after the first IPL on z/OS V1R9.

All systems (lower level systems as well) will be dumped if the z/OS V1.9 system is the one that detects the error, but if a lower level system detects the problem, only that system will be dumped.

The change will result in more dump space being used, because all systems in the sysplex now participate, when an RLS abend ends up in a dump being created.

7.3 Migration considerations

If the current dump space is limited, you might consider having the current dump space used by RLS abends extended to concurrently support dumps from all systems in a sysplex.

If more space for RLS abend dumps is needed, use the same procedures you currently do to increase dump data set size, but be aware that this might need to be done more frequently than in the past. In addition, you can use DAE to suppress dumps if needed.
The DFSMSrmm enhancements in DFSMS V1.9 provide improvements in the areas of new hardware support, checking of long running requests, improved interaction with system-managed libraries, unqualified data set name support, and subcommand enhancements.

In this chapter we discuss the following enhancements:

- Task management support
- Multitasking of utilities
- Control data set (CDS) serialization
- JCL data set names
- Data set names in RMM subcommands
- Shared PARMLIB support
- TSO subcommands
- 3592 Model E05 software support
- REPORT17 of EDGRRPTE REXX exec
- Migration and coexistence considerations
- Common Information Model (CIM) provider

With DFSMSrmm V1.9 enhancements, you can manage your removable media as one enterprise-wide library across systems and sysplexes, and manage your installation's tape volumes and the data sets on those volumes. DFSMSrmm also manages the shelves where volumes reside in all locations except in automated tape libraries.
8.1 Task management support

Any task in the system that requests DFSMSrmm subsystem services and fails, or is interrupted because a TSO-user used Attention (ATTN), or is cancelled by the operator, results in any corresponding long-running subsystem request failing. In addition, there are checkpoints built into long-running requests so that when the requestor ends (such as a job being cancelled), DFSMSrmm processing is interrupted at a safe and convenient point. Long running local tasks are DFSMSrmm subsystem requests that last long enough to be included in the results of a QUERY ACTIVE command, and the task token can be obtained and used.

If the requester is inventory management, the results of the partial processing are available in the MESSAGE file. The following long running tasks support interruption:

- EDGHSKP inventory management, VRSEL, DSTORE, EXPROC, RPTEXT, and CATSYNCH. EDGHSKP ends with return code 12 when cancelled.
- SEARCH subcommands. These end with return code 4, reason code 16 when cancelled.
- EDGINERS when building lists of volumes to process. EDGINERS processing is still attempted even though one or more search requests of the DFSMSrmm control data set might have been cancelled by the operator. Also, cancelling a task that is processing on behalf of EDGINERS does not cause EDGINERS to be cancelled. To cancel EDGINERS processing, you have to cancel the batch job.
- ADD and DELETE subcommands with COUNT specified. These end with return code 4, reason code 12 when cancelled.

Additional operator controls are provided to enable simpler management of queued and active tasks running in the DFSMSrmm address space. These controls allow long running inventory management tasks to be stopped, interrupted, and then restarted. This enables better management either by system automation or by the operator because of operational priorities.

In previous releases, if a task or address space ended while waiting for DFSMSrmm processing to complete, the processing would continue and only when it was completed would DFSMSrmm check and discover that the requester had ended. Figure 8-1 shows the previous DFSMSrmm task management process. When a task or address space ends while it is waiting for DFSMSrmm processing to be complete, the control blocks identifying the requests are updated to reflect that the requester task has ended. It is only when the subsystem request completes that DFSMSrmm checks if the requester is still waiting, and only if still waiting is the requester notified.

![Figure 8-1 DFSMSrmm previous task management process](image-url)
8.1.1 New support in z/OS V1.9

In z/OS V1.9, DFSMSrmm long running local requests now check on the requester’s status, and if the requester has ended, the current processing is interrupted and ended early. For example, if a batch inventory management job is cancelled, DFSMSrmm is notified and inventory management ends early. Also, if a TSO user uses an attention (ATTN), DFSMSrmm is notified and any long-running command might be interrupted by the DFSMSrmm changed task management process.

8.1.2 Long running tasks

Long running local tasks are DFSMSrmm subsystem requests which last long enough to be included in the DFSMSrmm QUERY ACTIVE command display and the task token to be obtained and used. If the requester is inventory management, the results of the partial processing will be available in the message file.

The following long running tasks support interruption:

- **Inventory management:**
  - VRSEL Running Vital Record Processing
  - DSTORE Running Storage Location Management Processing
  - EXPROC Running Expiration Processing
  - RPTEXT Creating an extract data set
  - CATSYNCH Running DFSMSrmm Catalog Synchronization

- **RMM TSO subcommands:**
  - ADD adding new information into the CDS
  - DELETE deleting information from the CDS
  - SEARCH retrieving information from the CDS

- **EDGINERS when building lists of volumes to process.**

They all run within the DFRMM address space and check at key points in processing whether they should hold, release, or end processing.

**Note:** Task management applies only to long running local tasks.

8.1.3 Operator commands

In z/OS V1.9, DFSMSrmm operator commands also allow long running requests to be held, cancelled, and released, and allow new requests to be held and released. The STOP command processing is changed to prevent DFSMSrmm from stopping if inventory management is running. The operator must now request that inventory management is to end, in order for a STOP and MODIFY command to process immediately.

Additional operator commands provide ways to manage the running local subsystem requests which can be displayed by QUERY ACTIVE operator commands. When CANCEL, HOLD, or RELEASE operator commands are used the command is accepted and the appropriate local tasks marked with the operators request, unless the task is already processing the same operator command. When ALL or ACTIVE is used, only those local tasks which meet the processing criteria are affected.

The QUERY ACTIVE command has been extended to display the subsystem function name rather than by number, and the current task management status as shown in Figure 8-2.
Figure 8-2 DFSMSrmm QUERY ACTIVE command and result

Note: Refer to message EDG1113I for a list of functions and their meaning.

EDG1113I function:
   system requestor_type=requestor_name time
ten_token_value s ip_status

Explanation: DFSMSrmm issues this message as part of a multiline display in response to the operator MODIFY command with QUERY ACTIVE. In the message text, function identifies the requested DFSMSrmm function. The value of function can be one of the following possibilities:

<table>
<thead>
<tr>
<th>Function</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPEN</td>
<td>Tape OPEN processing</td>
</tr>
<tr>
<td>CLOSE</td>
<td>Tape CLOSE processing</td>
</tr>
<tr>
<td>LCSUX</td>
<td>EDGLCSUX OAM installation exit processing</td>
</tr>
<tr>
<td>CAT</td>
<td>Catalog processing</td>
</tr>
<tr>
<td>ACS</td>
<td>Pre-ACS processing</td>
</tr>
<tr>
<td>WTO</td>
<td>Special WTO/WTOR intercept processing</td>
</tr>
<tr>
<td>ADD</td>
<td>ADD subcommand processing</td>
</tr>
<tr>
<td>DEL</td>
<td>DELETE subcommand processing</td>
</tr>
<tr>
<td>CHG</td>
<td>CHANGE subcommand processing</td>
</tr>
<tr>
<td>LIST</td>
<td>LIST subcommand processing</td>
</tr>
<tr>
<td>SR</td>
<td>SEARCH subcommand processing</td>
</tr>
<tr>
<td>LC</td>
<td>LISTCONTROL subcommand processing</td>
</tr>
<tr>
<td>API</td>
<td>API subcommand parsing</td>
</tr>
<tr>
<td>DFHSM</td>
<td>EDGDFHSM/EDGTVEST processing to support releasing of tapes</td>
</tr>
<tr>
<td>INERS</td>
<td>EDGINERS processing</td>
</tr>
<tr>
<td>VRS</td>
<td>ADD, DELETE, UPDATE VRS subcommand processing</td>
</tr>
<tr>
<td>LS</td>
<td>LISTVRS subcommand processing</td>
</tr>
<tr>
<td>HSKP</td>
<td>Inventory management processing</td>
</tr>
<tr>
<td>BKUP</td>
<td>CDS Backup</td>
</tr>
<tr>
<td>C/S</td>
<td>Client/Server processing on a server system</td>
</tr>
</tbody>
</table>
Chapter 8. DFSMSrmm V1.9 enhancements

Any other values are internal to DFSMSrmm:

- **system** Identifies the DFSMSrmm SYSID value defined in the EDGRMMxx PARMLIB member of the originating system. A SYSID is only shown when the request originates on a client system. Local system tasks are listed without a system ID. You can use the HOLD, RELEASE, and CANCEL operator commands only for local system tasks.

- **requestor_type** Identifies the requestor. The values can be one of these:
  - JOB - the requestor is a batch job.
  - STC - the requestor is a started task.
  - TSU - the requestor is a time sharing

- **user.requestor_name** Identifies the requestor by name.

- **time** Lists the time that the request was started in hh:mm:ss. For client requests, this is the request start time from the client.

- **token_value** Uniquely identifies the request. The token is 8 alphanumeric characters.

- **s** The s column indicates special processing information for the DFSMSrmm task. It can be one of the following values:
  - blank No special information is available.
  - H The task is subject to HOLD processing.
  - C The task is subject to CANCEL processing.
  - + This task holds the RESERVE on the DFSMSrmm CDS.

- **ip_status** Identifies the status of IP processing. The value consists of three parts:
  - The status code, which can be Co (Connect), Wr (Write), Re (Read), or Cl (Close).
  - A < or > symbol, where < indicates that the request has started, and > indicates that the request has been completed.
  - The time that indicates when the ip_status last changed.

**System action:** Command processing continues.

**Operator response:** None.

**System programmer response:** None.

**Source:** DFSMSrmm

**Detecting Module:** EDGMCMD

**Routing Code:** 1,3

**Descriptor Code:** 5

### 8.1.4 New operator commands

The following new operator commands have been added, as shown in Example 8-1.

**Example 8-1 New DFSMSrmm operator commands**

```
F DFRMM,ABEND(TaskToken)
   
   |,CANCEL(TaskToken)
   |   |,HSKP
   |   |,ACTIVE
   |,CMD=command
   |,HOLD(TaskToken)
   |   |,HSKP
   |   |,ALL
```
Where:

The operands are as follows:

DFRMM
DFRMM is the default procedure name. Your installation might have defined a name other than DFRMM. See \textit{z/OS DFSMSrmm Implementation and Customization Guide} for more information about defining the started procedure name.

,CANCEL (token/HSKP/ACTIVE)
The CANCEL command option allows the operator to interrupt a long running local task. HOLD is used to interrupt a long running local task and cause the task to wait until you are ready to continue. RELEASE is used to resume processing after a task has been held.

,HOLD (token/HSKP/ACTIVE/NEW/ALL)
DFSMSrmm will find the first available task running Inventory Management and processes the CANCEL/HOLD/RELEASE. You can easily see the first available HSKP task, if any, in the results of the \texttt{QUERY ACTIVE} command.

,RELEASE (token/HSKP/ACTIVE/NEW/ALL)
A CANCEL/HOLD/RELEASE(ALL/NEW/ACTIVE) allows DFSMSrmm to inform all local and server tasks to process the CANCEL/HOLD/RELEASE.

- Using ‘HOLD(ALL)’ you can interrupt all local long running active RMM subsystem request processing and prevent new requests from starting. Using ‘RELEASE(ALL)’ you can resume all active RMM subsystem request processing and enable new requests to start.
- Using HOLD(NEW)’ you can prevent DFSMSrmm from processing any new, local subsystem requests. Using ‘RELEASE(NEW)’ you can allow DFSMSrmm to processing any new, local subsystem requests. Using ‘ACTIVE’ you can affect only the long running currently active local subsystem requests.
- You should be aware that while you have tasks in HOLD, the requester is also in a WAIT and might impact other processing in the system. When you HOLD(ALL), you can release tasks individually via RELEASE(token/HSKP/ACTIVE), but to enable new requests to be processed, you must use RELEASE(ALL/NEW).
After issuing the commands shown in Figure 8-3...

1. F DFRMM,HOLD(HSKP)
2. F DFRMM,RELEASE(HSKP)
3. F DFRMM,CANCEL(HSKP)

Figure 8-3  DFSMSrmm command sequence

You then receive the DFSMSrmm messages shown in Figure 8-4.

EDG2320I PROCESSING HELD BY OPERATOR COMMAND AT 06:43:55
EDG2318I PROCESSING RELEASED BY OPERATOR COMMAND AT 06:44:04
EDG2319I PROCESSING CANCELLED BY OPERATOR COMMAND AT 06:44:12
EDG2303E DFSMSrmm INVENTORY MANAGEMENT TASK ABEND U2223
EDG6901I UTILITY EDGHSKP COMPLETED WITH RETURN CODE 12

Figure 8-4  DFSMSrmm messages

8.1.5 Restarting the DFSMSrmm subsystem

The DFSMSrmm subsystem temporarily stops and re-initializes itself with the new options. Before stopping, DFSMSrmm completes any requests that it is processing. New and queued requests are not processed until re-initialization is completed. The operator response to restarting DFSMSrmm is shown in Figure 8-5.

F DFRMM,M=xx

Figure 8-5  Restarting the DFSMSrmm subsystem

8.1.6 Stopping the DFSMSrmm subsystem

Before you can shut down the DFSMSrmm subsystem, you must wait until all current requests are completed and any outstanding requests are flushed from the request queues. Also, DFSMSrmm cannot stop if inventory management is already running.

First, issue the QUERY ACTIVE command to determine the task that is actually preventing the stopping of DFSMSrmm. You can use this command to display the status of the tasks as shown in Figure 8-6.

F DFRMM,QUERY ACTIVE
or
F DFRMM,Q A

Figure 8-6  Display currently running local tasks

If any requests are subject to HOLD processing, you must RELEASE or CANCEL them in order for DFSMSrmm to STOP. If you want to end long running tasks in order to STOP DFSMSrmm, issue the CANCEL command as shown in Figure 8-7.
You must decide whether to cancel the tasks that have been HELD or to release them. To allow the existing tasks to complete while preventing new tasks starting, issue the HOLD command as shown in Figure 8-8.

Then to release the tasks that are HELD, issue the RELEASE command as shown in Figure 8-9.

You can also release a single task that is HELD, issue the RELEASE command as shown in Figure 8-10.

If you decide to cancel the tasks instead, issue the CANCEL command as shown in Figure 8-11.

In either case, you can now stop the DFSMSrmm subsystem task by issuing the STOP command as shown in Figure 8-12.

DFSMSrmm will not successfully shut down if another address space is using its resources. For example, there might be a DFSMSrmm WTOR outstanding for a batch job. If DFSMSrmm shutdown is delayed, DFSMSrmm issues a message to inform you of the delay.

**Note:** DFSMSrmm cannot stop:
- If inventory management is already running.
- If any long running tasks are in held.
- If new tasks are in HOLD processing.
8.1.7 QUIESCE the DFSMSrmm subsystem

To quiesce the DFSMSrmm subsystem, use the command shown in Figure 8-13:

```
MODIFY DFRMM,QUIESCE
or
F DFRMM,QUIESCE
```

*Figure 8-13  Quiesce DFSMSrmm subsystem*

DFSMSrmm completes any requests being processed and then stops all activity. Queued requests are not processed until you issue a command to take DFSMSrmm out of the quiesced state and re-initialization is completed. If you stop DFSMSrmm from the quiesced state and any requests are outstanding, message EDG1107D prompts you with your choices of action.

Take the following operator response as shown in Figure 8-14.

1. Use F DFRMM,QUERY ACTIVE command.
2. See EDG1113I message for the tasks that currently are in held (include HSKP).
3. Look for EDG1122I and EDG1123I messages to see which type of request is held.
4. Use F DFRMM,RELEASE(TaskToken/ALL/HSKP/ACTIVE) to resume processing.
5. Use F DFRMM,CANCEL(TaskToken/ALL/HSKP/ACTIVE) to end processing.

*Figure 8-14  DFSMSrmm operator response*

**Note:** DFSMSrmm cannot be quiesced if any long running tasks are in HELD status.

**EDG1107D**

REQUESTS WAIT TO BE PROCESSED - REPLY “STOP”, “QUIESCE”, “RESTART”, OR “M=xx”

**Explanation:** The STOP DFSMSrmm command has been entered, but DFSMSrmm is already quiesced. Either DFSMSrmm manual recovery is in progress, or the QUIESCE DFSMSrmm command was entered previously. Message EDG1105I or EDG1106I precedes this message. If the preceding message is EDG1106I, the requests that wait to be processed include one or more requests to update the DFSMSrmm control data set with data set catalog status.

**System action:** STOP command processing depends on the reply to this message.

<table>
<thead>
<tr>
<th>Reply</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>STOP</td>
<td>Stop DFSMSrmm and fail the waiting requests.</td>
</tr>
<tr>
<td>QUIESCE</td>
<td>Return to the quiesce state.</td>
</tr>
<tr>
<td>RESTART</td>
<td>Restart DFSMSrmm using the current parmlib member.</td>
</tr>
<tr>
<td>M=xx</td>
<td>Restart DFSMSrmm using a specified PARMLIB member, where xx is the PARMLIB member suffix. When you reply with either “RESTART” or “M=xx”, DFSMSrmm restarts and attempts to process the waiting requests.</td>
</tr>
</tbody>
</table>

**Operator response:** Reply to message EDG1107D.
System programmer response:
Determines how to reply to this message. If you choose to stop DFSMSrmm and not process the requests, you must re-synchronize the DFSMSrmm control data set with the system catalogs before running inventory management.

Source: DFSMSrmm
Detecting Module: EDGMCMD
Routing Code: 1,3
Descriptor Code: 2

EDG1108E cmd REJECTED WHILE TASKS ARE HELD
Explanation: You cannot issue STOP or MODIFY command while one or more tasks are held or while new requests are held. In the message text:

cmd
Either STOP or MODIFY operator command.

System action: DFSMSrmm fails the command.
Operator response: Do the following steps:
1. Use the F DFRMM,QUERY ACTIVE command to display the tasks that are held.
2. See messages EDG1122I and EDG1123I to see which type of requests are held.
3. See message EDG1113I to see which specific requests are held.
4. Use the F DFRMM,RELEASE operator command with ALL or NEW or a token to allow held requests to continue, or use the F DFRMM,CANCEL operator command with ACTIVE or a token to interrupt running tasks.

System programmer response: None.
Source: DFSMSrmm
Detecting Module: EDGMCMD
Routing Code: 1,3
Descriptor Code: 3,5

EDG1122I HELD local_number HELD server_number
Explanation: DFSMSrmm issues this message as part of a multi-line display in response to the operator MODIFY command with QUERY ACTIVE.

In the message text:
local_number This is the number of local tasks subject to HOLD processing.
server_number This is the number of server tasks subject to HOLD processing.

System action: DFSMSrmm processing continues.
Operator response: None.
System programmer response: None.
Source: DFSMSrmm
EDG1123I NEW REQUESTS ARE HELD

Explanation: DFSMSrmm issues this message as part of a multi-line display in response to the operator MODIFY command with QUERY ACTIVE.

System action: DFSMSrmm processing continues.

Operator response: None.

System programmer response: None.

Source: DFSMSrmm

Detecting Module: EDGMCMD
Routing Code: 1,3
Descriptor Code: 5

8.2 Multitasking of utilities

DFSMSrmm utilities, EDGUTIL and EDGHSKP, interaction with system-managed volumes in an IBM system-managed library is improved through multiple changes that should, especially in larger VTS installations, result in shorter elapsed time and more flexibility.

In this section, we discuss the following topics:

- EDGHSKP EXPROC
- EDGUTIL
- Using EDGSPLCS

8.2.1 EDGHSKP EXPROC

In z/OS V1.9, DFSMSrmm scratching of system-managed volumes can be handled either by writing control statements to an output file to be processed after EXPROC has completed or synchronously as is done today. The trigger is the new processing parameter that can be specified along with the selection options.

When you specify the EXPROC execution parameter, an optional SYSIN file allows you to select which subset of the available locations, and volume entries is to be processed during expiration. By default, all volumes in all eligible locations are processed.
Figure 8-15 shows an example of using EXPROC.

```
//NAID00CN JOB (POK,999),MSGCLASS=T,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=SC63
//HSKP EXEC PGM=EDGHSKP,
//       PARM='EXPROC'
//MESSAGE DD DISP=SHR,DSN=NAID00.HSKP.MESSAGES
//EDGSPLCS DD DISP=SHR,DSN=NAID00.SPLCS.DATA
//SYSIN DD *
EXPROC VOLUMES(DV0*,T*,MARY01) EDGSPLCS(YES)
```

Figure 8-15   EDGHSKP EXPROC JCL

Where:

The operands are as follows:

**EXPROC command**  The SYSIN EXPROC command does the following tasks:

- Selects the volumes to be processed.
- Request asynchronous return to scratch of system-managed volumes.

**EDGSPLCS DD**  Is written during EDGHSKP EXPROC if EDGSPLCS(YES) specified on the SYSIN statement. Instead of scratching the volumes, housekeeping creates statements to be used with the EDGSPLCS Utility.

Example 8-2 shows the format of the EDGHSKP EXPROC SYSIN parameters

*Example 8-2   EDGHSKP EXPROC - SYSIN parameter syntax*

```
|———EXPROC————| exproc_parameters |———|———|———|
|               |                   | —NO—|     | —EDGSPLCS(—YES——)——|

|———|———|———|
|     |     |     |
|     |     |     |
|     |     |     |
|     |     |     |
|     |     |     |
|     |     |     |
|     |     |     |
|     |     |     |
|     |     |     |
|     |     |     |
|     |     |     |

|———|———|———|———|———|
|—LOCATIONS(—location_name———)——|     |—VOLUMES(—volser————)——|

|———|———|
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |

|———|———|
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
|     |     |
```

Use this command one time only to specify the selection criteria for EXPROC processing. The selection criteria you specify selects volumes for processing by EXPROC which includes the following aspects of expiration processing:

- Releasing expired volumes.
- Setting and processing individual release actions.
- Returning volumes to scratch status.
If you have specified the EXPROC job step parameter but neither VRSEL nor DSTORE, Global confirmed actions and moves are not processed.

The operands are all optional and are as follows:

**LOCATIONS(location_name)**

Specifies a subset of the available volumes based on the volume's current location for processing. The location names specified should be a storage location defined with LOCDEF, an available system-managed library on the current system, or shelf. Volumes to be released can be in any location, but volumes can only be returned to scratch while not in transit, and resident in a system-managed library, shelf, or their home location. A location_name is one-to-eight characters and can be a location name mask. Each location_name can be specified in one of these ways:

- Specify a specific location using one-to-eight character names.
- Specify all locations using a single asterisk (*).
- Specify all locations that begin or end with specific characters, such as ATL* or *DR, or multiple locations by using * within a location name.
- Use % (percent sign) in the location name to replace a single character. You can specify up to eight % in a location name mask.

DFSMSrmm does not validate the specified location names against the DFSMSrmm LOCDEF entries or the names of the SMS libraries. You can specify a list of up to eight values.

**VOLUMES(volser)**

Specifies a list of volumes to be processed. You can specify the volumes as fully qualified or as a volser prefix ending in *. A fully qualified volume is one-to-six alphanumeric, national or special characters, but the first character must not be blank. Quotes are required for special characters. Any value ending in *, even if it is enclosed in quotes, is considered to be a volser prefix. You can specify a list of up to eight values.

**VOLUMERANGES(startvolser:endvolser)**

Specifies a subset of volumes based on the starting and ending volser values to be processed. The volser values must be one-to-six alphanumeric, national, or special characters, but the first character must not be blank. Quotes are required for each value regardless of the use of special characters. The end of range must not be lower than the start of the range. You can specify a list of up to eight values.

**EDGSPLCS(YES|NO)**

Use this operand to specify whether DFSMSrmm should return system-managed volumes to scratch during EXPROC processing or to write volume scratch statements in the EDGSPLCS file to be used with the EDGSPLCS utility. When you specify EDGSPLCS(YES), EDGHSKP opens the EDGSPLCS file. If the DD name is missing, EDGHSKP issues message EDG6101E and ends with return code 12. A scratch (S) statement is written to the EDGSPLCS file for each volume for which DFSMSrmm processing would normally have issued a CUA request to OAM to return the volume to scratch. When you specify EDGSPLCS(NO), each system-managed volume to be returned to scratch is returned to scratch during EXPROC using a CUA request to OAM.
Figure 8-16 and Figure 8-17 show an example of the statements in the EDGSPLCS file and MESSAGE file. These statements were generated when we ran the EXPROC job in Figure 8-15 on page 60. Refer to Table 8-1 on page 70 to see the syntax of the EDGSPLCS INDD input record format.

Figure 8-16   EDGSPLCS file

| S | TS4285 | LIB1 |
| S | DV0042 | LIB2 |
| S | DV0053 | LIB2 |
| S | DV0061 | LIB2 |

A scratch “S” statement is written to the EDGSPLCS file for each volume for which DFSMSrmm processing would normally have issued a CUA (change use attribute) request to OAM to return the volume to scratch when EDGSPLCS(NO) is specified.

Figure 8-17   MESSAGE file

EDG6001I INVENTORY MANAGEMENT STARTING ON 2007/135 AT 09:20:44
PARAMETERS IN USE ARE DATEFORM(J),EXPROC
EDG6013I THE SYSIN OPTIONS CURRENTLY IN USE ARE
   EXPROC
   EDGSPLCS(YES)
   VOLUMES(DV0*,T*,MARY01)
EDG2309I THE PARMLIB OPTIONS CURRENTLY IN USE ARE
   ........
EDG2308I CHANGES HAVE BEEN MADE TO VRS POLICIES SINCE THE PREVIOUS INVEN
EDG2420I PHYSICAL VOLUMES READ = 2151
EDG2420I LOGICAL VOLUMES READ = 15
EDG2424I TOTAL VOLUMES, THIS RUN, SET PENDING RELEASE = 0
EDG2425I TOTAL VOLUMES RETURNED TO SCRATCH = 1
EDG2426I TOTAL NUMBER OF SCRATCH RECORDS WRITTEN = 4
EDG2429I MAIN INVENTORY MANAGEMENT UPDATES HAVE COMPLETED SUCCESSFULLY
EDG2307I INVENTORY MANAGEMENT TASK EXPROC COMPLETED SUCCESSFULLY
EDG6901I UTILITY EDGHSKP COMPLETED WITH RETURN CODE 0

Note: If the run of EDGHSKP includes parameters other than EXPROC:

> DFSMSrmm processing will process all volumes, but only the selected volumes will be subject to EXPROC.

> Global confirmed actions and moves are completed if DSTORE or VRSEL are specified.

8.2.2 EDGUTIL

In z/OS V1.9, DFSMSrmm utility EDGUTIL is no longer considered to be part of inventory management processing. EDGUTIL can run at any time, even in parallel with other EDGUTIL instances. The requirement for MEND to be run against an unused CDS is unchanged.

> A subset of volumes can be selected to be processed on a run of EDGUTIL. An optional SYSIN file allows you to select the subset from the available locations, and volume entries during verification of volumes.
Verification processing of volumes includes:

- VERIFY/MEND(SMSTAPE/VOLCAT)
- VERIFY or MEND
- VERIFY/MEND(ALL)
- VERIFY/MEND(VOL)

**Processing subsets of volumes**

By default, all volumes are verified. The SYSIN commands in Example 8-3 can be used to select the subset of volumes.

**Example 8-3   EDGUTIL SYSIN parameter syntax**

```
|——INCLUDE——| include/exclude_parameters |———LOCATIONS(—location_name——)——| ———VOLUMES(—volser——)—— |——VOLUMERANGES(—startvolser:end volser——)—— |
|——EXCLUDE—| | | |
```

Where:

The operands are as follows:

**EXCLUDE**

Specifies the exclusion criteria for EDGUTIL processing. You can specify this command one time only. You can specify INCLUDE and EXCLUDE commands in any order.

You can specify one or more of these optional operands:

**LOCATIONS(location_name)**

Specifies volumes to be excluded based on the volume’s current location. For 3-way audit and VOLCAT processing, specify the system-managed library location names. For VOL processing, any system-managed library, storage location name known to DFSMSrmm, or SHELF can be specified. A location_name is one-to-eight characters and can be a location name mask. Each location_name can be specified in one of these ways:

- Specify a specific location using one-to-eight character names.
- Specify all locations using a single asterisk (*).
- Specify all locations that begin or end with specific characters, such as ATL* or "DR, or multiple locations by using * within a location name.
Use % (percent sign) in the location name to replace a single character. You can specify up to eight % in a location name mask.

DFSMSrmm does not validate the specified location names against the DFSMSrmm LOCDEF entries or the names of the SMS libraries.

When validation of a location name fails, the EDGUTIL utility stops processing and ends with the return code of 12. Any location_names specified are used to exclude volumes based on the TCDB volume record library name and the DFSMSrmm volume record current location. You can specify a list of values.

**VOLUMES(volser)**

Specifies a list of volumes to be excluded. You can specify the volumes as fully qualified or as a volser prefix ending in *. A fully qualified volume is one-to-six alphanumeric, national or special characters, but the first character must not be blank. Quotes are required for special characters. Any value ending in *, even if it is enclosed in quotes, is considered to be a volser prefix. You can specify a list of values.

**VOLUMERANGES(startvolser:endvolser)**

Specifies a subset of volumes based on the starting and ending volseres to be excluded. The volseres must be one-to-six alphanumeric, national, or special characters, but the first character must not be blank. Quotes are required for each value regardless of the use of special characters. The end of range must not be lower than the start of the range. You can specify a list of values.

The default is that no volumes are excluded.

**INCLUDE**

Specifies the inclusion criteria for EDGUTIL processing. You can specify this command one time only. You can specify INCLUDE and EXCLUDE commands in any order.

You can specify one or more of these optional operands:

**LOCATIONS(location_name)**

Specifies a subset of the available volumes based on the volume's current location for processing. For 3-way audit and VOLCAT processing, specify the system-managed library location names. For VOL processing, any system-managed library, storage location name known to DFSMSrmm, or SHELF can be specified. A location_name is one-to-eight characters and can be a location name mask.
Each location_name can be specified in one of these ways:

- Specify a specific location using one-to-eight character names.
- Specify all locations using a single asterisk (*).
- Specify all locations that begin or end with specific characters, such as ATL* or "DR, or multiple locations by using * within a location name.
- Use % (percent sign) in the location name to replace a single character. You can specify up to eight % in a location name mask.

DFSMSrmm does not validate the specified location names against the DFSMSrmm LOCDEF entries or the names of the SMS libraries.

When validation of a location name fails, the EDGUTIL utility stops processing and ends with the return code of 12.

Any location_names specified are used to select volumes based on the TCDB volume record library name and the DFSMSrmm volume record current location. You can specify a list of values.

**VOLUMES(volser)**

Specifies a list of volumes to be processed. You can specify the volumes as fully qualified or as a volser prefix ending in *. A fully qualified volume is one-to-six alphanumeric, national or special characters, but the first character must not be blank. Quotes are required for special characters. Any value ending in *, even if it is enclosed in quotes, is considered to be a volser prefix. You can specify a list of values.

**VOLUMERANGES(startvolser:endvolser)**

Specifies a subset of volumes based on the starting and ending volsers to be processed. The volsers must be one-to-six alphanumeric, national, or special characters, but the first character must not be blank. Quotes are required for each value regardless of the use of special characters. The end of range must not be lower than the start of the range. You can specify a list of values.

Processing stacked volumes

In z/OS V1.9, when a subset of volumes is processed, VERIFY/MEND(VOL) does not consider stacked volume processing. For stacked volumes, VERIFY/MEND(VOL) checks the consistency of exported logical volumes with stacked volumes.

Deferred correcting TCDB and LM database

Correction of the tape configuration database (TCDB) and the library management database (LM) based on the RMM CDS information was previously done by using EDGUTIL with PARM='MEND(SMSTAPE)’. The correction was done synchronously to the verify processing, resulting in a long elapsed time.

In z/OS V1.9, by using EDGUTIL with PARM="VERIFY(SMSTAPE)’ and EDGSPLCS DD and the EDGSPLCS utility:

- Control statements are generated in the EDGSPLCS output file by EDGUTIL.
- Volumes can be corrected after EDGUTIL processing is completed by using these statements as an input for the EDGSPLCS utility.
Figure 8-18 is an example of using the deferred TCDB and LM database correction method.

```//NAIDOOMN JOB (POK,999),MSGCLASS=T,NOTIFY=&SYSUID 
/*JOBPARM SYSAFF=SC63 
//HSKP EXEC PGM=EDGUTIL,PARM='VERIFY(SMSTAPE)' 
//EDGSPLCS DD DISP=SHR,DSN=NAIDOO.SPLCS.DATA 
//SYSPRINT DD SYSOUT=* 
//SYSIN DD * 
INCLUDE VOLUMES(M*,V*,T*) 
/*
```

**Figure 8-18**  EDGUTIL VERIFY(SMSTAPE)

**Note:** The EDGSPLCS DD:
- Is written during VERIFY(SMSTAPE).
- If specified, it causes EDGUTIL to create statements to be used with the EDGSPLCS utility.

Figure 8-19 shows the output from SYSPRINT DD.

```INCLUDE VOLUMES(M*,V*,T*)
EDG6433I STARTING VERIFICATION OF VOLUME RECORDS
EDG6824I VOLUME TST020 IS IN VOLUME CATALOG ERROR STATUS 0004 SECURITY CONFLICT
EDG6846I VOLUME CATALOG UPDATE REQUIRED - STATEMENT WRITTEN TO EDGSPLCS FILE FOR VOLUME TST020
EDG6824I VOLUME TST021 IS IN VOLUME CATALOG ERROR STATUS 0004 SECURITY CONFLICT
EDG6846I VOLUME CATALOG UPDATE REQUIRED - STATEMENT WRITTEN TO EDGSPLCS FILE FOR VOLUME TST021
EDG6418W CONTROL DATA SET VERIFY COMPLETED WITH ERRORS
EDG6901I UTILITY EDGUTIL COMPLETED WITH RETURN CODE 4
```

**Figure 8-19**  SYSPRINT DD output

Where:

**EDG6846I**  VOLUME CATALOG UPDATE REQUIRED - STATEMENT WRITTEN TO EDGSPLCS FILE FOR VOLUME *volser*

**Explanation:** You are running the DFSMSrmm EDGUTIL utility to VERIFY SMS information for volumes. DFSMSrmm found inconsistencies between the DFSMSrmm information for the volume and the volume catalog and Library Manager database. This message indicates that the error can be corrected. A control statement has been written to the data set represented by the EDGSPLCS DD statement.

In the message text:
- **volser** This is the volume serial number of the volume to be corrected.

**System action:** DFSMSrmm utility continues. A control statement has been written to the data set specified by the EDGSPLCS DD statement.

**Operator response:** None.
Application Programmer Response:
Use the EDGSPLCS utility to process the statements produced. Review the contents of the EDGSPLCS file. If you want to make those corrections, pass the file to the EDGSPLCS INDD file.

Source: DFSMSrmm
Detecting Module: EDGUTIL

Figure 8-20 shows the corresponding EDGSPLCS generated statements.

| S | TST020 | LIB1 |
| P | TST021 | LIB1 |

Figure 8-20   EDGSPLCS generated statements to update the IBM ATL or IBM VTS

3-way audit
For 3-way audit with system-managed libraries (for example, IBM VTSs and IBM ATLs), VERIFY(SMSTAPE), MEND, and MEND(SMSTAPE) processing exploits the use of a host library interface to return multiple volumes in a single request. In addition, DFSMSrmm allows the selection of libraries and subsets of volumes. This processing reduces the EDGUTIL elapsed time. EDGUTIL constructs a series of requests for the libraries containing the volumes to be verified. The information is retrieved as required and is processed together with entries from the TCDB and volume information from the DFSMSrmm control data set.

When a mismatch is detected between the TCDB, DFSMSrmm control data set, and the library manager data, DFSMSrmm uses the CBRXLCS QVR request so that any timing related change can be detected. Processing of all volumes, regardless of function, is subject to the subsetting through location and volume selection (Figure 8-21).

Figure 8-21   EDGUTIL 3-way audit
Using the diagram in Figure 8-21 on page 67, which explains how 3-way audit works:

1. First check these events:
   - The TCDB volume entries are retrieved from all connected volume catalogs, sorted and the volume subset extracted via the INCLUDE/EXCLUDE statements, and written into the VCINOUT file.
   - The CDS and the VCINOUT are read sequential in parallel, and the data for each volume are compared.
   - If a volume is recorded in VCINOUT (that is in the TCDB) and not in the CDS, or a volume is recorded in the CDS but not in the VCINOUT (that is, it is not in the TCDB), or a mismatch is found between the data in the CDS and TCDB the processing continues at “Mismatch Found:”
   - If the volume is not in the buffer, the next set of library manager records are read at once and saved in buffer. There is 1 buffer for each library. The volume information from the CDS and TCDB are compared to the ones in the buffer.
   - If a volume is recorded in the CDS and TCDB but not in the LM data base, or a mismatch is found between the data in the CDS, TCDB and LM the processing continues at “Mismatch Found:”.

2. Mismatch Found:
   - Because the data in VCINOUT (from TCDB) and in the buffered data from LM could be out of date, the LM data base and TCDB are read once more by using the CBRXLCS QVR macro. These data are compared once more with the volume information from the CDS.
   - If the volume is defined to TCDB but not defined to RMM, then it is reported to the MESSAGE file.
   - If the volume is defined to RMM but not defined to TCDB, then it is reported to the MESSAGE file, and if MEND(SMSTAPE) and the Library is an MTL then manual cartridge entry is done.
   - If the volume is defined to RMM and to TCDB but not defined to LM, then it is reported to the MESSAGE file.
   - If a mismatch found then it is reported to the MESSAGE file and if VERIFY(SMSTAPE) and EDGSPLCS DD specified then control statements are written to use with the EDGSPLCS Utility.
   - If MEND, then RMM volume records are fixed using the TCDB and LM data.
   - If MEND(SMSTAPE), TCDB and LM volume records are fixed using the RMM data.
   - If MEND(VOLCAT), then the processing is similar, but without LM processing.

8.2.3 Using EDGSPLCS

You can use the EDGSPLCS utility to issue supported commands to OAM for system-managed volumes. DFSMSrmm builds the input commands for this utility automatically during EDGUTIL VERIFY(SMSTAPE) processing and EDGHSKP EXPROC processing when you request them.

You can run multiple copies of EDGSPLCS. Using different parameters, EDGSPLCS can be processing in parallel for multiple libraries, but this utility does not ensure that each parameter is different from any other currently running.

You need RACF ALTER authority to the relevant volume catalog in order to use the EDGSPLCS utility to update the TCDB. For example, if you use just one volume catalog and
use the default volume catalog prefix, you need ALTER access to
SYS1.VOLCAT.VGENERAL.

Running EDGSPLCS
Example 7-1 shows the sample JCL for execution of EDGSPLCS. The SYSIN file in the JCL
is partly created from the running programs described in the previous chapter.

Example 8-4 shows the execution parameters for EDGSPLCS.

Example 8-4  EDGSPLCS EXEC parameters

|——ACTION(ALL)———|       |——LOCATION(-library_name-)——|
|——,——|       |——|——|

|——,——|       |——ACTION(-S———)——|
|——P——|       |——I——|       |——X——|       |——M——|       |——E——|
Where:

**ACTION(ALL/S/P/I/X/M/E)**

Specifies that only the specified requests in the input file are processed. You can optionally provide the name of a library to restrict the processing to only those requests. You can specify one or more of the possible actions.

- **S** set to Scratch status.
- **P** set to Private status.
- **I** Import volume.
- **X** eXport volume.
- **M** Manual cartridge entry.
- **E** Eject volume.

If you do not specify the ACTION parameter, the default value is **ALL**.

**LOCATION(library_name)**

Specifies the name the system-managed library for which the EDGSPLCS utility will process commands during this run. By default, all locations are considered. However, you can select a subset based on the library name using this parameter.

**INDD input file**

Figure 8-24 displays an existing file of LRECL 80 containing control statements that direct the processing that EDGSPLCS utility performs. The control statements in INDD have the format shown in Table 8-1.

<table>
<thead>
<tr>
<th>av</th>
<th>volser</th>
<th>options</th>
<th>library</th>
<th>message</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>4</td>
<td></td>
<td>1</td>
<td>2 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>7 6</td>
</tr>
</tbody>
</table>

Figure 8-24   EDGSPLCS input record format

Table 8-1 shows all of the control statements and operands that can be used in INDD the input file for the EDGSPLCS utility.

**Table 8-1   INDD control statements**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Explanation</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Action character</td>
<td>One of these values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S set to Scratch status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>P set to Private status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>I Import volume</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X eXport volume</td>
</tr>
<tr>
<td></td>
<td></td>
<td>M Manual Cartridge Entry</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E Eject volume</td>
</tr>
<tr>
<td>v</td>
<td>Verify request</td>
<td>One of these values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>V Verify volume is resident</td>
</tr>
<tr>
<td></td>
<td></td>
<td>blank Do not verify volume is resident</td>
</tr>
</tbody>
</table>
OUTDD output file

This is the output file that is written by the EDGSPLCS utility. It contains a copy of each of the input control statements, and each statement contains a completion message as shown in Figure 8-25. The return and reason codes that EDGSPLCS has added starting in column 36 are described in Table 8-2.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Explanation</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>volser</td>
<td>Volume Serial</td>
<td>Action specific values starting in column 11: S P Storage group name or blank followed by optional owner ID or blank. ‘Owner ID’ starts in column 19. I Cancel request. Specify C to cancel an existing import. Distributed library name to start an import in a specific library of a PtP VTS. X Cancel request. Specify C to cancel an existing export. Distributed library name to start an export in a specific library of a PtP VTS. M Library name into which the volume is to be entered. Column 20; media type of volume to be entered - for example; 5. E Eject destination. Either C (convenience) or B (high capacity).</td>
</tr>
<tr>
<td>options</td>
<td>depends on action character</td>
<td></td>
</tr>
<tr>
<td>library</td>
<td>Library name</td>
<td>Starting in column 27, this is an eight character field for you to specify the library name. This field is used by the LOCATION execution parameter and is only required if LOCATION parameter is specified.</td>
</tr>
<tr>
<td>message</td>
<td>Output area for EDGSPLCS</td>
<td>After processing, this area contains a function specific message from the EDGSPLCS utility.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>av</th>
<th>volser</th>
<th>options</th>
<th>library</th>
<th>message</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 8-25  EDGSPLCS output record format

Return codes for EDGSPLCS

EDGSPLCS issues the return codes shown in Table 8-2.

Table 8-2  EDGSPLCS return codes

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>All requested functions completed successfully. Refer to OUTDD records for individual messages from the input actions.</td>
</tr>
<tr>
<td>4</td>
<td>DFSMSrmm encountered a minor error during processing. Refer to OUTDD records for individual messages from the input actions.</td>
</tr>
<tr>
<td>8</td>
<td>All requested functions completed successfully. At least one requested action was not supported. Refer to OUTDD records for individual messages from the input actions.</td>
</tr>
<tr>
<td>12</td>
<td>DFSMSrmm encountered a severe error during processing. DFSMSrmm stops the utility.</td>
</tr>
</tbody>
</table>

Reason Codes for EDGSPLCS

EDGSPLCS issues the reason codes shown in Table 8-3.

Table 8-3  EDGSPLCS reason codes

<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SUCCESSFUL EXECUTION</td>
</tr>
<tr>
<td>2</td>
<td>OAM CONTROL BLOCKS NOT AVAILABLE</td>
</tr>
<tr>
<td>3</td>
<td>DELETED WITH MTL SOFTWARE-ONLY</td>
</tr>
<tr>
<td>4</td>
<td>VOLUME ALREADY SCRATCH</td>
</tr>
<tr>
<td>5</td>
<td>VOLUME ALREADY PRIVATE</td>
</tr>
<tr>
<td>6</td>
<td>CUA PROCESSING DISABLED</td>
</tr>
<tr>
<td>7</td>
<td>CUA PROCESSING NOT PERFORMED FOR THIS VOLUME PER INSTALLATION EXIT REQUEST</td>
</tr>
<tr>
<td>8</td>
<td>SCRATCH VOLUME THRESHOLD MESSAGE PROCESSING WAS NOT COMPLETED SUCCESSFULLY</td>
</tr>
<tr>
<td>9</td>
<td>REQUIRED TYPE PARAMETER NOT SPECIFIED</td>
</tr>
<tr>
<td>10</td>
<td>MUTUALLY EXCLUSIVE REQUIRED PARAMETERS SPECIFIED</td>
</tr>
<tr>
<td>11</td>
<td>INVALID TYPE VALUE SPECIFIED</td>
</tr>
<tr>
<td>12</td>
<td>REQUIRED FUNC PARAMETER NOT SPECIFIED</td>
</tr>
<tr>
<td>13</td>
<td>INVALID FUNC VALUE</td>
</tr>
<tr>
<td>14</td>
<td>REQUIRED USE PARAMETER NOT SPECIFIED</td>
</tr>
<tr>
<td>15</td>
<td>INVALID USE VALUE</td>
</tr>
<tr>
<td>16</td>
<td>REQUIRED VOLUME PARAMETER NOT SPECIFIED</td>
</tr>
<tr>
<td>17</td>
<td>INVALID VOLUME VALUE</td>
</tr>
<tr>
<td>18</td>
<td>REQUIRED UCBPTR NOT SPECIFIED</td>
</tr>
<tr>
<td>19</td>
<td>INVALID UCBPTR VALUE</td>
</tr>
<tr>
<td>Reason Code</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>20</td>
<td>REQUIRED VOLLIST NOT SPECIFIED</td>
</tr>
<tr>
<td>21</td>
<td>INVALID VOLUME LIST VALUE</td>
</tr>
<tr>
<td>22</td>
<td>INVALID VALUE IN STORAGE GROUP LIST HEADER</td>
</tr>
<tr>
<td>23</td>
<td>REQUIRED LIBRARY NAME NOT SPECIFIED</td>
</tr>
<tr>
<td>24</td>
<td>INVALID VOLUME LIST, MIXED MEDIA, REWRITABLE AND WORM VOLUMES</td>
</tr>
<tr>
<td>25</td>
<td>INVALID EXPIRATION DATE VALUE</td>
</tr>
<tr>
<td>26</td>
<td>LIBRARY NOT DEFINED TO STORAGE GROUP</td>
</tr>
<tr>
<td>27</td>
<td>INVALID LIBRARY NAME SPECIFIED</td>
</tr>
<tr>
<td>28</td>
<td>INVALID WRITE PROTECT STATUS VALUE</td>
</tr>
<tr>
<td>29</td>
<td>INVALID ADDRESS SPECIFIED FOR PARAMETER LIST OR MAPPING MACRO</td>
</tr>
<tr>
<td>30</td>
<td>ADDRESS NOT ON WORD BOUNDARY OR LEVEL</td>
</tr>
<tr>
<td>31</td>
<td>REQUIRED TAPE DEVICE SELECTION INFORMATION (TDSI) NOT SPECIFIED</td>
</tr>
<tr>
<td>32</td>
<td>REQUIRED LIBRARY ID OR LIBRARY NAME NOT SPECIFIED</td>
</tr>
<tr>
<td>33</td>
<td>INVALID VALUE SPECIFIED FOR LIBRARY ID</td>
</tr>
<tr>
<td>34</td>
<td>INVALID STORAGE GROUP NAME</td>
</tr>
<tr>
<td>35</td>
<td>REQUIRED MEDIA TYPE NOT SPECIFIED FOR MCE VOLUME</td>
</tr>
<tr>
<td>38</td>
<td>INVALID COMPACTION SPECIFIED IN TDSI</td>
</tr>
<tr>
<td>39</td>
<td>INVALID SPECIAL ATTRIBUTE SPECIFIED IN TDSI</td>
</tr>
<tr>
<td>40</td>
<td>INVALID COMBINATION OF TAPE DEVICE SELECTION VALUES SPECIFIED</td>
</tr>
<tr>
<td>41</td>
<td>AMBIGUOUS TDSI COMBINATION SPECIFIED</td>
</tr>
<tr>
<td>42</td>
<td>TAPE DEVICE SELECTION VALUE SPECIFIED WHERE NOT ALLOWED OR NOT APPLICABLE</td>
</tr>
<tr>
<td>43</td>
<td>INVALID POINTER TO TDSI SPECIFIED</td>
</tr>
<tr>
<td>44</td>
<td>INVALID VALUE SPECIFIED FOR DISP KEYWORD</td>
</tr>
<tr>
<td>45</td>
<td>LIBRARY NAME AS DEFINED IN VOLUME RECORD NOT FOUND IN TCDB</td>
</tr>
<tr>
<td>46</td>
<td>NO ENABLED STORAGE GROUPS</td>
</tr>
<tr>
<td>47</td>
<td>NOT ALL VOLUMES ASSOCIATED WITH THE SAME STORAGE GROUP</td>
</tr>
<tr>
<td>48</td>
<td>STORAGE GROUP STATE IS NOT CON, DISALL, OR DISNEW</td>
</tr>
<tr>
<td>49</td>
<td>NO DEVICE POOLS EXIST TO FULFILL REQUEST FOR TDSI SPECIFICATION</td>
</tr>
<tr>
<td>51</td>
<td>SPECIFIC VOLSER REQUEST FOR SCRATCH VOLUME</td>
</tr>
<tr>
<td>52</td>
<td>VOLUME(S) RESIDE OUTSIDE LIBRARY</td>
</tr>
<tr>
<td>53</td>
<td>LIBRARY FOR SPECIFIED VOLUME NOT DEFINED TO SMS CONFIGURATION</td>
</tr>
<tr>
<td>54</td>
<td>SMS STORAGE GROUP WAS NOT OF TYPE TAPE</td>
</tr>
<tr>
<td>Reason Code</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>55</td>
<td>REQUESTED DEVICE DOES NOT RESIDE IN SAME LIBRARY AS REQUESTED VOLUME</td>
</tr>
<tr>
<td>56</td>
<td>NO LIBRARIES ASSOCIATED WITH LIST OF STORAGE GROUPS OR LIBRARY IS UNKNOWN</td>
</tr>
<tr>
<td>58</td>
<td>FAILURE ACCESSING THE VOLUME RECORD IN THE CATALOG</td>
</tr>
<tr>
<td>59</td>
<td>FAILURE ACCESSING THE LIBRARY RECORD IN THE CATALOG</td>
</tr>
<tr>
<td>60</td>
<td>FAILURE ACCESSING THE SMS STORAGE GROUP CONSTRUCTS</td>
</tr>
<tr>
<td>61</td>
<td>FAILURE ACCESSING HARDWARE CONFIGURATION INFORMATION</td>
</tr>
<tr>
<td>62</td>
<td>SPECIFIED LIBRARY IS NOT DEFINED TO ACTIVE SMS CONFIGURATION</td>
</tr>
<tr>
<td>63</td>
<td>VOLUME RECORD NOT FOUND FOR REQUESTED VOLUME</td>
</tr>
<tr>
<td>64</td>
<td>LIBRARY RECORD NOT FOUND IN TCDB FOR REQUESTED LIBRARY</td>
</tr>
<tr>
<td>65</td>
<td>LIBRARY LOGICAL TYPE NOT DEFINED</td>
</tr>
<tr>
<td>66</td>
<td>NO DEVICE POOLS TO FULFILL REQUEST FOR SPECIFIED RECORDING TECHNOLOGY</td>
</tr>
<tr>
<td>67</td>
<td>NO DEVICE POOLS TO FULFILL REQUEST FOR SPECIFIED MEDIA TYPE</td>
</tr>
<tr>
<td>69</td>
<td>REQUEST FAILED BECAUSE VOLUME NOT IN LIBRARY INSTALLATION EXIT</td>
</tr>
<tr>
<td>70</td>
<td>VOLUME NOT FOUND IN LIBRARY MANAGER INVENTORY</td>
</tr>
<tr>
<td>72</td>
<td>DELETED WITH MTL SOFTWARE-ONLY</td>
</tr>
<tr>
<td>74</td>
<td>REQUEST FAILED BECAUSE VOLUME SERIAL NUMBER ALREADY EXISTS IN LIBRARY MANAGER INVENTORY</td>
</tr>
<tr>
<td>75</td>
<td>UNEXPECTED UCBSCAN ERROR ENCOUNTERED DURING PROCESSING</td>
</tr>
<tr>
<td>76</td>
<td>DELETED WITH MTL SOFTWARE-ONLY</td>
</tr>
<tr>
<td>77</td>
<td>DELETED WITH MTL SOFTWARE-ONLY</td>
</tr>
<tr>
<td>78</td>
<td>ERROR ATTEMPTING TO RETRIEVE VOLUME RECORD</td>
</tr>
<tr>
<td>79</td>
<td>ERROR ATTEMPTING TO WRITE VOLUME RECORD</td>
</tr>
<tr>
<td>80</td>
<td>ESTAE ROUTINE NOT ESTABLISHED</td>
</tr>
<tr>
<td>81</td>
<td>GETMAIN FAILED FOR DEVICE POOL NAMES LIST OR LOCAL WORKING STORAGE</td>
</tr>
<tr>
<td>82</td>
<td>ABNORMAL TERMINATION OCCURRED DURING INSTALLATION EXIT (CBRUXCUA) EXECUTION</td>
</tr>
<tr>
<td>83</td>
<td>INVALID RETURN CODE OR DATA RETURNED FROM CHANGE USE ATTRIBUTE INSTALLATION EXIT (CBRUXCUA)</td>
</tr>
<tr>
<td>84</td>
<td>ABNORMAL TERMINATION OCCURRED DURING EXECUTION</td>
</tr>
<tr>
<td>90</td>
<td>CARTRIDGE ENTRY PROCESSING HAS BEEN DISABLED</td>
</tr>
<tr>
<td>91</td>
<td>CARTRIDGE ENTRY PROCESSING HAS BEEN SUSPENDED FOLLOWING ERROR INVOKING INSTALLATION EXIT</td>
</tr>
<tr>
<td>92</td>
<td>LIBRARY FOR MCE NOT MANUAL TAPE LIBRARY</td>
</tr>
<tr>
<td>Reason Code</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>93</td>
<td>LIBRARY OFFLINE, PENDING OFFLINE, OR NOT OPERATIONAL</td>
</tr>
<tr>
<td>94</td>
<td>SPECIFIED VOLUME ALREADY RESIDES IN ANOTHER LIBRARY</td>
</tr>
<tr>
<td>95</td>
<td>INSTALLATION EXIT VETOED ENTRY OF VOLUME INTO LIBRARY</td>
</tr>
<tr>
<td>96</td>
<td>VOLUME NOT ENTERED INTO MTL BECAUSE INSTALLATION EXIT SAID TO IGNORE THE VOLUME</td>
</tr>
<tr>
<td>97</td>
<td>VOLUME OF SAME VOLSER IS KNOWN DASD VOLUME</td>
</tr>
<tr>
<td>120</td>
<td>VOLUME IS INELIGIBLE BECAUSE THE TYPE OF MEDIA DEFINED IN THE VOLUME RECORD MAY NOT BE MOUNTED ON SPECIFIED DEVICE</td>
</tr>
<tr>
<td>121</td>
<td>VOLUME IS INELIGIBLE BECAUSE THE TYPE OF MEDIA DEFINED IN THE TDSI DOES NOT MATCH MEDIA DEFINED ON THE VOLUME RECORD</td>
</tr>
<tr>
<td>122</td>
<td>VOLUME IS INELIGIBLE BECAUSE THE VOLUME RECORD REFLECTS AN ERROR STATUS</td>
</tr>
<tr>
<td>123</td>
<td>VOLUME IS INELIGIBLE BECAUSE THE SPECIFIED RECORDING TECHNOLOGY IS INCOMPATIBLE WITH THE VOLUME MEDIA TYPE OR THE SPECIFIED DRIVE TYPE</td>
</tr>
<tr>
<td>130</td>
<td>SPECIFIED VOLUME ALREADY RESIDES IN THIS TAPE LIBRARY</td>
</tr>
<tr>
<td>131</td>
<td>SCRATCH VOLUME THRESHOLD PROCESSING NOT PERFORMED BECAUSE LIBRARY WAS NOT OPERATIONAL</td>
</tr>
<tr>
<td>132</td>
<td>DELETED WITH 3590 SUPPORT</td>
</tr>
<tr>
<td>133</td>
<td>DELETED WITH 3590 SUPPORT</td>
</tr>
<tr>
<td>134</td>
<td>NO TCDB RECORD BUT VOLUME RESIDES IN SPECIFIED LIBRARY</td>
</tr>
<tr>
<td>135</td>
<td>LIBRARY MISMATCH, VOLUME RESIDES IN SPECIFIED LIBRARY</td>
</tr>
<tr>
<td>136</td>
<td>LIBRARY MISMATCH, VOLUME NOT FOUND IN SPECIFIED LIBRARY</td>
</tr>
<tr>
<td>137</td>
<td>LIBRARY MISMATCH, UNABLE TO ACCESS SPECIFIED LIBRARY</td>
</tr>
<tr>
<td>138</td>
<td>LIBRARY SCRATCH COUNT NOT UPDATED IN TCDB</td>
</tr>
<tr>
<td>139</td>
<td>LIBRARY NAME MISMATCH, SPECIFIED LIB NAME DID NOT MATCH MTL VOLUME RECORD LIB NAME</td>
</tr>
<tr>
<td>201</td>
<td>VOLUME ALREADY EJECTED</td>
</tr>
<tr>
<td>202</td>
<td>INVALID VALUE FOR EJECT OPTION</td>
</tr>
<tr>
<td>203</td>
<td>INVALID VALUE FOR BULK EJECT</td>
</tr>
<tr>
<td>204</td>
<td>INVALID TSO USERID SPECIFIED</td>
</tr>
<tr>
<td>215</td>
<td>NOT ALL VOLUMES HAVE THE SAME RECORDING TECHNOLOGY</td>
</tr>
<tr>
<td>216</td>
<td>INVALID TCDBCHK=NO SPECIFIED WITH MTL LIB NAME</td>
</tr>
<tr>
<td>217</td>
<td>AT LEAST ONE OPTIONAL KEYWORD MUST BE SPECIFIED WITH FUNCTION</td>
</tr>
<tr>
<td>218</td>
<td>REQUIRED DATATYPE NOT SPECIFIED</td>
</tr>
<tr>
<td>219</td>
<td>INVALID POLICY NAME</td>
</tr>
<tr>
<td>220</td>
<td>MUTUALLY EXCLUSIVE OPTIONAL KEYWORDS SPECIFIED</td>
</tr>
<tr>
<td>Reason Code</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>221</td>
<td>INVALID STORAGE HEADER AND/OR LENGTH</td>
</tr>
<tr>
<td>300</td>
<td>OAM ABEND DURING EJECT REQUEST</td>
</tr>
<tr>
<td>302</td>
<td>EJECT REQUEST ALREADY PENDING FOR VOLUME</td>
</tr>
<tr>
<td>303</td>
<td>UNABLE TO MAKE USER ADDRESS SPACE NON-SWAPPABLE</td>
</tr>
<tr>
<td>304</td>
<td>TCDB ACCESS ERROR IN OAM</td>
</tr>
<tr>
<td>305</td>
<td>TCDB AUTHORIZATION ERROR IN OAM</td>
</tr>
<tr>
<td>306</td>
<td>OAM INTERNAL ERROR</td>
</tr>
<tr>
<td>307</td>
<td>VOLSER NOT IN TCDB</td>
</tr>
<tr>
<td>310</td>
<td>MEDIA TYPE OR RECORDING TECHNOLOGY NOT SUPPORTED AT THIS SOFTWARE LEVEL</td>
</tr>
<tr>
<td>311</td>
<td>DELETED WITH MTL SOFTWARE-ONLY</td>
</tr>
<tr>
<td>312</td>
<td>FUNCTION NOT COMPATIBLE WITH THE LIBRARY</td>
</tr>
<tr>
<td>313</td>
<td>VOLUME IS CURRENTLY IN USE</td>
</tr>
<tr>
<td>314</td>
<td>IMPORT/EXPORT ALREADY IN PROGRESS OR HOST PROCESSING NOT COMPLETE</td>
</tr>
<tr>
<td>315</td>
<td>NOT ENOUGH PHYSICAL DRIVES AVAILABLE IN VTS</td>
</tr>
<tr>
<td>316</td>
<td>IMPORT/EXPORT NOT IN PROGRESS</td>
</tr>
<tr>
<td>317</td>
<td>EMPTY CATEGORY</td>
</tr>
<tr>
<td></td>
<td>IMPORT: NO IMPORT VOLUMES</td>
</tr>
<tr>
<td></td>
<td>EXPORT/IMPORT: NO SCRATCH VOLUMES</td>
</tr>
<tr>
<td>318</td>
<td>MAXIMUM LOGICALS DEFINED TO LIBRARY</td>
</tr>
<tr>
<td>319</td>
<td>NO TCDB RECORD AND VOLUME NOT FOUND IN SPECIFIED LIBRARY</td>
</tr>
<tr>
<td>320</td>
<td>NO TCDB RECORD AND UNABLE TO ACCESS SPECIFIED LIBRARY</td>
</tr>
<tr>
<td>321</td>
<td>MANUAL CARTRIDGE ENTRY FAILED MEDIA TYPE RETURNED FROM THE INSTALLATION EXIT DOES NOT MATCH THE MEDIA TYPE DEFINED IN THE VOLUME RECORD.</td>
</tr>
<tr>
<td>322</td>
<td>COMMAND REJECTED BY THE LIBRARY</td>
</tr>
<tr>
<td>323</td>
<td>UNABLE TO RETRIEVE POLICY NAME(S) FROM LIBRARY</td>
</tr>
<tr>
<td>324</td>
<td>VOLUME EXPIRE TIME HAS NOT ELAPSED</td>
</tr>
<tr>
<td>325</td>
<td>I/O TERMINATED DUE TO TIMEOUT DETECTION</td>
</tr>
<tr>
<td>400</td>
<td>OAM INITIALIZED WITH NULL CONFIGURATION</td>
</tr>
<tr>
<td>401</td>
<td>LIBRARY NOT ACCESSIBLE, OFFLINE, OR NOT OPERATIONAL</td>
</tr>
<tr>
<td>402</td>
<td>VISION SYSTEM NOT OPERATIONAL</td>
</tr>
<tr>
<td>403</td>
<td>EJECT PROCESSING HAS BEEN DISABLED BECAUSE AN ERROR IN THE EJECT INSTALLATION EXIT (CBRUXEJC) HAS BEEN DETECTED</td>
</tr>
<tr>
<td>404</td>
<td>OAM ADDRESS SPACE NOT AVAILABLE</td>
</tr>
</tbody>
</table>
EDGSPLCS utility example
You can run multiple copies of EDGSPLCS each using different parameters so that processing can be done in parallel for multiple libraries. This is shown in Figure 8-26.

Figure 8-26   EDGSPLCS utility example

Figure 8-27 shows the output file for each of the JCL EXEC statements run against the input file.

Figure 8-27   EDGSPLCS OUTPUT file
8.3 Control data set (CDS) serialization

DFSMSrmm now requires you to supply a control data set identifier in the CSDID parameter of SYS1.PARMLIB member EDGRMMxx, specifying the identifier of the CDS to be used on that system. DFSMSrmm uses the CSDID as part of the ENQ name used to serialize updates to the RMM CDS. The DFSMSrmm CIM Provider also uses the CSDID to distinguish between multiple control data sets. If the CSDID is not yet set in the DS, you can optionally use EDGUTIL UPDATE to set it, otherwise DFSMSrmm will set the CSDID when first started. We recommend that you use a unique CSDID for each CDS.

At DFSMSrmm startup time, DFSMSrmm matches the CDSID in the control data set control record value with the CDSID operand in PARMLIB member EDGRMMxx.

8.3.1 CDS serialization change

The serialization used by DFSMSrmm for its CDS is changed to use a new resource name that includes the CSDID. This avoids conflicts when multiple RMMplexes run in the same sysplex. Consider the scenario in Figure 8-28.

![Figure 8-28  DFSMSrmm CDS serialization](image)

In previous DFSMSrmm releases, even though the Production and the Test RMM are independent from each other, a write access to the Test CDS would cause an exclusive ENQ of the SYSZRMM MASTER.RESERVE resource. If you use global reserve, the accesses to the Production CDS and to the Test CDS are serialized each after the other. With z/OS V1.9, DFSMSrmm write accesses to the Production CDS and write accesses to the Test CDS are serialized independent from each other.
For example, if your production and test DFSMSrmm run in the same sysplex, you will get two exclusive SYSTEMS enqueues for write access—one for production and the other for test in parallel—as shown in Figure 8-29.

<table>
<thead>
<tr>
<th>SYSTEM ENQ STATUS</th>
<th>ROW 1 TO 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAJOR NAME PREFIX</td>
<td>. . . SYSZRMM (SYSDSN, SPFEDIT, ETC)</td>
</tr>
<tr>
<td>MAJOR MINOR</td>
<td>TYPE</td>
</tr>
<tr>
<td>+------------------</td>
<td>------------</td>
</tr>
<tr>
<td>SYSZRMM</td>
<td>MASTER.RESERVE.CDSPRODA</td>
</tr>
<tr>
<td>SYSZRMM</td>
<td>MASTER.RESERVE.CDSTEST1</td>
</tr>
<tr>
<td>SYSZRMM</td>
<td>RMM.ACTIVE</td>
</tr>
<tr>
<td>+------------------</td>
<td>------------</td>
</tr>
</tbody>
</table>

**Figure 8-29**  SYSTEM ENQ for write access

### 8.3.2 CDSID PARMLIB option

There is an option in the EDGRMMxx PARMLIB member called CDSID. This specifies the identifier of the control data set that must be used on this system. Specify a value one to eight alphanumeric characters long. This ID is used by DFSMSrmm Web services to distinguish retrieved data between multiple control data sets. Ensure that each DFSMSrmm control data set has a unique CDSID ID.

When you start DFSMSrmm, the CDSID ID is compared to the ID in the control data set control record. If the IDs match, DFSMSrmm startup continues. If the control data set does not have an ID, DFSMSrmm creates the ID in the control record from the CDSID. If the IDs do not match, DFSMSrmm startup fails and DFSMSrmm issues a message to the operator to select another PARMLIB member.

**CSDID(ID)**  Specifies the identifier of the control data set that must be used on this system. Specify a value one to eight alphanumeric characters long.

Figure 8-30 shows you the use of the CDSID PARMLIB option.

**Figure 8-30**  EDGRMMxx PARMLIB with a CDSID

### 8.3.3 Creating a CDSID to a new RMM system

The CDSID value in EDGUTIL will set or change the control data set ID. EDGUTIL does not validate the CDSID in the control data set control record; it simply sets the new value into the control record.

When you change the CDSID, any running systems that share the control data set can detect the change in CDSID and change the ENQ name that they use for serialization. Ensure that the GRSRNLLxx PARMLIB member is updated to reflect any CDSID changes you make.

You can use the sample JCL shown in Figure 8-31 to create a DFSMSrmm Control Data Set CDSID.
8.3.4 Providing a CDSID to an existing RMM system

We recommend that you provide a CDSID prior to the V1.9 installation. Ensure that each control data set has a unique CDSID. By using the RMM LISTCONTROL command as shown in Example 8-5, check whether the data element CDSID is filled or not.

Example 8-5  RMM LISTCONTROL subcommand

```
RMM LISTCONTROL OPTION
or
RMM LC OPT
```

In Figure 8-32 you can see the result of the command.

```
System options:
PARMLIB Suffix  = 02
Operating mode  = P  Retention period: Default = 0  Maximum = NOLIMIT
                  Catalog = 6  hours
Control data set name = RMM.CONTROL.DSET
Journal file data set name = RMM.JOURNAL.DSET
Journal threshold = 75%
Catalog SYSID = Notset
Scratch procedure name = EDGXPROC
Backup procedure name = EDGCDSBK
IPL date check = N  Date format = J  RACF support = N
SMF audit = 248  SMF security = 249  CDS id =
MAXHOLD value = 100  Lines per page = 54  System ID = SC70
BLP = RMM  TVEXT purge = RELEASE  Notify = N
Uncatalog = Y  VRS job name = 2  Message case = M
MASTER overwrite= USER
Accounting = J  VRS selection = NEW
VRS change = INFO
...
```

Figure 8-32  Result of the RMM LISTCONTROL subcommand

If the CDSID field is empty, as is the case in our example, run EDGUTIL with the UPDATE parameter as shown Example 8-6 and change your PARMLIB member accordingly.

Example 8-6  EDGUTIL JCL with the CDSIS specified

```
//UTIL EXEC PGM=EDGUTIL,PARM='CREATE'
//SYSPRINT DD SYSSOUT=*  
//MASTER DD DISP=SHR,DSN=RMM.CONTROL.DSET
//SYSIN DD *
CONTROL CDSID(CDSPRODA)
/*
```

Figure 8-31  Create a DFSMSrmm CDSID
Chapter 8. DFSMSrmm V1.9 enhancements

Note: You can use the EDGUTIL with PARM=’UPDATE’ to update the current defined CDSID if you would like to use another name.

Figure 8-33 shows you CDS ID information after we have updated the DFSMSrmm control record.

<table>
<thead>
<tr>
<th>System options:</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARMLIB Suffix  = 02</td>
</tr>
<tr>
<td>Operating mode   = P</td>
</tr>
<tr>
<td>Retention period: Default = 0 Maximum = NOLIMIT</td>
</tr>
<tr>
<td>Catalog = 6 hours</td>
</tr>
<tr>
<td>Control data set name = RMM.CONTROL.DSET</td>
</tr>
<tr>
<td>Journal file data set name = RMM.JOURNAL.DSET</td>
</tr>
<tr>
<td>Journal threshold = 75%</td>
</tr>
<tr>
<td>Catalog SYSID = Notset</td>
</tr>
<tr>
<td>Scratch procedure name = EDGXPROC</td>
</tr>
<tr>
<td>Backup procedure name = EDGCDSBK</td>
</tr>
<tr>
<td>IPL date check = N</td>
</tr>
<tr>
<td>Date format = J</td>
</tr>
<tr>
<td>SMF audit = 248</td>
</tr>
<tr>
<td>SMF security = 249</td>
</tr>
<tr>
<td>MAXHOLD value = 100</td>
</tr>
<tr>
<td>Lines per page = 54</td>
</tr>
<tr>
<td>BLP = RMM</td>
</tr>
<tr>
<td>TVEXT purge = RELEASE</td>
</tr>
<tr>
<td>Notify = N</td>
</tr>
<tr>
<td>Uncatalog = Y</td>
</tr>
<tr>
<td>VRS job name = 2</td>
</tr>
<tr>
<td>MASTER overwrite = USER</td>
</tr>
<tr>
<td>Accounting = J</td>
</tr>
<tr>
<td>VRS change = INFO</td>
</tr>
</tbody>
</table>

Figure 8-33 Result of the RMM LISTCONTROL subcommand after update of the control record

8.3.5 DFSMSrmm messages

The error message shown in Example 8-7 is issued during DFSMSrmm startup if there is no control data set identifier (CDSID) is set.

Example 8-7 EDG0237E error message

EDG0237E MISSING IDENTIFIER FOR THE CONTROL DATA SET

Where:

**EDG0237E**

**MISSING IDENTIFIER FOR THE CONTROL DATA SET**

**Explanation:** During initialization DFSMSrmm checks to ensure that you have a CDSID specified in the EDGRMMxx PARMLIB member. A CDSID is mandatory.

**System action:** DFSMSrmm initialization stops. This message is followed by message EDG0107A.

**System Programmer response:** Notify the system programmer. Reply to message EDG0107A as directed when the system programmer has corrected the error. Application Programmer Response: You must add a CDSID to the OPTION statement in EDGRMMxx before attempting to start DFSMSrmm on a release at z/OS V1.9 or higher. Correct the error in the startup parameters. See z/OS DFSMSrmm Implementation and Customization Guide for information about the DFSMSrmm PARMLIB options.
You also get an EDGUTIL message if no CDSID is provided during CREATE and UPDATE SYSIN control statements. EDGUTIL CREATE returns these messages in Figure 8-34 when the CDSID is not provided.

```
EDG6007E IKJ56701I MISSING IDENTIFIER FOR THE CONTROL DATA SET
EDG6414E CONTROL DATA SET CONTROL RECORD CREATE FAILED
EDG6901I UTILITY EDGUTIL COMPLETED WITH RETURN CODE 12
```

Figure 8-34   EDGUTIL message for missing CSDID

Where:

EDG6007E text

Explanation: The SYSIN or PARM statement contains an unsupported parameter. The text is the unsupported SYSIN or PARM statement. In the message text: text An incorrect keyword or verb.

System action: The program fails.

Operator response: None.

System Programmer Response: Correct the input parameters and resubmit the job.

Source: DFSMSrmm

Detecting Module: EDGBKUP

EDG6414E CONTROL DATA SET CONTROL RECORD CREATE FAILED

Explanation: An error occurred and prevented the DFSMSrmm control data set control record from being created.

System action: The request fails.

Operator response: None.

System Programmer Response: Refer to error messages issued and resubmit the job.

Source: DFSMSrmm

Detecting Module: EDGUTIL
EDG6901I UTILITY utility_name COMPLETED WITH RETURN CODE return_code

**Explanation:** The DFSMSrmm utility completed with the highest return code that occurred during processing.

In the message text:
utility_name: The name of the utility running
return_code: Value returned indicating the results of processing

**System action:** The utility ends.

**Operator response:** None.

**Application Programmer Response:** If the return code is not zero, refer to messages issued by DFSMSrmm to determine the cause of the error.

**Source:** DFSMSrmm

**Detecting Module:** EDGBKUP

### 8.3.6 Global resource serialization (GRS)

When you change the CDSID, any running systems that share the control data set can detect the change in CDSID and change the ENQ name they use for serialization. Ensure that the GRSRNLxx PARMLIB member is updated to reflect and CDSID changes you make. RNLDEF must be updated so that each system has a reserve with that systems CDSID specified. This is so that RMM can issue the ENQ on a per system basis.

Depending on how you have determined the GRS parameters, you should also update the GRS resource name list in SYS1.PARMLIB(GRSRNLxx) as shown in Example 8-1 on page 53.

> **Important:** The length of the specified `cdsid` must be appended with blanks to 8 characters if you have specified TYPE(SPECIFIC).
An alternative to creating multiple exclusion RNL entries is to change the existing definition from SPECIFIC to GENERIC. See Figure 8-36 for an example.

| RNLDEF RNL(EXCL) TYPE(GENERIC) |
| QNAME(SYSZRMM) |
| RNAME(MASTER.RESERVE) |
| RNLDEF RNL(EXCL) TYPE(GENERIC) |
| QNAME(SYSZRMM) |
| RNAME(MASTER.RESERVE.cdsid) |

Figure 8-36 Changing an existing definition from SPECIFIC to GENERIC and add a new one.

8.3.7 Migration and coexistence considerations

When a V1.9 system will ENQ with the new enqueue name, all lower systems in the SYSPLEX, which are using the same shared CDS, must do the ENQ with the same new enqueue name, too. With the coexistence APAR OA17965, the following functions are installed on lower releases (V1R6 to V1.8):

- It performs the ENQ with the new enqueue name, if the CDSID enqueue setting is enabled.
- It displays the CDSID enqueue setting with LISTCONTROL, REXX and API.
- A warning message appears if the lower system has not provided the parameter CDSID in PARMLIB.

Note: A lower system never switches on the CDSID enqueue setting. This is only be done by a z/OS V1.9 DFSMSrmm system.

8.4 JCL data set names

The data set naming requirements for tape data sets supported by the TSO subcommands and API are relaxed so that any 44 character string can be used. This enables use of unqualified data set names such as those that are accepted by the MVS JCL DSNAME keyword. Quoted data set name strings allow any character string to be specified for a data set name except for leading blank and leading hex zero which are not supported.

Note: Quotes make the difference!

8.4.1 Quoted data set name rules

The following rules are for a “quoted” data set name:

- 'My Data/Set"Name".123!'
  - The name can include special characters.
  - It can contain lower case characters.
  - It can be any string.
  - It can be a maximum of 44 characters long.
  - It must not start with BLANK.
Non-significant special characters
The following detailed rules are for using non-significant special characters in a data set name:

- When a data set name contains special characters that are not significant to the system, other than hyphens, enclose it in apostrophes. For example, DSNNAME='DS/29'.
- Code each apostrophe that is part of the data set name as two consecutive apostrophes. For example, code DAYS'END as DSNNAME='DAYS"END'.
- The system ignores blank characters at the end of a data set name, even if the data set name is enclosed in apostrophes.

Significant special characters
The following special characters are significant to the system. Do not enclose them in apostrophes:

- Periods to indicate a qualified data set name. However, you must enclose in apostrophes a period immediately before a right parenthesis, immediately after a left parenthesis, or immediately before a comma; for example, DSNNAME='(ABC)' and DSNNAME='(ABC.)' and DSNNAME='A.B.C.'.
- Double ampersands to identify a temporary data set name. Note that if you use apostrophes, DSNNAME='&&AB' and DSNNAME='&AB' refer to the same data set.
- Double ampersands to identify an in-stream or sysout data set name.
- Parentheses to enclose the member name of a partitioned data set (PDS) or partitioned data set extended (PDSE), or the generation number of a generation data set.
- Plus (+) or minus (-) sign to identify a generation of a generation data group.
- The asterisk to indicate a backward reference.

On a DD statement in a cataloged or in-stream procedure, if the data set name is a symbolic parameter, do not enclose it in apostrophes. If it is enclosed in apostrophes, the system performs correct substitution only if the symbolic parameter enclosed in apostrophes is preceded by a symbolic parameter not enclosed in apostrophes.

The data set name should not contain the 44 special characters (X'04') created by the 12-4-9 multiple punch or any operation that converts the value of characters to X'04'.

8.4.2 Un-quoted data set name rules

The following rules are for an “un-quoted” data set name:

- MVS.DATA.SET.NAME
  - A prefix can be applied.
  - It is translated to upper case.
  - It must follow MVS naming conventions.

8.4.3 Implementation

The new naming convention rules for data set names and data set name masks apply only on z/OS V1.9 and later releases. Caution should be used when creating new data set name VRSSs that include special characters or do not conform to the data set naming rules. The new masks might not be able to be processed by lower level systems, so ensure that VSREL runs on a z/OS V1.9 or later release system.
8.4.4 RMM subcommands

With this new support, the following RMM TSO subcommands are affected:

- Data set commands:
  - ADDDATASET
  - CHANGEDATASET
  - DELETEDATASET
  - LISTDATASET
  - SEARCHDATASET

- Volume commands:
  - ADDVOLUME DSNAME(dsname)
  - CHANGEVOLUME DSNAME(dsname)

- Vital record specification commands:
  - ADDVRS DSNAME(dsname)
  - CHANGEVRS DSNAME(dsname)
  - DELETEVRS DSNAME(dsname)
  - LISTVRS DSNAME(dsname)
  - SEARCHVRS DSNAME(dsname)

Data set name validation is only performed when a data set or VRS data set name record might be created in the CDS. This includes:

- ADDDATASET dsname
- CHANGEDATASET NEWDSNAME(dsname)
- ADDVOLUME DSNAME(dsname)
- CHANGEVOLUME DSNAME(dsname)
- ADDVRS DSNAME(dsname)
- CHANGEVRS DSNAME(dsname)

**Important**: The new masks might not be able to be processed by lower level systems. To avoid potential errors ensure that VSREL runs on a z/OS V1.9 or later release system.

In all other cases, the validation of a data set name is minimal so that any data set recorded already defined to RMM can be listed, changed, searched, or deleted.

You can define a retention policy for a specific data set by using a fully qualified data set name in a vital record specification. The vital record specification shown in Example 8-8 defines the retention policy for one data set, PRITCHAR.BACKUP.DATA. All copies of the data set should be retained for five days.

**Example 8-8  RMM ADDVRS**

RMM ADDVRS DSNAME('PRITCHAR.BACKUP.DATA') DAYS COUNT(5)

DFSMSrmm does not check quoted data set names for valid characters. Any string of up to 44 characters is accepted, except those that start with a blank or x'00'.
Figure 8-37 shows you the use of a Non-Significant Special Character data set name in a RMM TSO CHANGEDATASET command.

```plaintext
//SCHLUMGB JOB (999,POK),MSGLEVEL=1,NOTIFY=&SYSUID
/*JOBPARM SYSAFF=SC70
  //*
  //TESTCASE EXEC PGM=IKJEFT01
  //SYSTSPRT DD SYSOUT=* 
  //SYSTSIN DD *
  RMM CD 'MHLRES7/GARB/TEST/CASE/WITH.LONG.CHAR.STRING' -
   SEQ(1) VOLUME(TH5029) FORCE -
   NEWDSN('MHLRES7 GARB/TEST/CASE/WITH.LONG.CHAR.STRING')
/*
```

Figure 8-37   Use of non-significant special characters in a data set name

**8.4.5 Creating a CDS record**

When a CDS record is created, DFSMSrmm does not check quoted data set names or data set name masks for valid characters; any string of up to 44 characters is accepted except those which start with a blank or x'00'. Unquoted data set names and data set name masks must pass the data set naming rules which are described here one time and are as follows:

- A data set name can be one or more qualifiers.
- Each qualifier is 1 to 8 characters, the first of which must be alphabetic (A to Z) or national (# @ $). The remaining seven characters are either alphabetic, numeric (0 - 9), national, or a hyphen (-).
- Qualifiers are separated by a period (.)
- DFSMSrmm adds your TSO PROFILE PREFIX value as the high-level qualifier.
- The data set name must not include a member name.

**Note:** Because z/OS V1.9 DFSMSrmm no longer folds data set names to uppercase letters when you specify quoted data set names, when you specify data set names or data set name masks, be sure to specify the correct case for each character. If you create VRS data set name masks with lowercase or mixed case letters, these will not match to data sets with all uppercase characters.

In addition, data set name masks must pass the following data set mask naming rules:

- You can use *, %, or ¬ in a data set name mask.
- 
  - A single * represents a single qualifier of any number of characters.
  - A single * when used within a qualifier represents zero or more characters.
  - More than one single * can be used within a qualifier as long as a character precedes or follows the *.
- **
  - represents zero or more qualifiers. At the end of the mask, ** indicates to ignore any remaining characters.
** indicates to select all data sets. You can use this mask to define a vital record specification that sets your installation default retention criteria for data sets that are not covered by other vital record specifications.

% (percent sign)
- A place holder for a single character.

¬ (not sign)
- A place holder for a single character. The ¬ has special meaning in a VRS data set name mask and is used to specify a pseudo-GDG data set name.

### 8.4.6 Data set names in RMM subcommands

When an existing record in the CDS is being processed, the checks are as follows:

- The name is 1 to 44 characters, enclosed in quotes if any special characters are included. If the data set name is not enclosed in quotes PROFILE PREFIX is applied but there is no check against data set naming or data set mask naming rules.

- Data set names are validated differently when:
  - Creating new data sets
  - Using existing data sets

This differentiation is shown in Figure 8-38 and is shown by points 1, 2, and 3 in the diagram.

![Figure 8-38 RMM subcommands](image)

### 8.4.7 Examples of RMM subcommands

With reference to Figure 8-38, following are some examples of specifying data set names or data set name masks.
Data set names

Fully qualified data set names are checked following the rules described below:

1. Validation of an unquoted data set name when a data set is created.
   a. The data set name must follow the standard MVS naming convention. The creation of the data set names will give the following results:
      - **RMM AD TEST.NEW** Data set USERID.TEST.NEW will be added.
      - **RMM AD spec/** IKJ56702I INVALID DATA SET NAME, special" will be issued.

2. Validation of a quoted data set name when a data set is created.
   a. The data set name must not start with blank or null and the maximum length is 44 characters. The creation of these data set names will give the following results.
      - **RMM AD ‘01/file’** Data set name ‘01/file’ is added.
      - **RMM AD ‘ text1’** IKJ56702I INVALID DATA SET NAME is issued because the data set name starts with a blank.

3. Validation of a data set name when an existing data set is used.
   a. The maximum data set name length is 44 character. The display of the data set name will give the following results.
      - **LD ‘001/file’** The data set is listed.
      - **LD ‘ text1’** EDG3201I THE ENTRY IS NOT DEFINED TO DFSMSrmm is issued because the file was not created in point 2 above.

Data set name masks

Data set names masks are validated differently when using data set name masks as follows:

- RMM subcommands for creating a VRS
- RMM subcommands for existing generic data set names and masks

This differentiation is shown in Figure 8-39 and explained in points 1, 2, 3 and 4 shown in the diagram.

![Figure 8-39 Masks in RMM subcommands](image)
With reference to Figure 8-39 on page 89, consider the following points:

1. Validation of an unquoted data set name VRS at create time (AS):
   a. The data set name mask must follow standard MVS naming conventions.
   b. It must follow common filtering and GDG rules (*, **, %, ¬):

   \[
   \text{RMM AS DSNAMES('TEST.NEW.**') VRS DSNAMES USERID.TEST.NEW.** will be added.}
   \]

   \[
   \text{RMM AS DSNAMES('TEST.NEW.**') VRS DSNAMES TEST.NEW.** will be added.}
   \]

2. Validation of a quoted Data Set Name VRS at create time (AS):
   a. It must follow filtering rules for period and blank, which are as follows:
      - The data set name must not start with blank or null.
      - It must not start or end with period nor use consecutive periods.
   b. It must follow common filtering and GDG rules (*, **, %, ¬).
   c. Maximum length is 44 characters.

   \[
   \text{RMM AS DSNAMES('01/file.**') VRS DSNAMES 01/file.** will be added.}
   \]

3. Validation of a Data Set Name Mask when an existing mask is used (LS, DS, CS):
   a. Maximum length is 44 characters.

4. Validation of a Data Set Name Mask when an existing mask is used in SEARCH subcommands (SS, SD):
   a. When a data set name mask does not follow common filtering or GDG rules or filtering rules for period and blank:
      - The SEARCH subcommand is processed with fully qualified data set name.
   b. Maximum length is 44 characters.
Figure 8-40 shows a few examples of data set name masks in RMM add subcommands without using quotes. As you can see, the current TSO prefix of the user is always added in front to the data set name.

Where:

The operands are as follows:

- **TE****: ** is not allowed within a qualifier.
- **SPE****: the / (slash) is not a valid character, because you can use only alphabetic (A-Z), numeric (0-) or national (#, @ $) characters.
In Figure 8-41 we show you some examples of data set name masks in RMM ADD and LIST subcommands using quotes. As you can see, DFSMSrmm no longer folds data set names to uppercase letters, nor are the data sets validated to following the MVS naming convention.

In Figure 8-41 we show you some examples of data set name masks in RMM ADD and LIST subcommands using quotes. As you can see, DFSMSrmm no longer folds data set names to uppercase letters, nor are the data sets validated to following the MVS naming convention.

<table>
<thead>
<tr>
<th>RMM ADDVRS DSNAME('test.special.z/OS 1.9.**')</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMM LISTVRS DSNAME('test.special.z/OS 1.9.**')</td>
</tr>
<tr>
<td>Data set mask = test.special.z/OS 1.9.** Type = DSNAME</td>
</tr>
<tr>
<td>Job name mask = Retain until expired = NO</td>
</tr>
<tr>
<td>Count = 99999 CYCLES Retain while cataloged = NO</td>
</tr>
<tr>
<td>Delay = 0 Days in the HOME location</td>
</tr>
<tr>
<td>Store number = 99999 CYCLES in the HOME location</td>
</tr>
<tr>
<td>Priority = 0 Release Options:</td>
</tr>
<tr>
<td>Expiry date ignore = NO</td>
</tr>
<tr>
<td>Scratch immediate = NO</td>
</tr>
<tr>
<td>Next VRS in chain = using VRS</td>
</tr>
<tr>
<td>VRS Owner = MHLRES7</td>
</tr>
<tr>
<td>Description =</td>
</tr>
<tr>
<td>Last Reference: Date = Time =</td>
</tr>
<tr>
<td>Vital Record Specification to be deleted on 1999/365</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RMM SEARCHVRS DSNAME('test.special.z/OS 1.9.**') OWNER(<em>) LIMIT(</em>)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vital Record Specification Job name Type Location Next VRS</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td>test.special.z/OS 1.9.**</td>
</tr>
<tr>
<td>DSN HOME</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EDG3011I 1 ENTRY LISTED</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMM ADDVRS DSNAME('z/OS 1.9**.*')</td>
</tr>
<tr>
<td>IKJ56702I INVALID DATA SET NAME MASK., 'z/OS</td>
</tr>
<tr>
<td>RMM ADDVRS DSNAME('.spe/.')</td>
</tr>
<tr>
<td>IKJ56702I INVALID DATA SET NAME MASK., '.spe/'</td>
</tr>
<tr>
<td>IKJ56701I MISSING DATA SET NAME MASK.+</td>
</tr>
<tr>
<td>IKJ56701I MISSING DATA SET NAME MASK.</td>
</tr>
<tr>
<td>RMM ADDVRS DSNAME(' spe/.' )</td>
</tr>
<tr>
<td>IKJ56702I INVALID DATA SET NAME MASK., '</td>
</tr>
</tbody>
</table>

Figure 8-41 Add and display VRSs with quoted data set names

8.4.8 Changes in vital record specification (VRSEL) processing

The new naming convention rules for data set names and data set name masks apply only on z/OS V1.9 and later releases. Caution should be used when creating new data set name VRSs which include special characters or do not conform to the data set naming rules. The new masks might not be able to be processed by lower level systems, so ensure that VRSEL runs on a z/OS V1.9 or later release system.
In z/OS V1.9 DFSMSrmm VRSEL processing data set names are not translated to upper case. This is shown in Figure 8-42.

![Diagram](image)

**Prior to V1R9:**
In VRSEL processing Data Set Names are translated to UPPER case.

**V1R9:**
In VRSEL processing Data Set Names are not translated to UPPER case.

For data sets with lower or mixed case names that have been created prior to V1.9, the following migration aid must be honored. To check whether you have any lowercase or mixed case data set names that are on volumes that are VRS-retained, do the following tasks:

1. From a lower-level system with access to the z/OS V1.9 SYS1.SAMPLIB, copy the SAMPLIB member EDGGDSNM to your own report library (userid.REPORT.LIB). Although the report is ready to use, you can modify this EDGGDSNM sample to create your own tailored report.
2. Using the DFSMSrmm report generator, generate the JCL to be used to create the report. If you do not have an existing report extract data set available to be used as input for the report, select the option to have the generated JCL create a new extract.
3. Run the report. The report lists only those data sets that have lowercase or mixed case data set names that are on volumes that are VRS-retained. Note that when you run a report extract data set created for a release earlier than z/OS V1.9, there is no matching VRS information listed because earlier z/OS releases did not maintain this information for mixed case or lowercase data set names.

If the report indicates that you have lowercase or mixed case data set names on volumes that are VRS-retained, identify the VRSs that retain the data sets. If any of the matching DSNAME VRSs contain uppercase letters that match the lowercase or mixed case letters in the data set names, the VRSs will no longer match the data set names in z/OS V1.9. You must create new VRSs or change the existing VRSs for the data sets. Data sets that match generic characters in the VRS data set name masks are not affected.

**Note:** While you run VRSEL vital record processing on releases below z/OS V1.9, you should not delete any VRSs that retain data sets with lowercase or mixed case letters, or you might lose data.
Figure 8-43 is an example output of the migration aid (EDGGDSNM).

### 8.4.9 Security considerations

If you have implemented protection for your tape data sets as well as your data sets on DASD, you need access to the resource protecting this kind of data set. If you do not have access to the resource protecting the data set you get a security violation as shown in Figure 8-44. To bypass any security check for this data set, you can specify in the JCL statement the special parametric EXPDT=98000, but in this case you need access to the resource STGADMIN.EDG.IGNORE.TAPE.volser in RACF class FACILITY.
In our case the System Authorization Facility (SAF) was set as follows:

- TAPEAUTHDSN: YES
- TAPEAUTHF1: YES
- TAPEAUTHRC4: FAIL
- TAPEAUTHRC8: FAIL

8.5 Shared PARMLIB support

Some information in the EDGRMMxx PARMLIB might need to be specific to a subset of your systems. For example, the REJECT or VLPOOL entries might need to be different across systems. To enable this information to be handled on a system by system basis you can specify a second PARMLIB member to be used.

8.5.1 Visual overview

Use the MEMBER operand in the primary DFSMSrmm PARMLIB to identify a second PARMLIB member that contains overriding or additional PARMLIB options. Some information in the EDGRMMxx PARMLIB member might need to be specific to a subset of your systems. For example, the REJECT or VLPOOL entries need to be different as shown in Figure 8-45.

Figure 8-45 Second PARMLIB overview

8.5.2 OPTION MEMBER(xx)

The parmlib_suffix must be any 2 characters used as the suffix of the EDGRMMxx PARMLIB member name. Starting with z/OS V1.9, you can use system symbols (like &SYSCLONE) to enable easier sharing of the EDGRMMxx PARMLIB member. The system symbol must resolve to 2 characters used as the suffix of the EDGRMMxx PARMLIB member name.

DFSMSrmm processes the commands in the EDGRMMxx PARMLIB member; then, if there is a second member named, it processes the second member. Any OPTION operands specified, other than MEMBER, override the values set in the first member. Any other PARMLIB commands can add to, but not replace, update, or duplicate any command from the first PARMLIB member. DFSMSrmm processing is as if all of the PARMLIB contents of both members had been specified in a single PARMLIB member as shown in Figure 8-46.
8.5.3 LISTCONTROL OPTION

The RMM LISTCONTROL OPTION output has been extended to display the second PARMLIB member name suffix as shown in Figure 8-47.

<table>
<thead>
<tr>
<th>System options:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PARMLIB Suffix</strong> = 02 70</td>
</tr>
<tr>
<td>Operating mode = P</td>
</tr>
<tr>
<td>Retention period: Default = 0, Maximum = NOLIMIT</td>
</tr>
<tr>
<td>Catalog = 6 hours</td>
</tr>
<tr>
<td>Control data set name = RMM.CONTROL.DSET</td>
</tr>
<tr>
<td>Journal file data set name = RMM.JOURNAL.DSET</td>
</tr>
<tr>
<td>Journal threshold = 75%</td>
</tr>
<tr>
<td>Catalog SYSID = Notset</td>
</tr>
<tr>
<td>Scratch procedure name = EDGXPROC</td>
</tr>
<tr>
<td>Backup procedure name = EDGCDSBK</td>
</tr>
<tr>
<td>IPL date check = N</td>
</tr>
<tr>
<td>Date format = J</td>
</tr>
<tr>
<td>RACF support = N</td>
</tr>
<tr>
<td>SMF audit = 248</td>
</tr>
<tr>
<td>SMF security = 249</td>
</tr>
<tr>
<td>CDS id = SC70</td>
</tr>
<tr>
<td>MAXHOLD value = 100</td>
</tr>
<tr>
<td>Lines per page = 54</td>
</tr>
<tr>
<td>System ID = SC70</td>
</tr>
<tr>
<td>BLP = RMM</td>
</tr>
<tr>
<td>TVEXT purge = RELEASE</td>
</tr>
<tr>
<td>Notify = N</td>
</tr>
<tr>
<td>Uncatalog = Y</td>
</tr>
<tr>
<td>VRS job name = 2</td>
</tr>
<tr>
<td>Message case = M</td>
</tr>
<tr>
<td>MASTER overwrite = USER</td>
</tr>
<tr>
<td>Accounting = J</td>
</tr>
<tr>
<td>VRS selection = NEW</td>
</tr>
<tr>
<td>VRS change = INFO</td>
</tr>
<tr>
<td>VRSMIN action = INFO</td>
</tr>
<tr>
<td>VRSMIN count = 1</td>
</tr>
<tr>
<td>VRSDROP action = INFO</td>
</tr>
<tr>
<td>VRSDROP count = 0</td>
</tr>
<tr>
<td>percent = 10</td>
</tr>
<tr>
<td>VRSRETAI N action = INFO</td>
</tr>
<tr>
<td>VRSRETAI N count = 0</td>
</tr>
<tr>
<td>percent = 80</td>
</tr>
<tr>
<td>EXPDTRDROP action = INFO</td>
</tr>
<tr>
<td>EXPDTRDROP count = 0</td>
</tr>
<tr>
<td>percent = 10</td>
</tr>
</tbody>
</table>

Figure 8-46 Sample EDGRMMnn PARMLIB member

Figure 8-47 LISTCONTROL OPTION command
8.5.4 Restarting DFSMSrmm

After you have updated your EDGRMMnn PARMLIB definitions, you can restart DFRMM using the modify command as shown in Figure 8-48.

```
F DFRMM,M=02
```

*Figure 8-48  Modify DFRMM settings*

The use of the new member or the multiple members is displayed in message EDG0204I. For each member that is used you get this message as shown in Figure 8-49. In the first example we are using only one EDGRMMnn member and in the second example we are using the new MEMBER option.

```
F DFRMM,M=00
EDG1101I DFSMSrmm MODIFY COMMAND ACCEPTED
EDG0204I DFSMSrmm BEING INITIALIZED FROM MEMBER EDGRMM00 IN RMM.PARMLIB
EDG0105I DFSMSrmm SUBSYSTEM INITIALIZATION COMPLETE

F DFRMM,M=02
EDG1101I DFSMSrmm MODIFY COMMAND ACCEPTED
EDG0204I DFSMSrmm BEING INITIALIZED FROM MEMBER EDGRMM02 IN RMM.PARMLIB
EDG0204I DFSMSrmm BEING INITIALIZED FROM MEMBER EDGRMM70 IN RMM.PARMLIB
EDG0105I DFSMSrmm SUBSYSTEM INITIALIZATION COMPLETE
```

*Figure 8-49  EDG0204I message display*

8.5.5 DFSMSrmm installation wizard

Figure 8-50 shows the new V1.9 wizard version to enable a common PARMLIB member for multiple systems and system specific PARMLIBs that have different values for each system.

The URL for the wizard is:

```
```

*Figure 8-50  DFSMSrmm installation wizard*
8.6 TSO subcommands

TSO subcommand parsing rules are further relaxed to support different product versions and declassification of data sets and volumes. We discuss the following topics:

- Changes in subcommand parsing:
  - The rules for the Product Level have changed.
  - A new NOSECLEVEL parameter was introduced to reset the SECLEVEL for a volume or data set.

- CLIST enhancements:
  - New parameters START and ADD were introduced to enable the user to either overwrite or add to an existing CLIST data set.
  - Any existing or user specified DCB record format parameters are now supported.

- Search command enhancements:
  - A new CONTINUE parameter was introduced for all search commands as a way to break down search results into manageable quantities.
  - There is a new STORAGEGROUP parameter for the Search Volume command.

- Report 17:
  - A new report, REPORT 17, is added to the EDGRRPTE exec. It summarizes information for logical and stacked volumes to support stacked volume management.

8.6.1 Changes in subcommand parsing

The product LEVEL specifies a software product's version. The form is VxxRxxMxx, indicating the Version, Release, and Modification level. Until now ‘x’ had to be numeric. With this release ‘x’ can be alphanumeric or national.

This change affects the following commands:

- ADDPRODUCT LEVEL( software_product_version )
- CHANGEPRODUCT LEVEL( software_product_version )
- LISTPRODUCT LEVEL( software_product_version )
- ADDVOLUME LEVEL( software_product_version )
- CHANGEVOLUME LEVEL( software_product_version )

Note: The default value is V01R01M00, Version 1, Release 1, Modification 0.
8.6.2 NOSECLEVEL parameter

The SECLEVEL specifies a volume or data sets security class, that is defined for your installation. To remove the security classification for a volume or data set a new parameter was introduced with DFSMSrmm V1.9 called NOSECLEVEL. The command syntax is as shown in Figure 8-51.

This change affects the following commands:

- CHANGEVOLUME NOSECLEVEL
- CHANGEDATASET NOSECLEVEL

![Diagram of the NOSECLEV subparameter of CHANGEVOLUME](image)

**Attention:** Using ISPF panels, you can reset the security level by clearing the SECLEVEL field. This will cause the appropriate execs to use the NOSECLEVEL parameter, when they generate the appropriate command.

8.6.3 CLIST enhancements

With the CLIST parameter for all search commands, you have the option to extend the results of your search to executable commands and to route these into a data set. Until now RMM generated a new CLIST data set (VB format, LRECL 255) for each command issued. With the new START/ADD parameters, the user now has the choice to either add records to an existing data set or, as before, to replace the existing records in the CLIST data set.

The default value is START, what means write the records to the start of the data set, which will replace any existing records. If you specify ADD, the records will be appended at the end of the data set. In addition to the subcommand request, the disposition of the allocated RMMCLIST data set is taken into account. DISP=MOD overrides the START operand.

This change affects the following commands:

- SEARCHBIN START/ADD
- SEARCHDATASET START/ADD
- SEARCHOWNER START/ADD
- SEARCHPRODUCT START/ADD
- SEARCHRACK START/ADD
- SEARCHVOLUME START/ADD
- SEARCHVRS START/ADD

**Note:** If DISP=MOD is coded, records will always be appended to the RMMCLIST data set.
Like the CLIST parameter, START / ADD can be specified in any SEARCH subcommand. The syntax of the START / ADD parameters is shown in Figure 8-52.

Figure 8-52  START/ADD subparameter of SEARCHVOLUME

Where:

The operands are as follows:

ADD  Specify this operand to request that new records written to the CLIST data set are added after any existing records in the data set. When the CLIST data set is empty or DFSMSrmm creates the CLIST data set during command execution, specifying ADD is the same as specifying START. ADD is mutually exclusive with START.

You can easily build a set of commands from CLIST processing using multiple SEARCH subcommands of the same or different resource types. For variable length records, the minimum record length can cause the LRECL to be increased. For fixed length records, if the minimum length cannot be accommodated, the subcommand fails.

START  Specify this operand to request that records written to the CLIST data set start from the beginning of the data set. START is mutually exclusive with ADD. START is the default value.

8.6.4 Sample JCL for SEARCH subcommand

Figure 8-53 contains two samples of JCL to issue a SEARCH subcommand and write data to a CLIST data set:

- In the first sample, ddname RMMCLIST &SYSUID..CLIST.DATA is specified in the JCL and the RMM subcommand request specifies parameter ADD.
  - Data will be appended to data set &SYSUID..CLIST.DATA.
- In the second sample, the same RMMCLIST data set is specified in the JCL and the subcommand request specifies parameter START. Because DISP=MOD is specified for the RMMCLIST data set, the result will be the same as in the first sample.
  - Data will be appended to data set &SYSUID..CLIST.DATA.
In Figure 8-54 you can see that now the result of the SEARCHDATASET is added to the previously used SEARCHVOLUME result.

```
CARTOA
CARTOB
CARTOC
CARTOD
CARTOE
CARTOF
CARTOI
CARTO2
CARTO3
CARTO4
'D9CG.D9C1.ALOG1.DR.D08066.T1657435.A0000025' VOL(CARTO1) FILESEQ(00001)
'D9CG.D9C1.ALOG1.DR.D08066.T1657435.A0000025' VOL(CART29) FILESEQ(00001)
'D9CG.D9C1.ALOG1.DR.D08066.T1736118.A0000026' VOL(CART2A) FILESEQ(00001)
'D9CG.D9C1.ALOG1.DR.D08066.T1745095.A0000027' VOL(CART2C) FILESEQ(00001)
'D9CG.D9C1.ALOG1.DR.D08067.T1406527.A0000028' VOL(CARTO2) FILESEQ(00001)
'D9CG.D9C1.ALOG1.DR.D08067.T1408194.A0000029' VOL(CART12) FILESEQ(00001)
'D9CG.D9C1.ALOG1.DR.D08067.T1434473.A0000030' VOL(CART14) FILESEQ(00001)
'D9CG.D9C1.ALOG1.DRD08066.T1736118.A0000026' VOL(CART25) FILESEQ(00001)
'D9CG.D9C1.ALOG1.DRD08066.T1745095.A0000027' VOL(CART26) FILESEQ(00001)
'D9CG.D9C1.ALOG2.DR.D08066.T1657435.A0000025' VOL(CART16) FILESEQ(00001)
```

**Figure 8-54  Use of ADD result**

### 8.6.5 User defined format of CLIST data set

Until now, the format of the RMM CLIST data set was fixed to VB 255. With release z/OS V1.9, DFSMSrmm starts to honor the DCB attributes of an existing CLIST data set. This means that you can pre-allocate data sets in a format that you prefer, and RMM will leave this format unchanged.
The following restrictions apply:

- Pre-allocated data sets can be fixed or variable format
- LRECL must be at least long enough to contain the CLIST information for the record type you are searching.
- If LRECL is too short, RMM increases LRECL to the minimum needed. If this is impossible, message EDG3360E is issued.
- The maximum LRECL supported is 32760.
- If data set does not exist, defaults are LRECL 255, VB format.

Following are the details to allocate a new CLIST data set:

- The data set used for the CLIST output by default will have variable length records and a maximum logical record size of 255 unless you pre-allocate the data set and specify a different record format and LRECL. CLIST data set records can be either fixed or variable length and can optionally be blocked.
- When you use a variable record length, the record length includes a 4-byte length field at the front of the record. DFSMSrmm honors the existing DCB attributes of the CLIST data set. When DFSMSrmm allocates a new CLIST data set or you do not specify DCB attributes, the defaults are variable length and an LRECL of 255. If the length of the output record exceeds the LRECL, DFSMSrmm splits the record into multiple records and adds a continuation character, +, to all but the last record.
- The LRECL must be at least long enough to contain the CLIST information for the record type you are searching. In addition, if a suffix or prefix is specified, the minimum LRECL is increased by 1 to allow for the continuation character. If the record is too short, DFSMSrmm increases the record to the minimum size required to hold the output information. The maximum LRECL supported is 32760.

EDG3360E CLIST DATA SET LRECL lrecl IS TOO SHORT - MINIMUM LRECL min_lrecl

Explanation: You have issued a SEARCH subcommand with the CLIST operand and the ADD or DISP=MOD operand. However, the LRECL for an existing data set with fixed record length is shorter than the minimum LRECL required for new records.

In the message text:

lrecl This is the existing logical record length of the CLIST data set.

min_lrecl This is the minimum record length required to create the CLIST output for a fixed length record data set.

System action: The subcommand fails.

Operator response: None.

Application Programmer Response: Either recreate the CLIST data set with a larger fixed LRECL or use variable length records.

Source: DFSMSrmm

Detecting Module: EDGTSOCL
8.6.6 Search command enhancements

When you issue a subcommand that you know you might want to continue, you specify the **CONTINUE** operand without any value. You use the **LIMIT** operand to specify how many search result entries you can manage each time you continue the search. After the first command you process the data returned and the continue_information returned, and, if more records exist, you repeat the command (now with continue_information) until all results are returned.

This change affects the following commands:

- **SEARCHBIN**
  CONTINUE
  CONTINUE(BIN(bin_number) MEDIANAME(media)
  LOCATION(loc_name) STORE(builtin_store))

- **SEARCHDATASET**
  CONTINUE
  CONTINUE(DSNAME(dsname) VOLUME(volser) FILESEQ(seq))

- **SEARCHOWNER**
  CONTINUE
  CONTINUE(OWNER(ownerid))

- **SEARCHPRODUCT**
  CONTINUE
  CONTINUE(NUMBER(product_number) LEVEL(version_number))

- **SEARCHRACK**
  CONTINUE
  CONTINUE(RACK(rack_number) MEDIANAME(media))

- **SEARCHVOLUME**
  CONTINUE
  CONTINUE(VOLUME(volser))

- **SEARCHVRS**
  CONTINUE
  CONTINUE(DSNAME(dsn_mask) JOBNAME(jobname_mask))
  CONTINUE(VOLUME(volume_mask))
  CONTINUE(NAME(vrs_mask))

The continue_information must be passed back to DFSMSrmm unchanged in order to continue the previous search. You should also specify the exact same subcommand unchanged, just changing the **CONTINUE** value on each additional command required.

The content of the continue_information depends on the type of search done. In general, the continue_information contains a valid key for the specific RMM resource searched for. For example:

- If you are searching for volumes, the continue_information will contain a volser:
  - VOLUME('V10035') or VOLUME('V10071')

- If you are searching for data sets, the continue_information will contain the data set name plus volser plus file sequence number:
  - DSNAME('RMMUSER.DATA36')VOLUME('V10003')FILESEQ(6)

In Figure 8-55 we see a typical SEARCH example of using the CONTINUE parameter. If you issue a search command from TSO environment, you get a table of resources returned and at the end of this table, there will be messages:

- EDG3203I SEARCH COMPLETE - MORE ENTRIES MAY EXIST
- EDG3012I xxxx ENTRIES LISTED
If you specify the CONTINUE parameter, there will be a new additional message now:

- EDG3025I continue_information

You can use this information to build a new search command, pulling in the next RMM resources in sequence.

```plaintext
//TESTCASE EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*  
//RMMCLIST DD DSN=SYSUID..CLIST.DATA2,DISP=(MOD,KEEP),  
//UNIT=SYSDA,SPACE=(TRK,(1,1),RLSE)
//SYSTSIN DD *
RMM SV OWNER(*) CLIST(' 1 - ',) LIMIT(10) CONTINUE START
*comment : next run the clist below using the highest volser# from the display* 
RMM SV OWNER(*) CLIST(' 2 - ',) LIMIT(10) CONTINUE(VOLUME(CART04)) ADD /*
```

**Figure 8-55** Example use of CONTINUE

Where:

The operands are as follows:

- **CONTINUE**: To continue a previous search subcommand
- **VOLUME(CART04)**: The last volume serial number of the previously search result you got in message EDG3025I.

Figure 8-56 shows you the result of the first SEARCHVOLUME command we used, and you can see the continuation information in message EDG3025I.

```plaintext
RMM SV OWNER(*) CLIST(' 1 - ',) CONTINUE START
Volume Owner    Rack   Assigned   Expiration Location Dsets St Act Dest.
------ -------- ------ ---------- ---------- -------- ----- -- ----- --------
CART0A PAOLOR6         2008/066   2008/072   SHELF    5     M
CART0B PAOLOR6         2008/066   2008/072   SHELF    5     M
CART0C PAOLOR6         2008/066   2008/072   SHELF    11    M
CART0D PAOLOR6         2008/066   2008/072   SHELF    11    M
CART0E PAOLOR6         2008/066   2008/072   SHELF    5     M
CART0F PAOLOR6         2008/066   2008/072   SHELF    5     M
CART01                 2008/070              SHELF    1     S
CART02                 2008/070              SHELF    1     S
CART03                 2008/070              SHELF    1     S
CART04                 2008/070              SHELF    1     S
EDG3203I SEARCH COMPLETE - MORE ENTRIES MAY EXIST
EDG3012I 10 ENTRIES LISTED
EDG3025I VOLUME('CART04')
READY
```

**Figure 8-56** Result of a SEARCHVOLUME using the CONTINUE operand
SEARCHBIN command syntax

In Figure 8-57 you can see the use of the new SEARCHBIN operand CONTINUE.

Where:

The operands are as follows:

**CONTINUE** Specify the CONTINUE operand without any value to notify DFSMSrmm SEARCH subcommand processing that you want to break down the search results based on the LIMIT value and request that DFSMSrmm return the search continue information for use with the next command. For TSO, the continue information is returned either as a REXX variable or as a linemode message. When the subcommand is issued from the DFSMSrmm API, the continuation information might be either a linemode message or an SFI or XML attribute.

CONTINUE is an optional operand.

Use the LIMIT operand to control the maximum number of entries to be returned each time you start or continue the search.

To continue a previous search subcommand, the CONTINUE operand value includes the following values to identify the current search position:

**BIN**  
*bin_number* is one to six characters enclosed in single quotation marks if it contains any special characters, or blank.

**LOCATION**  
*loc_name* is one to eight characters enclosed in single quotation marks if it contains any special characters, or blank.

**MEDIANAME**  
*media* is one to eight characters enclosed in single quotation marks if it contains any special characters, or blank.

**STORE** Enter one of the built-in storage location names: LOCAL, DISTANT, REMOTE. Abbreviations are acceptable.

For example, L stands for LOCAL.

- LOCAL The local storage location.
- DISTANT The distant storage location.
- REMOTE The remote storage location.
SEARCHDATASET command syntax

In Figure 8-58 you can see the use of the new SEARCHDATASET operand CONTINUE.

Where:

The operands are as follows:

**CONTINUE** Specify the CONTINUE operand without any value to notify DFSMSrmm SEARCH subcommand processing that you want to break down the search results based on the LIMIT value and request that DFSMSrmm return the search continue information for use with the next command. For TSO, the continue information is returned either as a REXX variable or as a linemode message. When the subcommand is issued from the DFSMSrmm API, the continuation information might be either a linemode message or an SFI or XML attribute.

CONTINUE is an optional operand.

Use the LIMIT operand to control the maximum number of entries to be returned each time you start or continue the search.

To continue a previous search subcommand, the CONTINUE operand value includes the following values to identify the current search position:

**DSNAME** $dsname$ is one to 44 characters enclosed in single quotation marks if it contains any special characters, or blank.

**VOLUME** $volser$ is one to six characters enclosed in single quotation marks if it contains any special characters, or blank.

**FILESEQ** $seq$ is a number within the value range 0 to 65535.
SEARCHOWNER command syntax
In Figure 8-59 you can see the use of the new SEARCHOWNER operand CONTINUE.

```
SEARCHOWNER
   |—— SD ———-
   |—— CONTINUE(——OWNER(—owner_id)—)——
```

Figure 8-59 SEARCHOWNER command syntax

Where:

The operands are as follows:

**CONTINUE** Specify the CONTINUE operand without any value to notify DFSMSrmm SEARCH subcommand processing that you want to break down the search results based on the LIMIT value and request that DFSMSrmm return the search continue information for use with the next command. For TSO, the continue information is returned either as a REXX variable or as a linemode message. When the subcommand is issued from the DFSMSrmm API, the continuation information might be either a linemode message or an SFI or XML attribute. CONTINUE is an optional operand.

Use the LIMIT operand to control the maximum number of entries to be returned each time you start or continue the search.

To continue a previous search subcommand, the CONTINUE operand value includes the following values to identify the current search position:

**OWNER** ownerid is one to eight characters enclosed in single quotation marks if it contains any special characters, or blank.

SEARCHPRODUCT command syntax
In Figure 8-60 you can see the use of the new SEARCHPRODUCT operand CONTINUE.

```
SEARCHPRODUCT
   |—— SP ———-
   |—— CONTINUE(——NUMBER—(-product_number—)–LEVEL—(-version_number—)——)
```

Figure 8-60 SEARCHPRODUCT command syntax
Where:

The operands are as follows:

**CONTINUE** Specify the CONTINUE operand without any value to notify
DFSMSSrmm SEARCH subcommand processing that you want to
break down the search results based on the LIMIT value and
request that DFSMSrmm return the search continue information for
use with the next command. For TSO, the continue information is
returned either as a REXX variable or as a linemode message.
When the subcommand is issued from the DFSMSrmm API, the
continuation information might be either a linemode message or an
SFI or XML attribute.
CONTINUE is an optional operand.

Use the LIMIT operand to control the maximum number of entries
to be returned each time you start or continue the search.
To continue a previous search subcommand, the CONTINUE
operand value includes the following values to identify the current
search position:

**NUMBER**  
*product_number* is one to eight characters enclosed in single
quotation marks if it contains any special characters, or blank.

**LEVEL**  
*version_number* is one to nine characters enclosed in single
quotation marks if it contains any special characters or blank.

**SEARCHRACK command syntax**

In Figure 8-61 you can see the use of the new SEARCHRACK operand CONTINUE.

```
+-------------------------------+<-
| SEARCHRACK                    |
| | SR                          |
| +-------------------------------+---
| CONTINUE(------------------------) |-
| RACK-(-rack_number-)-MEDIANAME-(-media-) |
```

*Figure 8-61 SEARCHRACK command syntax*

Where:

The operands are as follows:

**CONTINUE** Specify the CONTINUE operand without any value to notify
DFSMSSrmm SEARCH subcommand processing that you want to
break down the search results based on the LIMIT value and
request that DFSMSrmm return the search continue information for
use with the next command. For TSO, the continue information is
returned either as a REXX variable or as a linemode message.
When the subcommand is issued from the DFSMSrmm API, the
continuation information might be either a linemode message or an
SFI or XML attribute.

CONTINUE is an optional operand.
Use the LIMIT operand to control the maximum number of entries to be returned each time you start or continue the search.
To continue a previous search subcommand, the CONTINUE operand value includes the following values to identify the current search position:

- **RACK**
  - `rack_number` is one to six characters enclosed in single quotation marks if it contains any special characters, or blank.

- **MEDIANAME**
  - `media` is one to eight characters enclosed in single quotation marks if it contains any special characters, or blank.

**SEARCHVOLUME command syntax**

In Figure 8-62 you can see the use of the new SEARCHVOLUME operand CONTINUE.

![Figure 8-62 SEARCHVOLUME command syntax](image)

Where:

The operands are as follows:

- **CONTINUE**
  - Specify the CONTINUE operand without any value to notify DFSMSrmm SEARCH subcommand processing that you want to break down the search results based on the LIMIT value and request that DFSMSrmm return the search continue information for use with the next command. For TSO, the continue information is returned either as a REXX variable or as a linemode message. When the subcommand is issued from the DFSMSrmm API, the continuation information might be either a linemode message or an SFI or XML attribute.

  CONTINUE is an optional operand.

  Use the LIMIT operand to control the maximum number of entries to be returned each time you start or continue the search.

  To continue a previous search subcommand, the CONTINUE operand value includes the following values to identify the current search position:

- **VOLUME**
  - `volser` is one to six characters enclosed in single quotation marks if it contains any special characters, or blank.
SEARCHVRS command syntax

In Figure 8-63 you can see the use of the new SEARCHVRS operand CONTINUE.

Where:

The operands are as follows:

- **CONTINUE**: Specify the CONTINUE operand without any value to notify DFSMSrmm SEARCH subcommand processing that you want to break down the search results based on the LIMIT value and request that DFSMSrmm return the search continue information for use with the next command. For TSO, the continue information is returned either as a REXX variable or as a linemode message. When the subcommand is issued from the DFSMSrmm API, the continuation information might be either a linemode message or an SFI or XML attribute.

   CONTINUE is an optional operand.

   Use the LIMIT operand to control the maximum number of entries to be returned each time you start or continue the search.

   To continue a previous search subcommand, the CONTINUE operand value includes the following values to identify the current search position:

   - **DSNAME**: dataset_mask is one to 44 characters enclosed in single quotation marks if it contains any special characters, or blank.
   - **JOBNAME**: jobname_mask is one to eight characters enclosed in single quotation marks if it contains any special characters, or blank.
   - **VOLUME**: volser_mask is one to six characters enclosed in single quotation marks if it contains any special characters, or blank.
   - **NAME**: vrs_name is one to eight characters enclosed in single quotation marks if it contains any special characters, or blank.

In Figure 8-64 we see a typical SEARCH example of using the CONTINUE parameter. If you issue a search command from TSO environment, you get a table of resources returned and at the end of this table, there will be messages:

- EDG3203I SEARCH COMPLETE - MORE ENTRIES MAY EXIST
- EDG3012I xxxx ENTRIES LISTED
If you specify the CONTINUE parameter, there will be a new additional message now:

\> EDG3025I continue_information

You can use this information to build a new search command, pulling in the next RMM resources in sequence.

```plaintext
//TESTCASE EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*  
//RMMCLIST DD DSN=&SYSUID..CLIST.DATA2,DISP=(MOD,KEEP),
//         UNIT=SYSDA,SPACE=(TRK,(1,1),RLSE)
//SYSTSIN DD *
RMM SV OWNER(*) CLIST(' 1 - ',) LIMIT(10) CONTINUE START
*comment : next run the clist below using the highest volser# from the display*
RMM SV OWNER(*) CLIST(' 2 - ',) LIMIT(10) CONTINUE(VOLUME(CART04)) ADD /
/*

Figure 8-64  Example use of CONTINUE

Where:

The operands are as follows:

- **CONTINUE** To continue a previous search subcommand
- **VOLUME(CART04)** The last volume serial number of the previously search result you got in message EDG3025I.

Figure 8-65 shows you the result of the first SEARCHVOLUME command we used, and you can see the continuation information in message EDG3025I.

```plaintext
RMM SV OWNER(*) CLIST(' 1 - ',) CONTINUE START

<table>
<thead>
<tr>
<th>Volume Owner</th>
<th>Rack</th>
<th>Assigned date</th>
<th>Expiration date</th>
<th>Location</th>
<th>Dsets</th>
<th>St</th>
<th>Act</th>
<th>Dest.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CART0A</td>
<td>PAOLOR6</td>
<td>2008/066</td>
<td>2008/072</td>
<td>SHELF</td>
<td>5</td>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CART0B</td>
<td>PAOLOR6</td>
<td>2008/066</td>
<td>2008/072</td>
<td>SHELF</td>
<td>5</td>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CART0C</td>
<td>PAOLOR6</td>
<td>2008/066</td>
<td>2008/072</td>
<td>SHELF</td>
<td>11</td>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CART0D</td>
<td>PAOLOR6</td>
<td>2008/066</td>
<td>2008/072</td>
<td>SHELF</td>
<td>11</td>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CART0E</td>
<td>PAOLOR6</td>
<td>2008/066</td>
<td>2008/072</td>
<td>SHELF</td>
<td>5</td>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CART0F</td>
<td>PAOLOR6</td>
<td>2008/066</td>
<td>2008/072</td>
<td>SHELF</td>
<td>5</td>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CART01</td>
<td></td>
<td>2008/070</td>
<td></td>
<td>SHELF</td>
<td>1</td>
<td>S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CART02</td>
<td></td>
<td>2008/070</td>
<td></td>
<td>SHELF</td>
<td>1</td>
<td>S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CART03</td>
<td></td>
<td>2008/070</td>
<td></td>
<td>SHELF</td>
<td>1</td>
<td>S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CART04</td>
<td></td>
<td>2008/070</td>
<td></td>
<td>SHELF</td>
<td>1</td>
<td>S</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
EDG3203I SEARCH COMPLETE - MORE ENTRIES MAY EXIST
EDG3012I 10  ENTRIES LISTED
EDG3025I VOLUME('CART04')
READY
```

Figure 8-65  Result of a SEARCHVOLUME using the CONTINUE operand
REXX and API considerations

When you issue a search command from REXX and you request results as REXX variables, the EDG@CONT variable contains the character string you should use to continue the subcommand. When you use the API, and data is requested back as SFI, the CONT SFI contains the character string you should use. This is shown in Figure 8-66.

![Figure 8-66  REXX and API search](image)

In Figure 8-67 you can see how easy it is to use the new EDG@CONT variable in a REXX exec to process a large number of records. In previous releases of DFSMSrmm, it was not always possible to search for all information stored in the DFSMSrmm control data set depending on the storage limitation. In this case you always got one of the messages EDG3921I or EDG3025I “Insufficient storage for search processing. More records might exist” and the function ended before all information are processed.

```rexx
/* REXX */
... 
  address "TSO" "RMM SV OWNER(*) LIMIT(1) CONTINUE"
  cnt = cnt + edg@vol.0
  call processing
  do while EDG$CONT ^= ''
    address "TSO""RMM SV OWNER(*) LIMIT(500) CONTINUE("EDG$CONT")"
    cnt = cnt + edg@vol.0
    do I = 1 to EDG$VOL.0
      call processing
    end
    say ' number of records processed: ' cnt
  end
  exit
processing:
... 
return
```

![Figure 8-67  Sample REXX exec using EDG@CONT](image)

When you use the High Level Language API or the RMM Web Service and data is requested back as XML, you will find message EDG3025I, containing the continue_information, in the group at the end of the returned resources as follows:

```xml
</VOLUME>
<INFO>
```
Any continue_information you enter is handled as is by the RMM command processor. The case you specify is retained and used by RMM, so that CONTINUE(VOLUME('ab1000')) and CONTINUE(VOLUME('AB1000')) are different continuation points as shown in Figure 8-68. RMM does not change any values in the CONTINUE operand to upper case.

The last command in our example requests 1000 resources, but only 27 are returned. This means that there are no more resources to return. In this case there is NO continue_information returned. No EDG3025I is issued and EDG@CONT will also be blank.

### ISPF considerations

You will see no change in the ISPF panels for use of the CONTINUE parameter but under the covers the search execs will exploit this new feature. When the LIMIT value is either set to * or is larger than 2000, the execs will use CONTINUE as shown in Figure 8-69.

The EDGRxSCH execs will internally handle the panel request for a limit >2000 and specify LIMIT on the SEARCHxxxx subcommand accordingly, using LIMIT(2000) until the last SEARCH when LIMIT is set to panel_specified_limit-(n*2000).

![Figure 8-68 Case sensitive CONTINUE](image-url)

#### Note: RMM returns continue_information values delimited with quotes.

![Figure 8-69 LIMIT larger than 2000](image-url)
8.6.7 STORAGEGROUP parameter

For the SEARCHVOLUME command the new STORAGEGROUP parameter was introduced.

▶ RMM SEARCHVOLUME STORAGEGROUP(storage_group_name)
  – You can specify the storage group name to select a subset of volumes based on the assigned storage group name.
  – A storage group name is one-to-eight alphanumeric characters.
  – A storage group name can be a value that matches to a VLPOOL NAME value but does not need to be defined on a VLPOOL definition.
  – DFSMSrmm accepts the abbreviation STORGRP.
  – Use the STORAGEGROUP operand to build lists of exportable volumes that are in the same VTS physical volume pool.

In Figure 8-70 you can see the use of the new SEARCHVRS operand CONTINUE.

```
SEARCHVOLUME
   / SV
   | STORAGEGROUP(=storage_group_name=)
   | STRGRP
```

**Figure 8-70  STORAGEGROUP command syntax**

Where:

STORAGEGROUP Specifies the storage group name in order to select a subset of volumes based on the assigned storage group name. A storage group name is one to eight alphanumeric characters. A storage group name can be a value that matches to a VLPOOL NAME value, but does not need to be defined on a VLPOOL definition.

8.7 REPORT17 of EDGRRPTE REXX exec

When a stacked volume, containing exported logical volumes, is ejected from the library, as the logical volumes expire, RMM places the volumes in a “pending release” state and then when the logical volumes are imported into the library, RMM completes the return to scratch process enabling the volumes to be reused. As the exported logical volumes expire, you need the ability to do off-site stacked volume management so you can determine when to bring a stacked volume back on-site for possible reuse. RMM has enough information in its database for you to create and run reports, however, a specific stacked volume management report did not exist.

With z/OS V1.9, RMM provides a new stacked volume management report for customers to customize and run, that includes the ability to report on the percentage of active data on a stacked volume and to also report on the percentage of active logical volumes on a stacked
volume. A new report, REPORT17, is added to EDGRRPTE reporting exec which summarizes information for logical and stacked volumes. The stacked volumes are presented in order of increasing percentage of active number of volumes and percentage used. The least used stacked volumes are listed first. Figure 8-71 shows an example of a REPORT17 report.

Figure 8-71   REPORT17 printout

The report columns are explained as follows:

- **% Act**: Percentage of the contained logical volumes which are active.
- **# Active**: The number of active logical volumes. Active logical volumes are all those which are neither scratch nor pending release.
- **# Logical**: The number of contained logical volumes.
- **% Use**: The approximate percentage of active data.
- **Capacity**: The size of the stacked volume in MB.
- **Retention Date**: When VRS retained this is the VRS calculated retention date otherwise it is the latest expiration date of all contained active volumes.
- **Export Date**: The date when the stacked volume was exported from a VTS.
- **Export Time**: The time when the stacked volumes was exported from a VTS.

### 8.8 3592 Model E05 software support

Using the existing media types (MEDIA5, MEDIA6, MEDIA7, and MEDIA8) and the two extended length future media types (MEDIA9 and MEDIA10), an encryption enabled 3592 Model E05, reads and writes using the new Enterprise Encrypted Format 2 (EEFMT2) recording technology. It can also read and write using the Enterprise Format 1 (EFMT1) and Enterprise Format 2 (EFMT2) non-encrypted recording technologies.

In order to request EEFMT2 in the stand-alone and in the system-managed IBM tape library environment, a DFSMS dataclass must be used which specifies EEFMT2 as its recording technology, otherwise EFMT2 is the default recording technology that is used. Dataclass can also be used to request the EFMT1 recording format and to explicitly request the EFMT2 format. A mix of recording formats is not supported on the same tape cartridge. An enhanced 3592 Model E05 that does not have the encryption feature enabled, can only read and write in the non-encryption formats (EFMT1 and EFMT2) which is the same as the base 3592 Model E05. In a mixed 3592 environment, new microcode is also needed for the 3592 Model J and the base 3592 Model E05 so that it recognizes a volume with the new EEFMT2 recording technology.
8.8.1 Change summary

There is a new recording format external for encrypted media (EEFMT2).

- Enterprise encrypted format 2.

Encryption requested through the DFSMS data class:

- Through specification of the new recording format EEFMT2 (EE2).
- If the encrypted format (EEFMT2) is not specified, the non-encrypted formats EFMT1 or EFMT2 are used.

Encryption is supported with all existing 3592 media types.

- MEDIA5 – MEDIA10 (including WORM media).

Existing data class options, performance scaling, and performance segmentation is also supported with encryption.

Figure 8-72 shows where you specify EEFMT2 in the DFSMS data class.

![Figure 8-72 EEFMT2 in recording technology field](image)

8.8.2 Migration and coexistence considerations

The following APARS must be researched and installed on the relevant systems.

- **RMM preconditioning** APAR OA16523 (V1R6 – V1.8)
- **RMM toleration** APAR OA16524 (V1R4 – V1.8)
RMM tape encryption  APAR OA15698 (V1R6 – V1.7)
   APAR OA17574 (V1.8)
   - The toleration APAR OA16524 is based on OA15623 (preconditioning).
   - The tape encryption APARs OA15698 and OA17574 are based on OA16524 (toleration).
   - For the installation on a client/server, RMMplex is mandatory:
     a. Install pre-conditioning PTF on all systems of the RMMplex.
     b. Then install the toleration PTF on all systems.
     c. ONLY then can the RMM tape encryption PTF be installed.
   - Pre-Conditioning and Tolerance APARs contain a ++HOLD(MULTSYS) text, which describes this dependency.

8.8.3 Label anomaly processing

APAR OA18455 was created to prevent DFSMSrmm from setting the PENDING RELEASE ACTION INIT flag, when a “servo track format error” is detected. The tape is not blocked anymore by DFSMSrmm, with the message as shown in Figure 8-73.

EDG4033I VOLUME xxxxxx REJECTED. THE VOLUME IS WAITING TO BE REINITIALIZED.

Figure 8-73  EDG4033I Message

8.9 Common Information Model (CIM) provider

The RMM CIM provider is the link for a customer to obtain real-time RMM data within a Common Information Model (CIM) environment. This functionality will enable a CIM client such as a PC to obtain RMM data.

In z/OS V1.8 DFSMSrmm shipped a Common Information Model (CIM) Provider for use with the OpenPegasus CIMOM. The provider supported a subset of the resources managed by DFSMSrmm. The CIM model supported by DFSMSrmm is further extended in this release to cover all of the resources managed by DFSMSrmm. The CIM provider is updated to support out-of-process mode functionality added to OpenPegasus CIM Server with 2.5.1. This has an impact on the way that the DFSMSrmm provider code is installed and run.

In addition, the RMM extensions to the CIM object model are based on the latest CIM V2.11 specification. To further enable selection of resources via the API the SEARCH subcommands are all now capable of specifying continuation, enabling “chunking” of returned entries.

8.9.1 RMM CIM classes

Four new abstract super-classes have been introduced as an intermediate layer between native CIM schema classes and RMM classes. These abstract classes hold a common set of keys for their child classes. Abstract classes cannot have instances. Derived child classes can have instances.
Four new abstract super-classes have been introduced as an intermediate layer between native CIM schema classes and RMM classes. These abstract classes hold a common set of keys for their child classes. Abstract classes can not have instances, but derived child classes can have.

The DFSMSrmm CIM interface has been completed by implementing the remaining resources, not covered in V1.8, which are:

- IBMrmm_Product
- IBMrmm_PolicyRule
- IBMrmm_Control

The following CIM classes are new in z/OS V1.9:

- IBMrmm_LogicalMedia (abstract class)
- IBMrmm_PhysicalMedia (abstract class)
- IBMrmm_StorageMediaLocation (abstract class)
- IBMrmm_Identity (abstract class)
- IBMrmm_PhysicalVolume
- IBMrmm_LogicalVolume
- IBMrmm_Dataset
- IBMrmm_Owner
- IBMrmm_Location
- IBMrmm_ShelfLocation
- IBMrmm_Product
- IBMrmm_PolicyRule
- IBMrmm_Control

### 8.9.2 CIM association classes

All aspects of a volume to a location or shelf location are now modelled by various new association classes as follows:

- IBMrmm_PhysicalLogicalVolume (association 1:1)
- IBMrmm_LogicalVolumeDataset (association 1:N)
- IBMrmm_LogicalVolumeOwner (association N:1)
- IBMrmm_DatasetOwner (association)
- IBMrmm_PhysicalVolumeCurrentLocation (association N:1)
- IBMrmm_PhysicalVolumeDestinationLocation (association N:1)
- IBMrmm_PhysicalVolumeHomeLocation (association N:1)
- IBMrmm_PhysicalVolumeLoanLocation (association N:1)
- IBMrmm_PhysicalVolumeOldLocation (association N:1)
- IBMrmm_PhysicalVolumeRequiredLocation (association N:1)
- IBMrmm_PhysicalVolumeCurrentShelfLocation (association 1:1)
- IBMrmm_PhysicalVolumeDestinationShelfLocation (association 1:1)
- IBMrmm_PhysicalVolumeOldShelfLocation (association 1:1)

An example being traversing IBMrmm_PhysicalVolumeDestinationLocation left-to-right will return the destination location of a volume, which moves because of an inventory management run. Vice versa, when traversing right-to-left, all volumes are listed, that are intended to move into a specific location.

Volume chains and PolicyRule (VRS) chains are also supported with release V1.9 as follows:

- IBMrmm_LogicalVolumeChainedLogicalVolume (association 1:1)
- IBMrmm_LogicalVolumeLogicalVolumeInChain (association 1:N)
- IBMrmm_LocationShelfLocation (association 1:N)
- IBMrmm_ProductLogicalVolume (association 1:N)
The association IBMrmm_LogicalVolumeChainedLogicalVolume lets you traverse through a chain of volumes, back and forth. IBMrmm_LogicalVolumeLogicalVolumeInChain returns the whole set for a given chain member. The same is true for IBMrmm_PolicyRulePolicyRuleInChain.


**Note:** A policy rule can have more than one previous rule, hence the N:1 relationship.

### 8.9.3 Supported CIM operations

A new method `invokeMethod()` is introduced from the MethodProvider2 interface as shown in Figure 8-74. Search is the only supported method name for the `invokeMethod` function. It returns a list of objects from DFSMSrmm by name. It is particularly useful when working with the CONTINUE operand for search requests. Only the number of objects that are actually returned to the client are specified by the LIMIT operand. The client is able to get the list incrementally, in an interactive way.

**Table:**

<table>
<thead>
<tr>
<th>CIM Class</th>
<th>CIM Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main classes</td>
<td></td>
</tr>
<tr>
<td><code>enumerateInstances</code></td>
<td>ei</td>
</tr>
<tr>
<td><code>enumerateInstanceNames</code></td>
<td>eim</td>
</tr>
<tr>
<td><code>getInstance</code></td>
<td>gi</td>
</tr>
<tr>
<td><code>createInstance</code></td>
<td>ci</td>
</tr>
<tr>
<td><code>modifyInstance</code></td>
<td>mi</td>
</tr>
<tr>
<td><code>deleteInstance</code></td>
<td>di</td>
</tr>
<tr>
<td><code>invokeMethod</code></td>
<td>im</td>
</tr>
</tbody>
</table>

**Figure 8-74**  CIM `invokeMethod`

### 8.9.4 Search operands

Whereas in z/OS V1.8 the IBMrmm_SearchOperands class has no special importance, it gains importance in z/OS V1.9 in conjunction with the CONTINUE operand. The CONTINUE operand is added to the search string within the instances of IBMrmm_SearchOperands.
The CIM search method (enumerateInstanceNames) does not allow passing the search operand. Therefore, any operands are set (1) before in the corresponding instance of the class IBMrmm_SearchOperands, which is later used during the search operation (2) as shown in Figure 8-75.

To make use of the CONTINUE operand, you have to add it to an instance of IBMrmm_SearchOperands.

The V1.9 CIM provider fully supports the new CONTINUE operand, to divide a huge set of returned volumes or data sets into multiple sub-sets. You can either specify the CONTINUE operand in the appropriate instance of IBMrmm_SearchOperands or use the new API method invokeMethod(). Here we show an example of listing all volumes from DFSMSrmm in groups of ten, as follows:

- Point 1 lets the provider incrementally pull chunks of data from RMM, but finally returns the whole matching set to the client, as follows in wbemcli mii:
  
  http://<userid>:<password>@<cimon_uri>:<port> /root/cimv2:IBMrmm_SearchOperands

- Point 2 passes the concept of chunking down to the client, and the client has to keep requesting data until the set is complete and no more data is available, as follows in wbemcli ein:
  
  http://<userid>:<password>@<cimon_uri>:<port> /root/cimv2:IBMrmm_LogicalVolume

![Figure 8-75  CIM search operands](image)

### 8.9.5 Software dependencies

A fully functional Pegasus CIM Server is a fundamental prerequisite for the RMM CIM-Provider.

- For z/OS the Pegasus CIM Server is preinstalled in /usr/lpp/wbem and must be invoked as UNIX System Service (USS) process. Refer to the *eServer zSeries Common Information Model User's Guide*, SC33-7998 for how to set up the CIM server, particularly the necessary RACF settings for the CIM server and client user IDs.
For Linux®, the provider files has to be downloaded from /usr/lpp/dfsms/rmm to the Linux system. The Pegasus CIM Server 2.5.3 has to be downloaded from the following Web site:

http://www.openpegasus.org

The minimum supported Java™ level is 1.4.2. The designated Java libraries are required to be added to the CLASSPATH. Refer to Figure 8-76 for a list of these libraries. For how to set up the CLASSPATH, refer to the “Exports” sections in the readme file rmmcim.txt.

<table>
<thead>
<tr>
<th>Library name</th>
<th>Version</th>
<th>Download location</th>
</tr>
</thead>
<tbody>
<tr>
<td>j2ee.jar</td>
<td>1.3.1</td>
<td><a href="http://java.sun.com/j2ee/index.jsp">http://java.sun.com/j2ee/index.jsp</a></td>
</tr>
<tr>
<td>soap.jar</td>
<td>2.3.1</td>
<td><a href="http://apache.mpi.c.uk/ws/soap/version-2.3.1/">http://apache.mpi.c.uk/ws/soap/version-2.3.1/</a></td>
</tr>
<tr>
<td>mail.jar</td>
<td>1.3.1</td>
<td><a href="http://java.sun.com/products/javamail/downloads/index.html">http://java.sun.com/products/javamail/downloads/index.html</a></td>
</tr>
<tr>
<td>xerces-2.4.0.jar</td>
<td>2.4.0</td>
<td><a href="http://www.ibiblio.org/maven/xerces/jars/">http://www.ibiblio.org/maven/xerces/jars/</a></td>
</tr>
<tr>
<td>log4j-1.2.8.jar</td>
<td>1.2.8</td>
<td><a href="http://logging.apache.org/log4j/docs/download.html">http://logging.apache.org/log4j/docs/download.html</a></td>
</tr>
<tr>
<td>uddi4j.jar</td>
<td>2.0</td>
<td><a href="http://sourceforge.net/projects/uddi4j">http://sourceforge.net/projects/uddi4j</a></td>
</tr>
</tbody>
</table>

Figure 8-76  CLASSPATH - Java libraries

8.9.6 CIM installation

Use script rmmutil.sh or alternatively, load CIM classes and register providers manually by invoking from the shell:

- cimmo -i. -n root/cimv2 rmmcimp.mof
- z/OS: cimmofl -i. -n root/PG_InterOp -R/var/wbem rmmcimpr.mof
- Linux: cimmofl -i. -n root/PG_InterOp rmmcimpr.mof

Note: The “Miscellaneous Tests” menu offers various post-installation tests, which we highly recommended you execute, before actually working with the provider. For example, the Web service or direct API can be tested, to make sure the underlying communication channels work properly.

Set the environmental variables:

- Export RMMCIM_NAMESPACE=root/cimv2
- Export RMMCIM_CONFIG=/var/rmm/rmm.properties

Set external link (z/OS only):

- ln -e EDGXHCLL libEDGXHCLL.so (within $LIBPATH)

Note: The external link is necessary to be able to connect to the direct API C++ DLL EDGXHCLL, which actually resides in SYS1.SIEALNKE.
8.9.7 Migration and coexistence considerations

IBM WebSphere® Application Server (v5.02 or higher) and the z/OS CIM Server V1.9 has to be up and running on the target z/OS system in order to work with the RMM CIM provider V1.9. If running under Linux, OpenPegasus CIM Server 2.5.3 or above has to be installed. The RMM CIM provider readme with setup instructions can be found under /usr/lpp/dfsms/rmm/rmmcim.txt within the UNIX System Services (USS).

In addition, the physical file system (PFS) after the First IPL, requires the following tasks:

The customer has to unregister the V1.8 CIM providers and unload all V1.8 CIM classes, by using the rmmutil1.sh tool. Following this, the complete set of V1.9 providers must be registered and the V1.9 CIM classes must be loaded, by using the same tool.

Note: Customers migrating from z/OS V1.7 should follow the migration steps documented for migration to V1.8 before completing the V1.9 registration steps.
NFS enhancements in DFSMS V1.9

In this section we discuss the NFS enhancements provided in z/OS V1.9.

We cover the following topics:

- Network File Systems (NFS) enhancements
- 24-bit addressing relief
- Multi TCP/IP stack support
- Usage and invocation
- Addds operator command
- RACF data labeling
- NFS v4 client support
- Client attribute syntax
- Server Ctrace upgrade
9.1 Network File Systems (NFS) enhancements

Network File System (NFS) is a base element of z/OS, that allows remote access to z/OS host processor data from workstations, personal computers, or any other system on a TCP/IP network that is using client software for the Network File System protocol.

In the following sections we discuss each of the new enhancements to NFS.

9.2 24-bit addressing relief

Currently all z/OS NFS Server tasks that interact with z/OS MVS data sets have their stacks and heaps defined below the 16MB line. This puts major constraints on the number of z/OS NFS Server tasks that can exist.

In z/OS V1R9, NFS task structures have been modified to permit stacks and heaps above the 16 MB line. This will enable NFS server tasks to allow more parallel NFS request processing.

Note: There will be a performance improvement due to more parallel NFS request processing.

9.3 Multi TCP/IP stack support

Currently there is a restriction limiting the z/OS NFS RPCSEC security mechanism to only support a single IP address, limiting it to a single TCPIP stack. z/OS NFS Server V1R9 will now be able to successfully interact with multiple TCPIP Stacks, including VIPA (Virtual IP Assignment).

z/OS supports the ability to have multiple TCPIP stacks on a single system, including Virtual IP Assignment (VIPA). These multiple stacks and VIPA are useful for providing enhanced system reliability, providing alternate message paths should a given stack break.

The z/OS NFS RPCSEC security mechanism has been modified to allow it to successfully interact with multiple TCPIP Stacks.

9.4 Usage and invocation

To set up the multiple TCP/IP stack support, you must do the following tasks:

- The BPXPRMxx PARMLIB member must be updated to include all stacks:
  - One stack must be marked as DEFAULT.
- Portmapper/rpcbind must be defined as generic server:
  - No stack affinity is allowed.
- NFS server must be defined as a generic server:
  - No stack affinity is allowed.
- You must have a TCPIP profile defined for each stack.
- For Kerberos, because each stack has its own hostname and IP, a keytab must be created for each stack.
Here we show the CINET configuration in the BPXPRMxx PARMLIB member to start a single NFS server in a multi stack environment:

```plaintext
FILESYSTYPE TYPE(CINET) ENTRYPOINT(BPXTCINT)
NETWORK TYPE(CINET)
  DOMAINNAME(AF_INET)
  DOMAINNUMBER(2)
  MAXSOCKETS(64000)
  INADDRANYPORT(4901)
  INADDRANYCOUNT(100)
NETWORK TYPE(CINET)
  DOMAINNAME(AF_INET6) /* activate IPV6 */
  DOMAINNUMBER(19)
SUBFILESYSTYPE TYPE(CINET) NAME(TCPIPRX) ENTRYPOINT(EZBPFINI) DEFAULT
SUBFILESYSTYPE TYPE(CINET) NAME(TCPIPRY) ENTRYPOINT(EZBPFINI)
```

**Note:** Common Inet Sockets is intended to be used only if multiple network socket file systems (such as two TCP/IPs) are to be active at one time. There is a performance degradation with using Common Inet Sockets with just a single sockets physical file system.

### 9.5 Addds operator command

The new `addds` operator command will allow a z/OS System Programmer to specify a replacement for one of the NFS control data sets. This is useful in the event that one of these control data sets becomes unusable.

- Using the `addds` operator command, a z/OS system programmer can replace the following types of NFS control data sets:
  - Lock data sets
  - Mount handle database data sets
- The `addds` command requires the existing NFS control data set to be freed before this command is issued.
  - This can be achieved using the new `freeds` operator command.
- It is only possible to free the currently inactive data set of the MHDB, or LDB, pair. Therefore, if the active data set is the one to be freed, it is necessary to first swap the data set pair.
  - This can be accomplished with a new `swapmhdr`, or `swapp1d`, command. This command swaps the active and inactive data sets in the database. After this is done, it is then possible to free the previously active data set.

**Command syntax**

The syntax of the `addds` command is:

```plaintext
MODIFY mvsnfs,ADDDS=ddname(dsname)
```
Where:

- mvsnfs is the name of the procedure in the system PROCLIB that was used to start the server.
- ddbname is the ddbname of the NFS server control data set which is to be replaced.
  - The valid ddnames are FHDBASE, FHDBASE2, LDBASE and LDBASE2.
- dsnname is the name of the lock data set or mount handle data set to be enabled for use by the z/OS NFS server.

The syntax of the `freeds` command is:

```
MODIFY mvsnfs,FREEDS=ddname
```

Where:

- mvsnfs is the name of the procedure in the system PROCLIB that was used to start the server.
- ddbname is the ddbname of the NFS server control data set which is to be freed.
  - The valid ddnames are FHDBASE, FHDBASE2, LDBASE and LDBASE2.

The syntax of the swap commands is:

```
MODIFY mvsnfs,SWAPMHDB
```

Where:

- mvsnfs is the name of the procedure in the system PROCLIB that was used to start the server.

### 9.6 RACF data labeling

The z/OS NFS Server currently does not provide support for the RACF Data Labeling option MLNAMES (also known as Name-Hiding). z/OS V1R9 NFS Server will now provide this support, enabling active MVS data set names to be hidden from NFS users who do not have at least READ access to the data sets.

**MLNAMES overview**

Here we discuss several considerations for this option:

- When this option is active, MVS data set names will be hidden from NFS users who do not have at least READ access to the data sets. Therefore, it might change the contents of an MVS data set index list when requested via the `ls -l` command.
- The z/OS NFS server only supports this option in SAF or SAFEXP SECURITY mode.
- This function only applies to MVS data set access, not to z/OS UNIX file access.
- The name-hiding function can degrade system performance because it requires authorization checks for every object for which a non-SPECIAL user attempts to list the name.

This option is activated by RACF command:

```
SETROPTS MLNAMES
```

And it is deactivated by the RACF command:

```
SETROPTS NOMLNAMES
```
9.7 NFS v4 client support

Unlike previous z/OS releases, z/OS NFS Client V1R9 now provides support for the NFS v4 Protocol. It exploits the enhancements in the v4 protocol in the areas of Internet performance and cross-platform interoperability.

NFS V4 protocol

The Network File System (NFS) version 4 is a distributed file system protocol that owes its heritage to NFS protocol version 2 and 3. Unlike earlier versions, the NFS V4 protocol supports traditional file access while integrating support for file locking and the mount protocol. In addition, support for strong security, compound operations, client caching, and internationalization have been added. Attention has been applied to make NFS V4 operate well in an Internet environment.

The client application interface to the NFS Client does not change with the NFS version 4 protocol. It is the NFS Client's responsibility to convert the application requests to the appropriate V4 requests. From an NFS Client perspective, the functional characteristics of this protocol are partitioned into several major areas:

- Security
- Name space and pseudo file systems
- Client ID
- Locking
- Attribute

Restrictions

The z/OS NFS v4 Client in V1R9 does not currently include:

- RPCSEC_GSS Security
- Locking

The NFS Version 4 Protocol is an industry-wide standard. Therefore, the z/OS NFS V1R9 NFS Client, which is implementing that standard, should successfully communicate with any NFS Server (such as AIX/SUN) that adheres to that standard.

Because this standard requires continued support of the NFS Version 2 and Version 3 Protocols as well, the z/OS NFS V1R9 Client should also successfully communicate with any NFS V2/V3 Server.

Mount command

The NFS V4 protocol has been added to the `vers` parameter of the `mount` command:

```bash
mount -o vers=n
```

Where “n” specifies the NFS protocol version to be used:

- 2 or 3 for Linux.
- 2, 3 or 4 for others.
9.8 Client attribute syntax

A new public option and stringprep option has been added to the z/OS NFS client attribute syntax.

public Forces the use of the public file handle when connecting to the NFS server. This option is valid only during mount processing.

Note: The public keyword is valid only for the NFS version 4 protocol.

stringprep(Y/N) Specifies whether z/OS NFS Client is to enable or disable stringprep normalization. Stringprep normalization is the NFS version 4 internationalization function for converting inbound strings to UTF-8 format.

Note: The stringprep attribute default value is “N”.

9.9 Server Ctrace upgrade

The NFS Server Ctrace function has been upgraded to use the common underlying support functions with the Client. This upgrade enables the NFS Server to exploit all of the function enhancements which were implemented for the NFS Client and improves serviceability.

Enhancements with z/OS V1R9

The following enhancements have been made:

- The “MODIFY mvsnfs,FLUSHCTR” operator command has been removed. The function performed by this command will now be performed automatically when the Component Trace external writer is stopped. It will no longer be necessary to explicitly execute this command first to flush the buffer.

- NFS component trace status and active options can be displayed by using the MVS display command:
  - DISPLAY TRACE, COMP= mvsnfs (for the NFS server)
  - DISPLAY TRACE, COMP= mvsnfsc (for the NFS client)

Figure 9-1 shows the result of a DISPLAY TRACE command for the NFS server.

<table>
<thead>
<tr>
<th>DISPLAY TRACE,COMP=NFSMVS</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEE843I 13.24.41 TRACE DISPLAY 240</td>
</tr>
<tr>
<td>SYSTEM STATUS INFORMATION</td>
</tr>
<tr>
<td>ST=(ON,0256K,01536K) AS=ON BR=OFF EX=ON MO=OFF MT=(ON,064K)</td>
</tr>
<tr>
<td>COMPONENT MODE BUFFER HEAD SUBS</td>
</tr>
<tr>
<td>NFSMVS ON 0010M</td>
</tr>
<tr>
<td>ASIDS <em>NOT SUPPORTED</em></td>
</tr>
<tr>
<td>JOBNAMES <em>NOT SUPPORTED</em></td>
</tr>
<tr>
<td>OPTIONS FFDC INFO WARNING ERROR</td>
</tr>
<tr>
<td>WRITER <em>NONE</em></td>
</tr>
</tbody>
</table>

Figure 9-1  DISPLAY TRACE command
Four new trace record types have been added:

- **Buffer**  
  Buffer Management (i.e. BUFNODE)

- **Lock_Request**  
  Control Block Lock Requests

- **Lock_Result**  
  Control Block Lock Request results (lock granted, lock in use, error, etc.)

- **Lock_Release**  
  Control Block Lock Release

Trace command option values where added for the preceding new record types listed:

- **Msg**  
  Error, Warning, and Info Records

- **Wait**  
  Suspend and Resume record types

- **Queue**  
  Schedule and Dispatch record types

- **Lock**  
  Lock_Req, Lock_Result and Lock_Release record types

The ability to turn off individual options was added.

- An option can be turned off by preceding the option value with a minus sign (e.g. OPTIONS=(-GENERAL)).

- Options are processed from left to right, first processing all values to turn on options and then processing all values to turn off options. Thus all options except network can be turned on with the following options specification: OPTIONS=(ALL,-NETWORK).

- If an options value of “-ALL” is specified, the option revert back to a setting of MIN.

**Note:** A value of “-MIN” is invalid and will be ignored.

The trace buffer size can be specified either by the BUFSIZE value in the startup PARMLIB member, or it can continue to be specified via the “DSPS=” startup parameter. The new default BUFSIZE value of 10 MB also applies to the NFS Client.

**Note:** The size of the trace buffer might still not be altered after startup.

NFS component trace buffers captured in an MVS dump, or a Component Trace data set, can be viewed using the IPCS CTRACE command.

**Migration and coexistence considerations**

APAR OA18325 must be applied to z/OS V1R7 and R8. The APAR is for rollback of the NFS V1R9 support to allow server ctrace to start even if the BUFSIZE parameter is present. The BUFFER parameter is added to the CTRACE DEFINE macro and the BUFSIZE parameter in the CTINFSxx PARMLIB member is then ignored.
9.10 Terminal ID based restricted MVSLOGIN

When the z/OS NFS Server is used in SECURITY (SAF or SAFEXP) mode, it is necessary for users on NFS clients to issue an NFS Client Enabling Utility "MVSLOGIN" command from the NFS client system before they can access any files on the NFS Server. Normally, assuming the user has a valid z/OS userid and password, this is not a problem and will successfully provide the user with access to the z/OS system through NFS.

However, with the appropriate RACF configuration specifications, the z/OS NFS server also provides the ability to restrict MVSLOGINs based on an NFS client's IP address. In order to support this capability, the z/OS NFS server transforms an NFS client's IP address into an 8-byte character string, which is then used as the Terminal ID (termid) for that NFS Client. Each decimal number of the IP address is transformed into two hex digits. For example, the following IP addresses are transformed into hexadecimal numbers:

- 12.15.16.32 0C0F1020
- 9.157.161.12 099DA10C

To use this capability, the z/OS system administrator must do the following tasks:

- Activate the RACF class "TERMINAL" with this RACF command:
  
  SETROPTS CLASSACT(TERMINAL)

- Define the proper resource in the TERMINAL class with this RACF command:

  RDEFINE TERMINAL termid UACC(NONE)

- Grant permission to some users (for example, CIANKA and KEEGAN) from the NFS client with IP address 9.157.161.12 to successfully execute the MVSLOGIN NFS Client Enabling Utility with this RACF command.

  PERMIT 099DA10C CLASS(TERMINAL) ID(CIANKA KEEGAN) ACCESS(ALTER)

Note: If a termid value of 099DA10C is specified, then non-SPECIAL users on the NFS client with IP address 9.157.161.12 will not be able to execute the MVSLOGIN NFS Client Enabling Utility.
z/OS V1.10 DFSMS enhancements

This part of the book contains the enhancements introduced with z/OS V1.10 DFSMS.
DFSMSdfp V1.10 enhancements

In this chapter we discuss new and changed functions in DFSMSdfp.

We cover the following topics:
- DFSMS data class space override
- System determined block size set by data class
- New read-only variables in ACS routines
- Message enhancements
10.1 DFSMS data class space override

In DFSMS V1R10, it has become possible to override space options already set in different sources (such as JCL, AMS define statement or attributes set by using LIKE= in your allocation). The option is enabled through a new data class setting (where NO is the default):

OVERRIDE SPACE (YES or NO)

If you enable the new override option, you have to specify YES on the override space field in the data class and additionally specify allocation criteria on the data class used in the override. After the override is requested, you must fill in the following fields in the dataclass:

▸ AVGREC
▸ AVG VALUE
▸ PRIMARY
▸ SECONDARY

If you specify RECORG in the dataclass, you also have to specify a value in the cisize data field. If the RECORG is set to KS, then %freespace CI and %freespace CA must be specified.

See the updated data class panel shown in Figure 10-1.

The JCL space parameters and dynamic allocation parameters that can be overwritten are:

▸ Space type (CYL, TRK, block length, record length or AVREC)
▸ Primary and secondary quantity
▸ # of directory blocks
In relation to IDCAMS, the following space attributes can be overwritten:

- Primary and secondary quantity (tracks/cylinders/kilobytes/megabytes/records)
- Controlinterval size
- Freespace (CI-/CA-percent)

**Note:** The space information needed for the overwrite must all be included in the dataclass, otherwise the allocation will fail or produce unexpected results.

**Migration considerations**

The new override option could be useful for changing an extensive number of fixed allocations in your current JCL, without having to change each individual job/allocation. Consider also using the feature for one-time needs or for returning varying capacity needs.

For selecting the candidate allocations, you might consider using the new read-only variables described later in this chapter.

### 10.2 System determined block size set by data class

Previously, system determined block size was only set, when the user-specified block size was set to zero (=not specified). In DFSMS V1R10, a new option, *System Determined Blocksize* on the DFSMS data class panel, enables enforcing of system determined block size.

System Determined Blocksize can be set to either YES or NO on the data class panel shown in Figure 10-2.

<table>
<thead>
<tr>
<th>Command ===&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCDS Name . . . : SYS1.SMS.SCDS</td>
</tr>
<tr>
<td>Data Class Name : BIGHFS</td>
</tr>
<tr>
<td>To ALTER Data Class, Specify:</td>
</tr>
<tr>
<td>Data Set Name Type . . . . HFS (EXT, HFS, LIB, PDS, Large or blank)</td>
</tr>
<tr>
<td>If Ext ...................... (P=Preferred, R=Required or blank)</td>
</tr>
<tr>
<td>Extended Addressability . . N (Y or N)</td>
</tr>
<tr>
<td>Record Access Bias ....... S=System, U=User or blank)</td>
</tr>
<tr>
<td>Space Constraint Relief .... N (Y or N)</td>
</tr>
<tr>
<td>Reduce Space Up To (%) .... (0 to 99 or blank)</td>
</tr>
<tr>
<td>Dynamic Volume Count ..... (1 to 59 or blank)</td>
</tr>
<tr>
<td>Compaction ............... (Y, N, T, G or blank)</td>
</tr>
<tr>
<td>Spanned / Nonspanned ..... (S=Spanned, N=Nonspanned or blank)</td>
</tr>
<tr>
<td>System Managed Buffering ... (1K to 2048M or blank)</td>
</tr>
<tr>
<td><strong>System Determined Blocksize</strong> N (Y or N)</td>
</tr>
</tbody>
</table>

Use ENTER to Perform Verification; Use UP/DOWN Command to View other Panels; Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
Specifying NO on this panel will mean block size processing as in previous releases. Specifying YES will change the block size processing, as system determined block size processing will occur even if the user specified a block size.

Note: System determined block size processing will still not override a blocksize set from a program. Only block sizes set from a DD statement or from a dynamic allocation will be changed.

10.3 New read-only variables in ACS routines

DFSMS introduces two new read-only variables in the ACS routines that enable better control of the secondary allocation quantity. The variables specify the space type and the secondary quantity.

Here we describe these variables in more detail:

- **&space_type**
  - The variable holds the allocation unit, specified in JCL or IDCAMS. Valid units are:
    - Cyl & trk
    - K, M, U
    - BLK
    - Blank (=unspecified)

- **&second_qty**
  - This variable holds the user specified quantity on the secondary allocation (from JCL or IDCAMS).

The values in these new read-only variables are available at allocation processing time and can then be used for assigning a data class with the override option set (see Figure 10-3).

FILTLIST PE10VS INCLUDE(MHLRES5.PE10VS.**)
.. WHEN (&DSN EQ &PE10VS AND &SPACE_TYPE EQ 'TRK'
AND &SECOND_QTY EQ 1000) DO
  SET &DATACLAS = 'PE10VS'
  WRITE 'DATACLAS IS SET TO ' &DATACLAS
  WRITE 'DUE TO SPACETYPE AND SECONDARY QTY'
  EXIT CODE(0)
END

Figure 10-3 Example of data class assignment based on &space_type & second_qty

In this example, the ACS routines will detect when a user specifies a secondary allocation of 1000 tracks and can assign a data class other than the default with the override YES option set, allowing the allocation to be changed.
10.4 Message enhancements

DFSMS V1R10 introduces new informational messages related to storage constraints, deletion, and renaming of datasets. The messages are written to the joblog and hardcopy log.

The information is for the attention of the storage administrator and can prevent future storage constraints because they are now reported at an early stage.

There are numerous messages for deletion and renaming of datasets. Consequently, they are not listed here.

These are some constraint relief messages:

- IGD17286I
- IGD17287I
- IGD17288I
- IGD17289I
- IGD17291I
- IGD17292I

The message IGD17286I is shown here as an example of more detailed reporting. For the remaining messages, refer to MVS System Messages.

- IGD17286I SPACE CONSTRAINT RELIEF WAS USED TO ALLOCATE DATA SET dsname, var1, var2, var3,

  **Explanation**: Space could not be allocated for the data set in the normal manner. One or more of the following actions had to be taken:
  - The data set was spread over more than one volume.
  - The requested primary space amount was reduced by the percentage specified in the DATA CLASS.
  - More than 5 extents were used to allocate the requested space.

The text corresponding to var1, var2, and var3 is shown below and depending on the actions taken by SMS, one or more of these variants will be printed out on the second line of this message. In the message text:

*dsname*: The data set name

- var 1: DATA WAS SPREAD OVER MULTIPLE VOLUMES.
- var 2: THE 5 EXTENT LIMIT WAS RELAXED.
- var3: SPACE WAS REDUCED BY y%, where y is a percentage from 1 - 99.

**System action**: Processing continues.

**Application Programmer Response**: This is an informational message. No action is required.

**Source**: Data Facility Product (DFSMS)
PDSE enhancements in z/OS V1.0

In this chapter we describe the new commands provided to display statistics from the Buffer Management Facility (BMF) of PDSE. PDSE was enhanced in z/OS V1R6 to provide a restartable address space to allow recovery from PDSE related problems. After that enhancement, the SMSPDSE address space becomes the anchor environment, and the main PDSE activity runs under SMSPDSE1. This document assumes that the restartable address space has been implemented, in which case the system will be running the two started task address spaces, SMSPDSE and SMSPDSE1.

In z/OS releases after V1R6, the location of the buffers have progressively been moved as far as possible to 64-bit storage in the HiperSpace area.

Essential to optimal performance of all PDSE functions is the proper use of the PDSE buffering functions. Until the availability of these commands, it was necessary to use SMF to obtain the operational information. The SMF information is still collected, but the advantage of these new BMF display commands is that they can provide an instantaneous view of the situation.

We cover the following topics:

- BMF DISPLAY HSPSTATS command
- BMF Display VSTOR command
- Significance of the HSPSTATS
- Migration and coexistence considerations
11.1 BMF DISPLAY HSPSTATS command

There are several options on the DISPLAY SMS,PDSE,HSPSTATS or DISPLAY SMS,PDSE1,HSPSTATS command that can be selected depending on the granularity required on the output.

The new commands display statistical information about use of the Hi per Space area, hence the use of the keyword HSPSTATS in the commands. They are extensions to the existing DISPLAY SMS,PDSE commands.

The new versions of the command are:
- DISPLAY SMS,PDSE[1],HSPSTATS....
- DISPLAY SMS,PDSE[1],VSTOR

Because there are two PDSE address spaces, the user must select PDSE or PDSE1.

11.1.1 General DISPLAY HSPSTATS commands

In this section we illustrate the basic display commands.

In Example 11-1 we show the basic form of the Display HSPSTATS command. The example shown is for the standard SMSPDSE address space; the form for the restartable address space has PDSE replaced by PDSE1.

<table>
<thead>
<tr>
<th>DISPLAY SMS,PDSE,HSPSTATS</th>
</tr>
</thead>
</table>

*Figure 11-1   DISPLAY SMS,PDSE,HSPSTATS unqualified version of the command*

In Example 11-2 we show an example of output from the DISPLAY SMS,PDSE,HSPSTATS command. In this case, HiperSpace is not implemented.

<table>
<thead>
<tr>
<th>D SMS,PDSE,HSPSTATS</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGW048I PDSE HSPSTATS Start of Report(SMSPDSE ) 740</td>
</tr>
<tr>
<td>HiperSpace Size: 0 MB</td>
</tr>
<tr>
<td>LRUTime : 60 Seconds   LRUCycles: 15 Cycles</td>
</tr>
<tr>
<td>BMF Time interval 3600 Seconds</td>
</tr>
<tr>
<td>*** HiperSpace is Zero ***</td>
</tr>
<tr>
<td>*** No member caching ***</td>
</tr>
<tr>
<td>PDSE ANALYSIS   End of Report(SMSPDSE )</td>
</tr>
</tbody>
</table>

*Figure 11-2   Output from the DISPLAY SMS,PDSE,HSPSTATS command*

In Example 11-3 we show an example of output from the DISPLAY SMS,PDSE1,HSPSTATS command. In this case, HiperSpace is enabled and set to 256 MB. A number of data sets have been identified as being PDSEs.
To monitor ongoing operation, a summary of the SMSPDSE address space performance might be all that is required. In Example 11-4 we show a sample SUMMARY form of the command for the SMSPDSE address space. A similar command can be issued against the SMSPDSE1 address space. No additional operands can be used with the SUMMARY option.

In Example 11-5 we show the output from the DISPLAY SMS,PDSE,HSPSTATS,SUMMARY command for the SMSPDSE address space. As for the display in Example 11-2, there is no HiperSpace activity.

In Example 11-6 we show output from the DISPLAY SMS,PDSE1,HSPSTATS,SUMMARY command for the SMSPDSE1 address space. The output shows that HiperSpace is enabled, and summarizes information about the activity.
11.1.2 Tailored display HSPSTATS commands

After the general status of the PDSE function has been determined, it is possible to issue commands to concentrate on specific aspects of the PDSE environments, such as specific data sets and storage classes.

The DISPLAY HSPSTATS command can be limited by data set name, by storage class or UNMANAGED data sets, and a limit can be set on the number of data set names to display.

The MAXDSNS operand is useful to limit the number of lines of output, which could be very high on some systems.

Selection of UNMANAGED data sets

In Example 11-7 we show the command to select all the UNMANAGED data sets (those not managed by SMS). To select the equivalent entries for the SMSPDSE1 address, use PDSE1 in place of PDSE.

- The UNMANAGED option cannot be used at the same time as the STORCLAS option.
- The MAXDSNS option can be used with the UNMANAGED option.

D SMS,PDSE1,HSPSTATS,SUMMARY
IGW049I PDSE Summary Statistics Start of Report(SMSPDSE1) 751
HiperSpace Size: 256 MB
LRUTime : 50 Seconds   LRU Cycles: 200 Cycles
BMF Time interval 3600 Seconds
Number of PDSE datasets: 11
Number of PDSEs Eligible for caching: 11
Number of PDSEs Always in Cache: 0
Number of PDSEs Maybe in Cache: 3
Number of PDSEs Not Eligible for cache: 0
PDSE Summary Statistics  End of Report(SMSPDSE1)

Figure 11-6  Output from DISPLAY SMS,PDSE1,HSPSTATS,SUMMARY (for SMSPDSE1)

D SMS,PDSE,HSPSTATS,UNMANAGED
IGW048I PDSE HSPSTATS Start of Report(SMSPDSE ) 849
HiperSpace Size: 0 MB
LRUTime : 60 Seconds   LRU Cycles: 15 Cycles
BMF Time interval 3600 Seconds
*** HiperSpace is Zero ***
*** No member caching ***
PDSE ANALYSIS  End of Report(SMSPDSE )

Figure 11-7  Display SMS,PDSE,HSPSTATS,UNMANAGED

In Example 11-8 we show the output of HSPSTATS for unmanaged data sets in the SMSPDSE address space. This shows that SMSPDSE is not monitoring any UNMANAGED PDSE data sets.

D SMS,PDSE,HSPSTATS,UNMANAGED
IGW0481 PDSE HSPSTATS Start of Report(SMSPDSE ) 849
HiperSpace Size: 0 MB
LRUTime : 60 Seconds   LRU Cycles: 15 Cycles
BMF Time interval 3600 Seconds
*** HiperSpace is Zero ***
*** No member caching ***
PDSE ANALYSIS  End of Report(SMSPDSE )

Figure 11-8  Output from DISPLAY SMS,PDSE,HSPSTATS,UNMANAGED
In Example 11-9 we show the output of HSPSTATS for unmanaged data sets in the
SMSPDSE1 address space. The output shows almost all of the data sets that were listed in
Example 11-3. The remainder from that list are therefore SMS managed.

D SMS,PDSE1,HSPSTATS,UNMANAGED
IGW0481 PDSE HSPSTATS Start of Report(SMSPDSE1) 851
HiperSpace Size: 256 MB
LRUTime : 50 Seconds   LRU Cycles: 200 Cycles
BMF Time interval 3600 Seconds
---------data set name-----------------------Cache--Always-DoNot
Elig---Cache--Cache
DB9D9.SDSNLOAD                                 Y      N      N
TCP1P.SEZALOAD                                 Y      N      N
CBC.SCCNCMP                                     Y      N      N
CEE.SCEERUN2                                     Y      N      N
SYS1.SIEAMIGE                                    Y      N      N
CBC.SCLBDLL2                                     Y      N      N
SYS1.SHASLNKE                                    Y      N      N
SYS1.NFSLIBE                                     Y      N      N
SYS1.SCUIMG                                      Y      N      N
SYS1.SIEALNKE                                     Y      N      N
SYS1.SIEALNKE                                     Y      N      N
PDSE ANALYSIS  End of Report(SMSPDSE1)

Figure 11-9  Output from DISPLAY SMS,PDSE1,HSPSTATS,UNMANAGED

Selection of data sets by Storage Class (STORCLAS)
HSPSTATS can be selected based on the Storage Class of the data sets.

The STORCLAS option can be used at the same time as the MAXDSNS option

In Example 11-10 we show the command to get statistics for data sets in Storage Class
example. To get equivalent information for the PDSE1 address space, use PDSE1 in place of
PDSE.

D SMS,PDSE,HSPSTATS,STORCLAS(example)
Figure 11-10  DISPLAY SMS,PDSE,HSPSTATS,STORCLAS(STANDARD)

In Example 11-11 we show the output from the D SMS,PDSE,HSPSTATS,STORCLAS(STANDARD)
command. Equivalent output for the SMSPDSE1 address space is obtained by using PDSE1 instead of PDSE.

D SMS,PDSE,HSPSTATS,STORCLAS(STANDARD)
IGW0481 PDSE HSPSTATS Start of Report(SMSPDSE ) 900
HiperSpace Size: 0 MB     Storage Class: STANDARD
LRUTime : 60 Seconds   LRU Cycles: 15 Cycles
BMF Time interval 3600 Seconds
*** HiperSpace is Zero ***
*** No member caching ***
PDSE ANALYSIS  End of Report(SMSPDSE )

Figure 11-11  Output from DISPLAY SMS,PDSE,HSPSTATS,STORCLAS(example)
Output from the DISPLAY SMS,PDSE1,HSPSTATS,STORCLAS(STD) command is shown in Example 11-12. In this case, the data set MHLRES5.PEMAIN.CNTL has been found.

```
D SMS,PDSE1,HSPSTATS,STORCLAS(STD)
IGW048I PDSE HSPSTATS Start of Report(SMSPDSE1) 902
HiperSpace Size: 256 MB Storage Class: STANDARD
LRUtime : 50 Seconds LRUcycles: 200 Cycles
BMF Time interval 3600 Seconds
----------data set name-------------------Cache--Always-DoNot
          Elig----Cache--Cache
MHLRES5.PEMAIN.CNTL                       Y   N   N
PDSE ANALYSIS  End of Report(SMSPDSE1)
```

Figure 11-12 Output from DISPLAY SMS,PDSE1,HSPSTATS,STORCLAS(example)

Selection of data sets by data set name (DSN)

HSPSTATS can be selected based on the name of the data sets.

The MAXDSNS option can be used with the DSN option.

The data set name can be specified as a fully qualified name or as a partially qualified name, but note that wild cards within any part of the name are not supported.

Note: The syntax for the DSN operand is not the same as that used with other components of the system - ISPF, for example. In ISPF 3.4 a data set prefix of SYS1 selects all data sets with high level qualifier SYS1. To obtain the same with the DSN operand, the name must be specified as DSN(SYS1.*).

Here are some examples of specifying data set names:

- IBMUSER.PDSE.DATASET (specific data set)
- IBMUSER.* (all PDSEs with the high level qualifier of IBMUSE)
- IBMUSER.PDSE.* (all PDSEs starting with IBMUSER.PDSE)

In Example 11-13 we show the command to display HSPSTATS for data set MHLRES5.PEMAIN.CNTL relating to SMSPDSE address space. To obtain equivalent data for the SMSPDSE1 address space, change PDSE to PDSE1. There is no need to specify MAXDSNS in this case.

```
DISPLAY SMS,PDSE,HSPSTATS,DSN(MHLRES5.PEMAIN.CNTL)
```

Figure 11-13 DISPLAY SMS,PDSE,HSPSTATS,DSN(MHLRES5.PEMAIN.CNTL)

Output from the DISPLAY SMS,PDSE,HSPSTATS,DSN(MHLRES5.PEMAIN.CNTL) is shown in Example 11-14. The output shows that SMSPDSE is not managing data set MHLRES5.PEMAIN.CNTL.
Chapter 11. PDSE enhancements in z/OS V1.0

Output from the DISPLAY SMS,PDSE1,HSPSTATS,DSN(MHLRES5.PEMAIN.CNTL) is shown in Example 11-15. This output shows that SMSPDSE1 is managing data set MHLRES5.PEMAIN.CNTL.

In Example 11-16 we show the command to display HSPSTATS information for all PDSE data sets starting with MHLRES5.PEMAIN relating to the SMSPDSE address space. To obtain the equivalent data from the SMSPDSE1 address change PDSE to PDSE1.

Output from the command DISPLAY SMS,PDSE,HSPSTATS,DSN(MHLRES5.PEMAIN.*) is shown in Example 11-17. The output shows that no data sets meeting that specification are being managed by SMSPDSE.
In Example 11-18 we show the D SMS,PDSE1,HSPSTATS,DSN(MHLRES5.PEMAIN.*) command output. Because there is only one data set that begins with MHLRES5.PEMAIN, the same data set as specifically selected in Example 11-13 is found.

```
D SMS,PDSE1,HSPSTATS,DSN(MHLRES5.PEMAIN.*)
IGW048I PDSE HSPSTATS Start of Report(SMSPDSE1) 013
HiperSpace Size: 256 MB
LRUTime : 50 Seconds LRUCycles: 200 Cycles
BMF Time interval 3600 Seconds
---------data set name-----------------------Cache--Always-DoNot
---------data set name-----------------------Cache--Cache
MHLRES5.PEMAIN.CNTL                             Y      N      N
PDSE ANALYSIS End of Report(SMSPDSE1)
```

**Figure 11-18 Output from DISPLAY SMS,PDSE1,HSPSTATS,DSN(MHLRES5.PEMAIN.*)**

In Example 11-19 we show the command to select all PDSE data sets associated with the SMSPDSE1 address space.

```
D SMS,PDSE1,HSPSTATS,DSN(sys1.*)
```

**Figure 11-19 DISPLAY SMS,PDSE1,HSPSTATS,PDSE1,DSN(SYS1.*)**

In Example 11-20 we show the output from the D SMS,PDSE1,HSPSTATS,DSN(SYS1.*) command. The output shows that five data sets are associated with SMSPDSE1.

```
D SMS,PDSE1,HSPSTATS,DSN(SYS1.*)
IGW048I PDSE HSPSTATS Start of Report(SMSPDSE1) 063
HiperSpace Size: 256 MB
LRUTime : 50 Seconds LRUCycles: 200 Cycles
BMF Time interval 3600 Seconds
---------data set name-----------------------Cache--Always-DoNot
---------data set name-----------------------Cache--Cache
SYS1.SIEAMIGE                                  Y      N      N
SYS1.SHASLNKE                                   Y      N      N
SYS1.NFSLIBE                                     Y      N      N
SYS1.SCUNIMG                                     Y      N      N
SYS1.SIEALNKE                                    Y      N      N
PDSE ANALYSIS End of Report(SMSPDSE1)
```

**Figure 11-20 Output from DISPLAY SMS,PDSE1,HSPSTATS,DSN(SYS1.*)**

In Example 11-21 we show the command to display HSPSTATS information for all PDSE data sets starting with SYS1 relating to the SMSPDSE1 address space, but this time limited to three data set names.

```
DISPLAY SMS,PDSE1,HSPSTATS,DSN(SYS1.*),MAXDSNS(3)
```

**Figure 11-21 DISPLAY SMS,PDSE1,HSPSTATS,DSN(SYS1.*),MAXDSNS(3)**
In Example 11-22 we show the D SMS,PDSE1,HSPSTATS,DSN(SYS1.*),MAXDSNS(3) command output. In this case, three data sets are listed.

<table>
<thead>
<tr>
<th>D SMS,PDSE1,HSPSTATS,DSN(SYS1.*),MAXDSNS(3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGW048I PDSE HSPSTATS Start of Report(SMSPDSE1) 084</td>
</tr>
<tr>
<td>HiperSpace Size: 256 MB</td>
</tr>
<tr>
<td>LRUTime: 50 Seconds LRU Cycles: 200 Cycles</td>
</tr>
<tr>
<td>BMF Time interval 3600 Seconds</td>
</tr>
<tr>
<td>---------data set name-------------------------Cache--Always-DoNot Elig--Cache--Cache</td>
</tr>
<tr>
<td>SYS1.SIEAMIGE</td>
</tr>
<tr>
<td>SYS1.SHASLNKE</td>
</tr>
<tr>
<td>SYS1.NFSLIBE</td>
</tr>
<tr>
<td>PDSE ANALYSIS End of Report(SMSPDSE1)</td>
</tr>
</tbody>
</table>

**Figure 11-22  Output from DISPLAY SMS,PDSE1,HSPSTATS,DSN(SYS1.*),MAZDSNS(3)**

### 11.2 BMF Display VSTOR command

The **Display SMS,PDSE,VSTOR** command output shows the current PDSE 64-bit directory buffer virtual storage utilization.

In Example 11-23 we show an example of the command to display the memory usage of the PDSE address space. A similar command using PDSE1 would display the information for the restartable address space.

<table>
<thead>
<tr>
<th>DISPLAY SMS,PDSE,VSTOR</th>
</tr>
</thead>
</table>

**Figure 11-23  DISPLAY SMS,PDSE,VSTOR command**

In Example 11-24 we show the output from the DISPLAY SMS,PDSE,VSTOR command.

In this case the output shows that there is storage allocated to the SMSPDSE address space even though SMSPDSE does not have HiperSpace enabled.

<table>
<thead>
<tr>
<th>D SMS,PDSE,VSTOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGW050I PDSE Virtual Storage Start of Report(SMSPDSE ) 766</td>
</tr>
<tr>
<td>Large Virtual Memory allocated for Address Space: 00004A00 Pages</td>
</tr>
<tr>
<td>Directory Storage Size: 2000 MB</td>
</tr>
<tr>
<td>Storage utilization in pages: 8</td>
</tr>
<tr>
<td>PDSE Virtual Storage End of Report(SMSPDSE )</td>
</tr>
</tbody>
</table>

**Figure 11-24  Output from DISPLAY SMS,PDSE,VSTOR command**
In Example 11-25 we show the output from the DISPLAY SMS,PDSE1,VSTOR command. In this case the output shows that the Storage Utilization is higher than in the case with SMSPDSE which is consistent with the fact that the data sets are being managed by SMSPDSE1.

<table>
<thead>
<tr>
<th>D SMS,PDSE1,VSTOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>IGW050I PDSE Virtual Storage Start of Report(SMSPDSE1) 787</td>
</tr>
<tr>
<td>Large Virtual Memory allocated for Address Space: 00005000 Pages</td>
</tr>
<tr>
<td>Directory Storage Size: 2000 MB</td>
</tr>
<tr>
<td>Storage utilization in pages: 603</td>
</tr>
<tr>
<td>PDSE Virtual Storage End of Report(SMSPDSE1)</td>
</tr>
</tbody>
</table>

Figure 11-25  Output from DISPLAY SMS,PDSE1,VSTOR command

11.3 Significance of the HSPSTATS

The performance of access to PDSE data sets is dependent on the cache tuning, which is dependent on the types of PDSE data sets that are being used. The characteristics of DATA members and MODULE members differ. The Display HSPSTATS commands provide information about the LRUtime, LRUcycles, and Hiper Space size which work together. Managing the values for these values is a balance between performance and CPU consumption.

There is a significant amount more data available in the SMF records produced by the system.

Significant changes to PDSE support were introduced with z/OS V1R6, and we recommend that you refer to Partitioned Data Set Extended Usage Guide, SG24-6106 for background information.

Further changes were made in z/OS V1R7 and z/OS V1R7, and we recommend that you refer to the PDSE sections in the following Redbooks publications:

- z/OS V1R7 DFSMS Technical Update, SG24-7225
- z/OS V1R8 DFSMS Technical Update, SG24-7435

The source code examples provided in the Redbooks publication, z/OS V1R8 DFSMS Technical Update, SG24-7435, if assembled and run under z/OS V1R10, provide displays of much of the other data.

11.4 Migration and coexistence considerations

There are no migration or coexistence considerations. The commands are not available on releases prior to z/OS V1R10.
DFSMSShsm V1.10 enhancements

In this chapter we discuss new and changed functions in DFSMSShsm.

We discuss the following topics:

- BACKDS enhancements: NEWNAME and ARCHBACK
- CDS backup improvements
- DFSMSShsm usage of virtual concurrent copy
- EAV: Extended Address Volume support
- RAS: TAPE COPY enhancement

The enhancements to the report generator in ISMF and the RMM ISPF application, which relate to DFSMSShsm, are not covered in this section. Refer to Chapter 17, “DFSMSRmm V1.10 enhancements” on page 195 for details.
12.1 BACKDS enhancements: NEWNAME and ARCHBACK

In z/OS V1.10, DFSMSShsm has had new keywords added on the BACKDS command to provide more flexibility when backing up individual datasets.

The NEWNAME parameter on the HBACKDS command basically enables you to create new copies of a given data set name and assign it to a specific data set name. You also can influence the metadata stored within DFSMSShsm by the additional keywords DATE and TIME. DFSMSdss is invoked to create the backup copy.

The ARCHBACK macro has been changed accordingly with the new keywords.

This function is particularly useful when migrating from an OEM backup tool to DFSMSShsm.

Previously, you would have to restore all backup copies from any data set involved to temporary data set names. After that, the process would be to rename the base data set name to a temporary name (making it temporarily unusable), rename the backup-copies with the temporary data set name one by one to the base data set name and do a backup, finally ending up renaming the base data set name back to its original name making it usable again. This was a quite time consuming and disruptive process that also might need to be done outside business hours, to make the data sets available for the rename process.

You will still have to recover the backup copies in scope to a temporary data set name, but by using the NEWNAME option in combination with DATE and TIME, now you can time stamp your backup copies of these data sets with the original backup date and time and thereby put them in the correct sequence, because DFSMSShsm will use the date stamp to do this. By specifying the base data set name in the NENAME parameter, the backup will be assigned to this base data set.

If DATE is specified without TIME, the current time will be connected to the backup version. If DATE is not specified, the current date will be used.

You can also use the new feature for point-in-time copies (year-end or disaster recovery issues). Just be sure to use a good naming convention, making it easy for you to revert to a certain version and to prevent the data set from expiring inadvertently (due to management class backup attributes).
We set up a scenario to test DFSMShsm’s new keyword NEWNAME. In this environment there is one specific data set with three backup copies of a data set with a management class allowing five backup copies (data set exists). These backup versions are shown in Figure 12-1.

```
<table>
<thead>
<tr>
<th>DSN</th>
<th>BACK FREQ</th>
<th>MAX VERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>MHLRES5.TSO.CNTL</td>
<td>***</td>
<td>***</td>
</tr>
</tbody>
</table>
```

Next we performed a backup of a data set that should be considered part of this backup sphere, using the command shown in Figure 12-2.

```
HBACKDS 'MHLRES5.TSO.CNTL.D990328.T173447' target(dasd)
newname('MHLRES5.tso.cntl') DATE(1999/03/28) TIME(173447)
```

The new backup version was added successfully as a version '0' in addition to the three existing versions (3, 4 and 5). Note also that the date-stamp and time-stamp were set as requested and put the backup version in the correct sequence. Also note the ‘NEWNM=YES’ field, indicating that the backup was created through an HBACKDS with the NEWNAME parameter (see Figure 12-3).
We next executed a new HBACKDS command to add an additional backup within the sequence of the existing backups due to the date- and timestamp added to the backup version through the DATE and TIME keywords (see Figure 12-4).

```
HBACKDS 'MHLRES5.TSO.CNTL.D080827.T180633' target(dasd)
   newname('MHLRES5.tso.cntl') DATE(2008/08/27) TIME(180633)
```

Figure 12-4  DFSMSShsm backup command

Note that version number on the new backup stays 0, but the sequence on this is in place because the generation number is '2' (see Figure 12-5).
Chapter 12. DFSMShsm V1.10 enhancements

12.1 Security considerations

To protect the new backup function using NEWNAME, two new STGADMIN profiles have been created:

- STGADMIN.ARC.ENDUSER.HBACKDS.NEWNAME
- STGADMIN.ARC.ENDUSER.BACKDS.NEWNAME
12.1.2 Backup version cleanup

To clean up backup versions, the HBDELETE command was updated with new DATE (yyyy/mm/dd) and TIME (HHMMSS) subparameters. These subparameters must be used to clean up backup versions created using the NEWNAME parameter. The VERSIONS subparameter is obsolete in combination with the DATE and TIME parameter. An example of BDELETE with the new DATA and TIME subparameters is shown in Figure 12-6.

```
hsend bdelete (mhlres5.tso.cntl) date(1999/03/28) time(173447)

ARC0182I 1 BACKUP VERSION(S) DELETED FOR MHLRES5.TSO.CNTL
COMMAND REQUEST 00000403 SENT TO DFSMSHSM
```

*Figure 12-6   Example of DFSMSshm delete of a backup version*

To perform a DFSMSHsm RECOVER from the appropriate version, use the DATE keyword and GENERATION in combination as shown in the sample HRECOVER in Figure 12-7.

```
hrecover 'mhlres5.tso.cntl' newname('mhlres5.tso.cntl.recover') date(08/08/27) gen(3)
```

*Figure 12-7   Sample HRECOVER command using DATE and GENERATION keywords*

**Note:** DFSMSHsm will do a consistency check executing the backup you request. If attributes are different or data sets have mismatched SMS status (SMS/NON-SMS), the backup will fail.

12.2 CDS backup improvements

In z/OS V1.10, DFSMSHsm starts using XCF to request exclusive enqueue on the ARCGPA/ARCCAT resource needed for backing up the control datasets. Other (long running) DFSMSHsm functions in the HSMplex might have enqueued these, even if they release their enqueues on certain intervals. See the manual *DFSMShsm Implementation and Customization Guide*, SC35-0418 for more details on these intervals.

In a CDSSHR=RLS environment, the request from the host backing up the CDSs means that each DFSMSHsm host will release the global enqueue ARCENQG/ARCCAT and wait for it to become available again. In a non-CDSSHR=RLS environment, each DFSMSHsm host will release ARCGPA/ARCCAT and wait for these resources to become available again.

The new process using XCF will reduce the delay that the CDS backup spent in earlier releases waiting for the backup to be able to start. Also, the potential risk that the backup will be hung while waiting for a long running task will now be smaller.

In a mixed environment with DFSMS V1.10 and pre-DFSMS V1.10 systems, the lower level systems need to be updated to be able to ignore the XCF request from DFSMS V1.10 systems performing the CDS backup. Pre-DFSMS V1.10 systems will then operate as in previous releases. This will also be the case in environments not supporting XCF.
12.3 DFSMSHsm usage of virtual concurrent copy

With DFSMSHsm V1.10 support for the different concurrent copy techniques has been added. The options are made available through a management class setting (Backup Copy Technique). This setting will be used when invoking DFSMSdss during automatic backup, volume backup or data set command backup.

These are the available concurrent copy supported options:

- CONCURRENT PREFERRED
- CONCURRENT REQUIRED
- STANDARD
- VIRTUAL PREFERRED
- VIRTUAL REQUIRED
- CACHE PREFERRED
- CACHE REQUIRED

The ISMF management class panel in Figure 12-8 shows these options. The values specified here influence the backup as to backup copy technique.

![Figure 12-8 MANAGEMENT CLASS ALTER panel](image-url)
12.4 EAV: Extended Address Volume support

DFSMShsm APAR OA22804 enables support for VCC keywords and support of RECALL/RECOVER on a pre-z/OS V1.10 system of an EAS data set with FMT 8 DSCB that was migrated or backed up on z/OS V1.10. Without this support, if you are using DFSMShsm to manage Extended Address Volume (EAV) volumes on z/OS V1.10, then on the downlevel system, if you attempt to recall or recover an EAS data set with a format 8 that was migrated or backed up on z/OS V1.10, the function will fail. This will directly impact RECALL, which is done implicitly. It will also impact RECYCLE and AUDIT.

For VCC, keywords specified in the management class will not be processed correctly. If VPREF or VREQ is specified in the management class, data sets will be backed up using standard I/O method without this coexistence APAR.

12.5 RAS: TAPE COPY enhancement

The unit name in Tape Copy Needed (TCN) records for internally generated tape copies is the generic unit name generated for input use instead of the original unit name specified for output. This can cause problems in a mixed tape technology environment because esoteric unit names can be used to control allocations to the different technology tape drives. The ‘SETSYS unit for TCN unit name’ RAS item satisfies the requirement to use the specified SETSYS unit name in the tape copy MWE built from the TCN record.
DFSMSdss V1.10 enhancements

In this chapter we discuss new and changed functions in DFSMSdss.

We cover the following topics:
- DUMP using NEWNAMEUNCONDITIONAL
- DEFRAG performance enhancement
- CONSOLIDATE command
13.1 DUMP using NEWNAMEUNCHIONDIONAL

Currently, a user of DFSMSdss can only rename a data set during COPY DATASET or RESTORE DATASET processing. This is because the new name of the data set applies to a data set that is allocated or overwritten by DFSMSdss. This behavior and design of the RENAME or RENAMEUNCHIONDIONAL keywords do not satisfy user requirements. Therefore DFSMSdss is adding the NEWNAMEUNCHIONDIONAL keyword, which can be specified during logical data set DUMP processing.

For this support, DFSMSdss will choose a data set name to be associated with the dumped data from a source data set based on rename filtering criteria. Using the NEWNAMEUNCHIONDIONAL keyword in dump processing, a user can specify source data sets along with corresponding new names to control the assigning of a new name.

The filtering criteria for the NEWNAMEUNCHIONDIONAL keyword is exactly the same as that for the RENAME and RENAMEUNCHIONDIONAL keywords. For example, a user can also specify how to choose a new name based on the high level qualifier. All data sets that match the new name criteria will be assigned a new name. Those that do not meet the filtering criteria will have their original name associated with the data. If a data set with the new name exists and is cataloged in the standard order of search, DFSMSdss will associate the catalog name and component names for the new name data set in the dump tape data set.

Otherwise, if the new name data set is not cataloged, or the new UFO field was set requesting that DSS generate the component names, DFSMSdss will set the new name and derive the component names based on how the names are derived for the existing RENAME and RENAMEUNCHIONDIONAL keywords. For example, DFSMSdss will append a .DATA (for a data component) and .INDEX (for an index component) and use the new cluster name when deriving component names, so as long as the length meets data set naming standards.

DFSMSdss today currently places the user catalog name of the data set that is being dumped in various fields within the dump tape data set header area. With this support, if the new name data set is a non-VSAM data set and is assigned a new name, if the new name is not cataloged in the standard order of search, DSS will insert the master catalog name in the user catalog fields in the dump tape data set header fields. For VSAM data sets, if the new name is not cataloged in the standard order of search then the source catalog name will be inserted into the dump tape data set header fields.

If the NEWNAMEUNCHIONDIONAL keyword is specified in conjunction with the SPHERE keyword, and a base VSAM cluster is being dumped, the new name keyword will affect the entire VSAM sphere. This means that DFSMSdss will perform new name processing on the entire VSAM sphere by placing a new name for associated alternate indexes and paths that exist. In order for DFSMSdss to successfully process a source data set, all associated Alternate Indexes and Paths must also be renamed. If the rename filter criteria does not include rename criteria for all of the data sets in the Sphere, message ADR763E will be issued and the source data set will not be dumped successfully. Otherwise the dump of the data set will fail.
13.1.1 Keyword syntax

NEWNAMEUNCONDITIONAL and NEWNUNC are valid specifications where:

NEWNAMEUNCONDITIONAL(prefix)
- You can only rename the prefix of the source.

NEWNAMEUNCONDITIONAL(oldname,newname)
- oldname: Check if it matches the source data set name.
- newname: This is 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.

13.1.2 Restrictions

NEWNAMEUNCONDITIONAL will be restricted to API invocations of DFSMSdss only. Otherwise error message ADR129E will be issued.

DFSMSdss will not verify that the attributes of the source data set matches that of the new named data set. It is up to the program invoking DFSMSdss to perform this verification. This could cause a restore to fail when attempting to restore to an existing data set when the attributes of the data set you are trying to restore to do not match.

NEWNAMEUNCONDITIONAL specifies that a source data set should be given a new name during dump processing, whether or not the data set exists with the new name. If the old name filter matches and there is an error in the new name filter, the data set is not dumped and existing message ADR468E will be issued.

The keywords of the NEWNAMEUNCONDITIONAL parameter are:

- **prefix**: Specifies the prefix used to replace the first-level qualifier of the data set name. It is optional but, if specified, it 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 source 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.

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 source data set INCLUDE keyword for the DUMP command, and their rules match.

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.
13.2 DEFRAG performance enhancement

The DEFRAG function consolidates the free space on a volume to prevent out-of-space conditions on new allocations. DEFRAG accomplishes this by relocating data set extents on a DASD volume to reduce or eliminate free space fragmentation, and prints a report about free space and other volume statistics.

The new version of DEFRAG has some internal enhancements and keyword changes and is the default.

As volumes have increased in size, the time that it takes to process them with DEFRAG has also increased. The larger volume size inherently requires more time to process the increased number and size of data sets. This increased processing time affects most customers because they have limited time in which to run maintenance tasks on their systems.

CONSOLIDATE, being a function that is extracted from DEFRAG, shares the same performance problems. The changes introduced in this support will begin to address this issue, as discussed next.

13.3 CONSOLIDATE command

A new command, CONSOLIDATE, is being created beginning in this release that will perform both extent consolidation and reduction, and allow the support to specify which data sets are to be processed. This command will try to reduce the number of extents of a data set as much as possible even when the entire data set cannot be reduced to one extent.

See DFSMSdss Storage Administration Reference, SC26-7402 for command syntax.

The CONSOLIDATE function consolidates data set extents on a volume to improve performance when reading data sets. CONSOLIDATE accomplishes this by combining contiguous extents of a data set into one extent and by relocating multiple non-contiguous data set extents into contiguous space and prints a report of volume statistics.

CONSOLIDATE performs both data set extent consolidation and extent reduction for data sets that occupy multiple extents on a single volume. It allows you to specify which data sets are to be processed:

- It attempts to relocate multiple non-contiguous data set extents into contiguous space.
- It attempts to reduce the number of extents of a data set as much as possible even when the entire data set cannot be reduced to one extent.

It supports the same type of data set filtering as COPY, DUMP, and RESTORE functions.
13.3.1 RACF protection

The new STGADMIN.ADR.STGADMIN(CONSOLIDATE RACF FACILITY class profile restricts the CONSOLIDATE command.

The RACF profile allows you to perform a CONSOLIDATE operation without having READ access to the data sets that are moved. See DFSMSdss Storage Administration Reference SC26-7402 for command syntax.

Performance improvements will be made in the functions listed here by replacing the method of maintaining the locations of data set extents and free space with a more efficient method.

- **MMOVPC(n,p)**
  
  Where:
  
  - n: The percentage of tracks on the volume that DFSMSdss is to try to assemble as free tracks in a contiguous area.
  - p: The number of passes DFSMSdss is to make in attempting to assemble the tracks.
  
  The MMOVPC parameters stop the DEFRAG run when n% contiguous tracks on the volume are assembled as free. If n% contiguous tracks already exist as free tracks, the DEFRAG function tries to further reduce the fragmentation of the volume but no more than n% tracks are relocated. If more than n% tracks must be relocated, no DEFRAG is performed.

  The MMOVPC keyword is recommended instead of MAXMOVE when running DEFRAG on an EAV. The MMOVPC will apply separately to the track-managed space and the cylinder-managed space.

  MAXMOVE and MMOVPC are mutually exclusive.

- **MAXTIME(nummins)**
  
  Where:
  
  - nummins: A decimal number (0-9999) that specifies the maximum number of minutes the DEFRAG function will run. A value of 0 is ignored.
  
  MAXTIME specifies the maximum number of minutes that the DEFRAG function should be allowed to process. This allows the user to control the time that the job will run. MAXTIME will be checked after processing each data set. If the MAXTIME has passed, the DEFRAG function will end.

  We recommend using this keyword whenever customers has a limited window for this type of maintenance.

  If the CONSOLIDATE keyword is specified, MAXTIME will be ignored.

  **Note:** The elapsed time of the operation might be slightly longer than the MAXTIME specified because the MAXTIME is checked after each data set has been processed.
13.3.2 Coexistence APAR A22900

DFSMSdss full volume and tracks dumps of EAVs are not compatible with dumps of volumes that are 64 K cylinders or fewer due to changes required to format the extended-address space in the dump. Changes have been made in z/OS V1R10 DFSMSdss to identify dumps of EAVs. Because many customers run more than one level of z/OS in their environments or at times must provide data in the form of DFSMSdss dumps to out-centers that are not at the same z/OS level. DFSMSdss provides limited restore capability on supported lower levels of z/OS for data dumped from an EAV on z/OS V1R10 or higher levels.

Versions of DFSMSdss prior to V1R10 will not perform a full volume restore of a full volume dump from an EAV. In addition, tracks restore where track 0 is included will fail and tracks restore, not including track 0, will restore only the track-managed space from an EAV. Logical data set restore and physical data set restore will be changed to convert F8/F9 DSCBs to F1 DSCBs. When a data set is restored that had an F8/F9 pair when it was dumped, if attributes are being lost due to the inability to restore the F9, a new message, ADR556W, will be issued. If no attribute values exist, no message will be issued.
Virtual Concurrent Copy in DFSMS V1.10

With DFSMS V1.10, DFSMSdss and DFSMShsm have been enhanced to support Virtual Concurrent Copy (VCC) on DS8000® using FlashCopy® V2. VCC is equivalent to SnapShot used on the RVA Ramac devices. Like SnapShot, VCC also flashes data to an intermediate location before continuing the backup.

A similar feature, Concurrent Copy, uses a cache sidefile as the intermediate location. Cache capacity is becoming a constraint in some installations, and larger volumes—for instance, an EAV—might cause an even bigger constraint.

In this chapter we cover VCC and the implementation of VCC, also at the same time drawing comparisons to the SnapShot and the Concurrent Copy implementation.

In an environment where Extended Address Volumes are present, the SDM VCC enhancement will also allow the use of VSAM linear data sets (LDS) for Working Space Data Sets (WSDSs), so that the WSDSs can be allocated in the cylinder-managed space on an EAV.
14.1 Virtual Concurrent Copy overview

In order to understand Virtual Concurrent Copy (VCC), it is important to first understand Concurrent Copy. Concurrent Copy is a point-in-time copy method that uses a combination of microcode and software to preserve the original data as of the point-in-time when the operation starts, while allowing the data to be updated by the application. SDM provides low level support for performing the hardware functions of protecting the original data and reading tracks corresponding to that point-in-time. DFSMSdss provides the user interface, and is responsible for writing the data to its destination.

Concurrent Copy uses control unit cache, which often is a limited resource. The new VCC support using FlashCopy will alleviate the problem, because updates are preserved using FlashCopy and are therefore, not stored in the cache sidefile in the control unit but in WSDSs.

Prior to this release, Virtual Concurrent Copy used SnapShot to provide a Concurrent Copy-like, or Concurrent Copy-compatible function, when the source device supports SnapShot (RVA only), but did not support Concurrent Copy.

During VCC, data is flashed from the source location to an intermediate location, and the data is gradually copied to the target location using normal I/O methods. The operation is logically complete, after the source data is flashed to the intermediate location and physically complete after the data is moved to the target media. The use of SnapShot for VCC is automatic, when the Working Space Data Sets (WSDSs) have been defined on the system and the request is done through the Concurrent Copy parameter in DSS.

With this VCC enhancement, DFSMSdss and SDM will support VCC using FlashCopy V2, similar to the way SnapShot is used (as described in the previous paragraph). No physical background copy (NOCOPY) mode will be the normal FlashCopy mode in the VCC scenario.

In the Extended Address Volume (EAV) environment, the SDM VCC enhancement will allow the use of VSAM linear data sets (LDS) for WSDSs, so that the WSDSs can be allocated in the Cylinder-Managed space on an EAV. This VCC enhancement will update the existing optional CONCURRENT keyword for the COPY and DUMP commands. New sub-keyword options allow the user to designate the type of Concurrent Copy that DFSMSdss should use and whether DFSMSdss should retry with different method.

The new option of having a separate shared PARMLIB for ANTMAIN is also covered in this section.

14.2 Setting up the Virtual Concurrent Copy environment

There are several steps you must take to implement Virtual Concurrent Copy on a DFSMS V1.10 system.

14.2.1 ANTMAIN PARMLIB support

As mentioned new support is being added to allow use of PARMLIB to control ANTMAIN. ANTMAIN control parameters - which apply to all systems active using a given SYS1.PARMLIB concatenation- can be specified in a new member ANTMIN00.
ANTMAIN control parameters that apply to a specific system name can be specified in a component-specific PARMLIB data set named hlq.ANTMAIN.PARMLIB, under a member name equal to the system name. An example is SYS1.ANTMAIN.PARMLIB(SYSTEM1). In this way, systems can share a common SYS1.PARMLIB concatenation and still have different ANTMAIN configurations. For each parameter, the value found in hlq.ANTMAIN.PARMLIB(sysname) takes precedence, followed by the value in the ANTMINxx member of the SYS1.PARMLIB concatenation, followed by the default value, if not specified.

### 14.2.2 XRC PARMLIB support

The following parameters are supported:

- **STARTUP HLQ**: Specifies the HLQ to be used for data sets associated with Concurrent Copy and VCC.
- **FLAG**: Syntax is the same as the XRC flag parameter, but the valid list of flags is different.
- **PATCH**: Syntax is the same as the XRC patch, but the valid list of things to patch is different.

The SETPARMLIB MODIFY command causes the ANTMAIN address space to refresh its settings from SYS1.PARMLIB(ANTMINxx). The SYS1.PARMLIB concatenation is searched for member ANTMINxx, and its contents are applied. Message IEE252I reflects the actual location that ANTMINxx was found.

Figure 14-1 is an example of a ANTMINxx PARMLIB member.

```
/* ANTMIN00 SAMPLE */
/* Startup section */
/* change the HLQ from SYS1 to IBMUSER */
/* Allow up to 500 messages about WSDS per event */
/* don't allow WSDS to be on PPRC primary volumes */
STARTUP -
  HLQ(IBMUSER) -
  WSDMSGMAX(500) -
  ALLOWPPRCONWSDS(NO)
```

Figure 14-1  ANTMINxx PARMLIB member sample

The result of the command `F ANTMAIN,SETPARMLIB` is shown in Figure 14-2.

```
ANTI1020I PARAMETERS FOUND IN MEMBER ANTMIN00 OF DATA SET 'SYS1.PARMLIB' HAVE BEEN APPLIED
```

Figure 14-2  F ANTMAIN,SETPARMLIB command results

### 14.2.3 Working space data sets

Before you can use the FlashCopy function for Virtual Concurrent Copy, you must ensure that working space is available by allocating Working Space Data sets (WSDSs) on one or more volumes in the same Data Set FlashCopy V2 enabled, storage subsystem as the source data sets.
System Data Mover (SDM) uses the working space data sets as the intermediate location for Virtual Concurrent Copy. The naming convention for using these working space data sets is hlq.ANTMAIN.FCWKnnnn. Variable hlq is the high level qualifier that you specify in the SDM PARMLIB member and nnnn is a four-digit decimal number in the value range 0000–9999.

If you use both VSAM and physical sequential data sets, you must specify a unique nnnn component of the name across both kinds of data sets. The working space data sets must be cataloged. SDM performs a catalog search for usable working space data sets that match the naming convention.

You also must allocate data sets as single-volume, non-indexed VSAM data sets such as LDS and ESDS, or non-extended-format sequential data sets. You can use extended-format non-indexed VSAM data sets. The data sets can be SMS-managed or non-SMS-managed. If you want to allocate secondary space, you must extend the data set by filling it with data, before you start the DFSMSdss processing.

You can reallocate or extend a working space data set only, when SDM does not have the data set enqueued. SDM uses the new reallocated or extended data set on subsequent runs of the FlashCopy operation.

An example of defining a linear VSAM data set for WSDS is shown in Figure 14-3.

```
DEFINE CLUSTER -
(NAME(SYS1.ANTMAIN.FCWK0001) -
LINEAR -
REUSE -
SHAREOPTIONS(3 3) -
NOERASE -
CYL(4147 0) -
VOLUMES(VVVVVV)) -
DATA(NAME(SYS1.ANTMAIN.FCWK0001.DATA) -
)
```

Figure 14-3 VSAM linear data set allocation for WSDS

Allocating the WSDS as a VSAM data set, enables you to allocate the WSDS in the EAS on an EAV.

Figure 14-4 shows an example of allocation of a WSDS as a sequential data set.

```
//*
//ALLOC EXEC PGM=IEFBR14
//DD01 DD DSN=SYS1.ANTMAIN.FCWK0001 ),
// DISP=(NEW,CATLG,DELETE),
// UNIT=3390,
// VOL=SER=vvvvvv,
// SPACE=(CYL,3339),
// DCB=(DSORG=PS,RECFM=U,LRECL=0,BLKSIZE=27998)
```

Figure 14-4 JCL to allocate WSDS
The LRECL and block size can be any valid combination. The VSAM Control Interval (CI) size can be any value. SDM uses the tracks within the data set as the target of FlashCopy operations, and you should not try to access them using normal data access methods.

SDM holds an enqueue for the data set, when the FlashCopy operation uses working space data set and releases the enqueue after FlashCopy operation finishes using the data set.

You have to consider the number of WSDSs as the accumulated size of these must reflect the need for concurrently active backup jobs using VCC. Do a generic search using ISPF 3.4 or a LISTCAT on all the data sets being processed, adding up the total number of cylinders. The total capacity of the WSDSs allocated has to be bigger (or equal to) the total amount of space needed for the source data sets being backed up concurrently.

You can also discover how many WSDSs are needed by running with CC(CACHEREQ), collecting SMF 42 subtype 4, and sum up the total tracks processed field for all jobs running at the same time.

You can add more WSDSs after ANTMAIN completes the initialization process. SDM uses these data sets the first time it encounters an out-of-working-space condition. When this condition occurs during a FlashCopy operation, SDM refreshes the list of working space data sets by performing a catalog search for data set names that match the naming convention.

You can refresh the list of WSDS also by using the command:

```
F ANTMAIN, REFRESHWS data set name disposition
```

The `data set name and disposition` is optional. If you do not specify those, the output is sent to the console. Remember to preallocate the data set before issuing the command to use the attributes FB/LRECL 80 with appropriate blocksize.

### 14.2.4 WSDS considerations

To ensure that unauthorized users cannot access sensitive data, you should use RACF, or an equivalent security product, to protect the WSDSs.

You must allocate data sets on a volume in each storage subsystem, that you are using for Virtual Concurrent Copy, otherwise they will not be recognized.

If you define more than one device type on the storage subsystem, you must allocate a Working Space Data Set on each device type that contains a data set, which you intend to process using SnapShot or FlashCopy. You must allocate at least one Working Space Data Set, if a system or device type for Concurrent Copy operation runs simultaneously from more than one system and accesses data on the same storage subsystem.

For example, you must allocate three Working Space Data Sets to process data on a DASD subsystem from three z/OS systems, on devices of each device type containing data processed with Concurrent Copy.

The total size of all Working Space Data Sets that you allocate on each storage subsystem should be equal to or exceed the largest total amount of data to be processed in a single DFSMSdss COPY or DUMP operation on that storage subsystem.

If there is insufficient space, the Concurrent Copy initialization for one or more data sets in the job fails.
14.2.5 Running VCC in production

This VCC enhancement will update the existing optional CONCURRENT keyword for the DFSMSdss COPY and DUMP commands. New sub-keyword options allow the user to designate the type of Concurrent Copy that DFSMSdss should use and whether DFSMSdss should retry with different methods.

Currently, Space Efficient FlashCopy is supported for volume level only. In the future, when track level Space Efficient FlashCopy is supported and the user has allocated WSDSs on space efficient volumes, SDM will use Space Efficient FlashCopy for VCC.

The existing optional CONCURRENT keyword on the COPY and DUMP commands has been updated with the sub-keyword options that are described in Figure 14-6.

![Figure 14-5 CONCURRENT sub-keywords in support of VCC](image)

**ANYREQ or REQUIRED** Specifies that the data is to be processed with Virtual Concurrent Copy if possible. If Virtual Concurrent Copy could not be used or has failed, the data is to be processed with cache-based Concurrent Copy if possible. If both types of Concurrent Copy were not possible or have failed, the data will not be processed.

**VIRTUALREQ** Specifies that data is to be processed with Virtual Concurrent Copy if it is possible. Otherwise, the data is not processed. Cache-based Concurrent Copy and standard I/O are not attempted.

**CACHEREQ** Specifies that data is to be processed with cache-based Concurrent Copy if it is possible. Otherwise, the data is not processed. Virtual Concurrent Copy and standard I/O are not attempted.
**ANYPREF or PREFERRED** Specifies that the data is to be processed with Virtual Concurrent Copy if possible. If Virtual Concurrent Copy could not be used or has failed, the data is to be processed with cache-based Concurrent Copy if possible. If both types of Concurrent Copy were not possible or have failed, the data is processed as if CONCURRENT were not specified. ANYPREF is the default if CONCURRENT is specified without a sub-keyword option.

**VIRTUALPREF** Specifies that data is to be processed with Virtual Concurrent Copy if it is possible. Otherwise, the data is processed using standard I/O as if CONCURRENT were not specified. Cache-based Concurrent Copy is not attempted.

**CACHEPREF** Specifies that data is to be processed with cache-based Concurrent Copy if it is possible. Otherwise, the data is processed using standard I/O as if CONCURRENT were not specified. Virtual Concurrent Copy is not attempted.

**NONE or STANDARD** Specifies that data is to be processed with standard I/O as if the CONCURRENT keyword has not been specified. This option is the equivalent of SMS management class definition of the backup copy technique, CONCURRENT(STANDARD).

### 14.2.6 DSS data movement method preference

In general, DFSMSdss copy operations attempt to use data movement methods in the following order (when the device supports the data movement method):

1. Fast replication (for example, FlashCopy, SnapShot) - unless FASTREPLICATION(NONE) is specified.
2. Virtual Concurrent Copy - unless CONCURRENT(CACHEREQ | CACHEPREF | NONE | STANDARD) is specified, or CC is not specified.
3. Cache-based Concurrent Copy - unless CONCURRENT(VIRTUALREQ | VIRTUALPREF | NONE | STANDARD) is specified, or CC is not specified.
4. Standard I/O - unless FR(REQ) or CONCURRENT(ANYREQ | VIRTUALREQ | CACHEREQ) is specified.

**Note:** CC and VCC are affected by the RACF FACILITY CLASS profiles associated with the CONCURRENT keyword. The FACILITY class profile names are: STGADMIN.ADR.COPY.CNCURRNT and STGADMIN.ADR.DUMP.CNCURRNT
Figure 14-6 shows an example of a DSS dump job running a VCC session (VIRTUALREQUIRED keyword).

```
/*
//BACKUP01 EXEC PGM=ADROSSU ,PARM='TYPRUN=NORUN'
//BACKUP1 DD DSN=data set name ,
//      DISP=(,CATLG),
//      SPACE=(CYL,(150,50),RLSE)
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *
     DUMP DATASET(INCLUDE(                     -
     ** -
     )))  -
     OUTDD(BACKUP1) -
     ALLD(*) ALLEXC -
     LOGINDYNA((VYVVV1,3390), -
               (VYVVV2,3390), -
               (VYVVV3,3390), -
               (VYVVV4,3390), -
                )  -
     COMPRESS -
     OPT(4) -
     TOL(ENQF) CC(VIRTUALREQUIRED)
*/
```

Figure 14-6  DSS dump job running VCC

### 14.2.7 DFSMSHsm backup processing

During backup processing, DFSMSHsm will accept the new VCC management class BACKUP COPY TECHNIQUE keywords and pass the corresponding keyword to DFSMSdss. Likewise, during aggregate backup processing, the new ABACKUP COPY TECHNIQUE keywords will be recognized and passed to DFSMSdss.

These new keywords will be honored for backup processing of the DFSMSHsm control data sets when they are specified in the management class for the control data sets.

Refer to *DFSMShsm Storage Administration Guide*, SC35-0421 for more detailed DFSMSHsm information about using VCC. The updated sections include:

- Updated Concurrent Copy options for “Specifying Concurrent Copy for Volume Dumps”
- “Specifying Whether to Use Concurrent Copy”
- “Using Concurrent Copy for Data Set Backup”
- “Data Set Backup With Concurrent Copy”
- “Authorizing Concurrent Copy with RACF”
- “Concurrent Copy Management Class Support”
- “Authorizing The Use of Concurrent Copy for Data Set Backup Commands”
14.2.8 Displaying WSDS usage

Using the LISTWS command during the Virtual Concurrent Copy session will display the usage of the WSDSs. Issue the command:

```
F ANTMAIN,LISTWS ALL
```

The output from this command is displayed in Figure 14-7.

```
ANTM6020I LISTWS ALL OUTPUT - 001 530
ANTM6022I WSDS PREFIX SYS1.ANTMAIN
ANTM6023I NAME      STATUS  JOBS  RC    DS CYL    FREE CYL  FLAGS
ANTM6024I ----------------------------------------------------------
-          ANTM6025I FCWK0001  INUSE      1     0      3338         0  E0008020
ANTM6025I FCWK0002  INUSE      1     0      3338         0  E0008020
ANTM6025I FCWK0003  INUSE      1     0      3338         0  E0008020
ANTM6025I FCWK0004  INUSE      1     0       967         0  E0008020
ANTM6025I FCWK0005  FREE       0     0      4147      4147  00008020
```

Figure 14-7  LISTWS command output

As you see from the LISTWS display, four WSDSs are totally used during this active Concurrent Copy session. One WSDS is not in use. This might be due to the fact that it is not needed, but the reason might also be, that this WSDS is not allocated in the same storage control unit as the data sets currently being backed up - and consequently cannot be used.

If your job starts displaying ADR735W and ADR888I messages with the diagnostic information: 00001800-00000000, this might be the reason. The solution is then to allocate more WSDSs within the storage control unit, you are backing up.

For more information about Virtual Concurrent Copy and Working Space Data Sets, refer to the manual z/OS DFSMS Advanced Copy Services, SC35-0428.
In z/OS V1.10, you can define multiple, Secondary Lock Structures for VSAM RLS workloads, to reduce locking constraints. A new SMS Storage Class attribute called the lock set lets you specify a DFSMS lock structure to be used for VSAM records. An installation can define up to 256 lock sets per sysplex.

In this chapter we provide you with the information needed to implement and manage multiple Secondary Lock Structures. We discuss the following topics:

- VSAM RLS secondary lock structure overview
- Preparing for VSAM RLS Secondary Lock Structures
- Definition of RLS Secondary Lock Structures
- Migrating to multiple lock structures
- Operating secondary lock structures
- Fallback from using Multiple Lock Structures
15.1 VSAM RLS secondary lock structure overview

VSAM record-level sharing (VSAM RLS) is an extension to VSAM that provides direct record-level sharing of VSAM data sets from multiple address spaces across multiple systems. VSAM RLS uses the z/OS Coupling Facility for cross-system locking, local buffer invalidation, and cross-system data caching. Primary users of VSAM RLS include high-volume applications that access VSAM data sets, such as CICS® applications.

In previous releases of z/OS VSAM RLS applications used a single CF lock structure, the IGWLOCK00 lock table, for all record level sharing—regardless of the number of systems that might need this function. This limitation affected system and application availability, and inhibited the isolation of workloads. With only one lock structure to use for a sysplex, an installation’s test workloads could degrade the performance of its production workloads. And, if an application obtained a large number of record locks between commit operations, other applications had to wait until the application freed the locks.

In z/OS V1.10, you can define multiple, Secondary Lock Structures for VSAM RLS workloads, to reduce locking constraints. A new SMS Storage Class attribute called the lock set lets you specify a DFSMS lock structure to be used for VSAM records. An installation can define up to 256 lock sets per sysplex.

When an application opens a VSAM data set, RLS processing determines which lock structure to use by checking the Storage Class defined for the data set. If the Storage Class specifies a Secondary Lock Structure, RLS processing uses the Secondary Lock Structure for serializing access to records in the data set. Otherwise, RLS processing uses IGWLOCK00 for all record locking (as done in previous releases).

Other types of locks continue to use IGWLOCK00, which is still required for the initialization of the SMSVSAM address space. The usual tasks related to the definition, administration, and use of the primary lock structure, IGWLOCK00, do not change with this enhancement. The following table lists the types of tasks and associated procedures that you must complete to fully use this enhancement.

15.1.1 Basic record level sharing setup

This section describes how to set up the resources that you need for using VSAM record-level sharing (RLS) and DFSMS Transactional VSAM (DFSMStvs) basically to set the scene before going into more detail about the new Multiple Lock Structure feature in DFSMS V1.10.

To be able to support VSAM RLS processing on a VSAM data set has to be SMS managed, so if that is not currently the case, you have to convert the data set to be SMS managed. Also, sharing of data sets has to happen within a Parallel Sysplex®.

Change the attributes for a candidate data set to specify whether the data set is to be recoverable or nonrecoverable. Specify LOG(NONE) if the data set is non-recoverable; specify LOG(UNDO) or LOG(ALL) if the data set is recoverable. Finally, the data set also needs to have assigned a Storage Class and a Data Class enabling RLS support as described in more detail later in this chapter.

VSAM RLS exploits data sharing technology in the Coupling Facility (CF), including a CF based lock manager and a CF cache manager. RLS is a mode of access to VSAM data sets interpreted at open time. The open can happen by using the JCL parameter RLS or by specifying MACRF=RLS in the ACB.
Access Method Services are not using RLS, when performing IDCAMS EXPORT, IMPORT, PRINT or REPRO command.

RLS access is supported on KSDS, ESDS, RRDS, and VRRDS data sets and for alternate indexes.

15.1.2 SMSVSAM address space

The VSAM RLS functions are provided by the SMSVSAM server, residing in a system address space. Define and activate at least two sharing control data sets (SHCDS), and one spare SHCDS to be used by SMSVSAM for recovery purposes. The sharing control data set (SHCDS) is designed to contain the information required for DFSMS to continue processing with a minimum of unavailable data and no corruption of data when failures occur. The SHCDS data sets can be either SMS or non-SMS managed.

15.1.3 CF structures

You define CF cache and lock structures to MVS, using the CF resource manager (CFRM) policy. You have to specify names, sizes, and locations of the structures in the Coupling Facility resource management (CFRM) policy, using the IXCMIAPU data utility.

You can activate the new CFRM policy, using the SETXCF START,POLICY command.

For details on defining lock structures to the CFRM, see z/OS MVS Programming: Sysplex Services Guide, SA22-7617 and z/OS MVS Programming: Sysplex Services Reference, SA22-7618.

Storage Class definition

CF cache structures have to be defined in the SMS base configuration. The association of your CF cache structure to the Storage Class is enabled through the ISPF Cache Update Function in the Control Data Set menu. You define a cache set name and relate a cache structure to this name. Your RLS Storage Class has to point to this cache set name.

Caching

VSAM RLS support allows multiple levels of CF caching for DFSMS cache structures. The value in the DFSMS Data Class RLS CF Cache Value determines the level. The default level is ALL indicating that RLS caches both the data and index part of the VSAM data set in the Coupling Facility. If you specify NONE, only the index part is cached. Specifying UPDATESONLY will only allow updates during write to be cached in the CF.

To be able to run greater than 4K caching the IGDSMSxx PARMLIB parameter RLS_MAXCFFEATURELEVEL has to have a value of A in all systems in the sysplex (the default is Z). When DFSMS cache structures connect to the system, VSAM RLS issues an IGW500I message to indicate that greater than 4K caching is active. The cache structures connect to the system through the first instance of a data set opened on each system.

To determine the code level on each system in the sysplex and whether the RLS CF Cache Value keyword is honored, use the commands:

- D SMS, SMSVSAM
- D SMS, CFCACHE
**SMSVSAM communication**

SMSVSAM communicates with the requesting address spaces and contains data buffers with the transactions. In some cases with a high volume of transactions, storage constraint means performance slowdown. VSAM RLS offers a possibility to move a 64 bit addressable virtual storage. To achieve this assign a Data Class with the option RLS Above the 2-GB Bar (YES), on your RLS data class. IBM recommends using this option for applications with a high rate of transactions.

The necessary PARMLIB initialization definition RLSINIT(YES) in the IGDSMSxx PARMLIB member is essential for activating RLS. This parameter decides whether SMSVSAM initializes or not at IPL time. The activation of RLS support can happen through IPL, after the RLSINIT has been defined in IGDSMSxx or dynamically by the activate commands:

- `/V SMS,SMSVSAM,ACTIVE`
- `/SETSMS RLSINIT(YES)`

### 15.2 Preparing for VSAM RLS Secondary Lock Structures

In z/OS V1.10, you can define multiple, Secondary Lock Structures for VSAM RLS workloads, to reduce locking constraints.

A new SMS Storage Class attribute called the lock set lets you specify a DFSMS lock structure to be used for VSAM records. An installation can define up to 256 lock sets per sysplex. When an application opens a VSAM data set, RLS processing determines which lock structure to use by checking the Storage Class defined for the data set. If the Storage Class specifies a Secondary Lock Structure (a Lock Set Name), RLS processing uses the Secondary Lock Structure for serializing access to records in the data set. Otherwise, RLS processing uses IGWLOCK00 for all record locking (as done in previous releases).

A Secondary Lock Structure connection persists beyond data set closure. Secondary Lock Structures are disconnected, when the SMSVSAM address space is terminated. After the address space is terminated, you can delete the lock structure itself using the command:

`VARY SMS,SMSVSAM FORCEDELETESTRUCTURE(lock structurename)`

Secondary Lock Structures are intended for record locks only. Other types of internal locks used for controlling the shared protocol continue to use IGWLOCK00, which is still required for the initialization of the SMSVSAM address space. The usual tasks related to the definition, administration, and use of the primary lock structure, IGWLOCK00, do not change with this enhancement.

### 15.3 Definition of RLS Secondary Lock Structures

RLS Secondary Lock Structures are defined by using the new parameter **Lock Update** function in the SMS Base configuration. Using the **Lock Set** attribute, you can define a new Coupling Facility lock structure and associate it with a specific SMS Storage Class. DFSMS allows up to 256 Lock Sets and lock structures to be defined. Each Lock Set can contain a single lock structure name. However the maximum number of Lock structures that can be connected is between 10 and 14. Specify the number in the MAXCAD value of IEASYSxx.

When an application opens a VSAM data set, RLS determines which lock structure to use by checking the Storage Class defined for the data set. If the Storage Class specifies a Secondary Lock Structure, RLS processing uses the Secondary Lock Structure for serializing access to records in the data set. Otherwise RLS uses IGWLOCK00 for all record locking.
15.4 Migrating to multiple lock structures

When you migrate to DFSMS V1.10 VSAM RLS processing will by default continue using IGWLOCK00. If you plan to enable Secondary Lock Structures in a sysplex containing both z/OS V1.10 and pre-V1.10 systems, note that lower releases of z/OS cannot share a VSAM data set with systems that are using Secondary Lock Structures. If an application on a lower-level system is the first to open a VSAM data set, the z/OS V1.10 system reverts to using IGWLOCK00 for all record locks, regardless of whether a lock set is defined for the Storage Class.

PTFs on lower releases have to be installed to prevent the opening of a data set, if it is already open using a Secondary Lock Structure on a z/OS V1.10 system. If a lower level system tries to open an RLS enabled data set, already opened by a DFMS V1.10 system, the open will fail with the message:

IEC161I 215-799 - Open failed for the data set dsname. ccc = 799: Open failed for the data set dsname. The file was already open on a system that supports secondary lock structures.

**Note:** If a data set was previously assigned to a lock structure and there are related locks outstanding, then that lock structure will be selected even if the Storage Class lock set has been changed to another lock structure.

The decision to migrate to a Multiple Lock Structure based environment might be based upon a need to separate workloads from interfering with each other (for example, test and production workloads or batch and online applications). Migrating to using Multiple Lock Structures from a non-RLS environment, basically you first have to define the environment by going through the actions mentioned previously. For more details, refer to *DFSMS Using Data Sets*, SC26-7410.

15.4.1 Enabling multiple lock structure support

After defining the base RLS environment, you have some additional tasks that must be done to enable the Multiple Lock Structure support:

- Decide your Lock Set names and the corresponding CF Log Structure names and define these in the Lock Update option in the ISMF Control Data Set menu.
  - To relate the structure name to your RLS Storage Class define the lock set name on the appropriate field in the Storage Class.
  - If no lock set names are specified, RLS processing uses lock structure IGWLOCK00 for all record locking, as in previous releases. Validate and activate the SMS configuration, which causes the new definitions in the Source Control Data Set (SCDS) to take effect.
  - Define the new name, size and location of the new lock structures in the Coupling Facility resource management (CFRM) policy,
  - To activate the policy, issue the following command:

    `SETXCF START,POLICY,TYPE=CFRM,POLNAME=policyname`
The new CF Lock Set Update Panel is shown Figure 15-1. The Lock Set name shown points to the Lock Set Name in the Storage Class definitions, the CF Lock Structure Name relates to the Coupling Facility structure name just defined.

<table>
<thead>
<tr>
<th>CF LOCK SET UPDATE</th>
<th>PAGE 1 OF 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command ===&gt;</td>
<td></td>
</tr>
<tr>
<td>SCDS Name : SYS1.SMS.SCDS</td>
<td></td>
</tr>
<tr>
<td>Define/Alter/Delete CF Lock Sets: ( 001 Lock Sets Currently Defined )</td>
<td></td>
</tr>
<tr>
<td>Lock Set CF Lock Structure Name</td>
<td></td>
</tr>
<tr>
<td>RLSHSM01 RLSCACHEHSM01</td>
<td></td>
</tr>
</tbody>
</table>

More CF Lock Sets to Add? . . . N (Y/N)  
Use ENTER to Perform Validation; Use UP/DOWN Command to View other Pages;  
Use HELP Command for Help; Use END Command to Save and Exit.

If the RLS enabled data sets are open for processing, close the application and reopen it to have the serialization based on the new Secondary Lock Structure.

Your environment has now taken advantage of the new Multiple Lock support.

ISMF has new fields added displaying which Lock Set and which Lock Structures are being used on a given data set name, and a display function in relation to Lock Sets in the base configuration. See the following topic for information about how to further verify your installation by command.

15.4.2 Displaying secondary lock structures

You can use the D SMS, CFLS or the D SMS,SMSVAM commands to display the status of individual VSAM RLS lock structures, or all current VSAM RLS lock structures. If you enter the D SMS,CFLS command without lock structure name or the ALL keyword, only IGWLOCK00 data is returned.
Output from the D SMS,CFLS(RLSCACHEHSM01) command is shown in Figure 15-2.

```
RESPONSE=SC70
IEE932I 14:08:57 Display SMS,CFLS(RLSCACHEHSM01)
PRIMARY STRUCTURE:RLSCACHEHSM01 VERSION:0000000000000000 SIZE:0K
RECORD TABLE ENTRIES:0 USED:0
LOCK STRUCTURE MODE: SIMPLEX  STATUS:
  System  Interval  LockRate  ContRate  FContRate  WaitQLen
SC70     1 Minute  0.0      0.000      0.000       0.00
SC70     1 Hour    0.1      0.000      0.000       0.00
SC70     8 Hour    ---------- ---------- ---------- ----------
SC70     1 Day     ----------- ----------- ----------- -----------
(04)     1 Minute  0.0      0.000      0.000       0.00
(04)     1 Hour   0.0      0.000      0.000       0.00
(04)     8 Hour   0.0      0.000      0.000       0.00
(04)     1 Day    0.0      0.000      0.000       0.00

************************** LEGEND **************************
LockRate = number of lock requests per second
CONTRATE = % of lock requests globally managed
FCONTRATE = % of lock requests falsely globally managed
WaitQLen = Average number of requests waiting for locks
```

Figure 15-2  DISPLAY SMS,CFLS output

Another useful display is the D SMS,SMSVSAM display. This command displays the status of one or more SMSVSAM address spaces and the Secondary Lock Structures, including IGWLOCK00. The D SMS,SMSVSAM command will display the status on the local system, while D SMS,SMSVSAM,ALL will display the status on all systems in the sysplex.
Finally, you can do the IDCAMS SHCDS LISTDS display in batch or via ISPF on your RLS enabled data set. The display (see Figure 15-3) will show both the cache structure and the lock structure actually used for this data set making it possible for you to verify which Lock Structure is being used:

```
SHCDS LISTDS(HSM.*)
00700002
--- LISTING FROM SHCDS --- IDCSH02
---------------------------------------------------------------
- DATA SET NAME-----HSM.BCDS
  CACHE STRUCTURE----RLS_CACHE
  LOCK STRUCTURE-----RLSCACHEHSM01
  RETAINED LOCKS-------NO  NON-RLS UPDATE PERMITTED--------NO
  LOST LOCKS----------NO  PERMIT FIRST TIME---------------NO
  LOCKS NOT BOUND------NO  FORWARD RECOVERY REQUIRED-------NO
IDCAMS SYSTEM SERVICES  TIME:
11:10:12
  RECOVERABLE----------NO

  SHARING SUBSYSTEM STATUS
  SUBSYSTEM  SUBSYSTEM  RETAINED  LOST  NON-RLS UPDATE
  NAME      STATUS     LOCKS    LOCKS  PERMITTED
  ----------  ----------  ------    -----  --------------
  DFHSM70    BATCH---ACTIVE  NO      NO     NO

Figure 15-3  IDCAMS SHCDS LISTDS display
```

Look into the SHCDS LISTSUBSYS and the SHCDS LISTSUBSYSDS functions for additional information.

### 15.4.3 Monitoring multiple lock structures using SMF

Displaying the status of the Multiple Lock Structures by command has already been covered.

SMF logging has been extended to extract information about Secondary Lock Structures. The SMF monitoring can be enabled with the command:

```
V SMS,MONDS(data set name)
```

Display the monitoring status with this command:

```
D SMS,MONDS(*)
```

### 15.4.4 SMF Type 42 record

For statistics on Secondary Lock Structures as well as IGWLOCK00, check subtypes of SMF record Type 42, as follows:

- Subtype 15 (DFSMS Storage Class summary) contains new statistical sections for the Secondary Lock Structure connected for the Storage Class.
- Subtype 16 (Data Set Summary) contains the Secondary Lock Structure name and statistical information for that Secondary Lock Structure.
  - Subtype 17 (DFSMS lock table summary) contains new statistical sections for each Secondary Lock Structure known to DFSMS.
15.5 Operating secondary lock structures

If you for some reason want to quiesce access to RLS enabled data sets, use the quiesce / enable individual secondary structure names. This can be done by issuing the command V SMS,CFLS(lock structure name),ENABLE!QUIESCE.

Note that this command has no effect on IGWLOCK00. This is consistent with the VARY SMS command used on the DFSMS cache structures:
V SMS,CFCACHE(cache name),ENABLE!QUIESCE.

Enter the VARY SMS command with the ENABLE keyword. The ENABLE keyword allows new access to the VSAM RLS Secondary Lock Structure after having been QUIESCED.

Sample V SMS,CFLS commands are shown in Figure 15-4:

```
V SMS,CFLS(lockstructurename),ENABLE
RESPONSE=SC70
IGW477I DFSMS REQUEST TO ENABLE LOCK STRUCTURE RLSCACHEHSM01 IS COMPLETED.
DFSMS LOCK STRUCTURE STATUS = "ENABLED"
V SMS,CFLS(lockstructurename),QUIESCE
RESPONSE=SC70
IGW476I DFSMS REQUEST TO QUIESCE LOCK STRUCTURE RLSCACHEHSM01 IS COMPLETED.
LOCK STRUCTURE STATUS IS "QUIESCING"
```

Figure 15-4 Sample V SMS, CFLS output

Using the VARY SMS command with the Quiesce keyword prevents new access to the VSAM RLS Secondary Lock Structure. Any spheres which open for VSAM RLS access are not allowed to select the specified Lock Structure Name. Existing usage of this Secondary Lock Structure is not affected. The specified Lock Structure transitions to a fully quiesced state after all of the existing data sets using the Lock Structure have closed.

When the Secondary Lock Structure transitions to Quiesced state, SMSVSAM will not issue the MVS command to have the Lock Structure unallocated in the Coupling Facility. The Secondary Lock Structure can be unallocated only with the operator command:

```
V SMS,SMSVSAM,FORCEDELETELOCKSTRUCTURE(lock structure name)
```

If you do not specify a lock structure name at the end of the command, all Secondary Lock Structures will be deleted.

SMSVSAM can be stopped and started by the V SMS,SMSVSAM,TERMINATESERVER and V SMS,SMSVSAM,ACTIVE commands if needed.

Note: Evaluate the impact on your applications before closing down SMSVSAM. You might have to find an appropriate time for doing this.
15.6 Fallback from using Multiple Lock Structures

This description illustrates a fallback from usage of Multiple Lock Structures and not a total back out from VSAM RLS. See z/OS DFSMS Storage Administration Reference (for DFSMS/shsm, DFSMS/dss, DFSMS/dfp), SC26-7402, in the chapter titled “Falling Back from VSAM RLS Processing,” for details and considerations regarding a total fallback.

To revert to basic RLS support without use of Multiple Lock Structure support:
1. Remove the Lock Set definition on the Storage Class used for your RLS application.
2. Activate the SMS configuration
3. Restart your RLS application to open using the default lock structure IGWLOCK00.

To remove the persistent Lock Structure
1. Close down SMSVSAM on all your systems using the V SMS,SMSVSAM,TERMINATESERVER command.
2. The structure can now be deleted using the command,
   V SMS,SMSVSAM,FORCEDELETELOCKSTRUCTURE(structure name)
3. Answer YES or NO to the outstanding IGW526A message.
4. SMSVSAM can be restarted again by issuing the V SMS,SMSVSAM,ACTIVE command.
DFSMSdfp OAM V1.10 enhancements

In this chapter we provide an overview of the enhancements in DFSMS V1.10 to OAM and explain how to implement them.

We cover the following topics:

- DFSMSdfp OAM enhancement summary
- List of enhancements
- Migration considerations
- OAM enhancement implementation
- IEFSSNxx PARMLIB changes
- Changes to CBROAMxx PARMLIB
- Changes to F OAM commands
16.1 DFSMSdfp OAM enhancement summary

This section covers enhancements and migration considerations.

16.1.1 List of enhancements

In z/OS DFSMS V1.10, object access method (OAM) introduces the following enhancements:

- **Increased maximum size for objects that OAM can accept and manage:**
  The previous maximum object size of 256 MB is increased to 2000 MB (2,097,152,000 bytes) for the DASD level of the OAM hierarchy.

- **OSREQ API is extended to include new store-related functions:**
  STOREBEG, STOREPRT, and STOREEND make it easier to store objects greater than 256 MB and up to 2000 MB in size, by allowing for the object to be stored in individual parts.

- **Permanently mark volumes FULL with the F OAM,UPDATE,VOLUME command:**
  In previous releases, tape and optical volumes that were not filled to their capacity could be artificially marked as full by using the MODIFY OAM,UPDATE command to update FULL to ‘Y’ (full volumes are eligible for recycle and expiration processing, and are not considered as candidates for write requests). However, during OAM initialization, that temporary setting could be undone for tape volumes as the amount of free space is recalculated for the volumes.
  The MODIFY OAM,UPDATE,VOLUME command now lets you mark an individual tape or optical volume’s full status to permanently full by updating FULL to ‘P’. OAM initialization will not reset the full status of any volume that has been marked permanently full.

- **New FORCE keyword to stop OSMC faster:**
  The MODIFY OAM,STOP,OSMC command is updated with the FORCE keyword to provide a hard stop function, allowing the operator to stop all OSMC processing faster.

- **IEFSSNxx PARMLIB member keyword for changed OSREQ QUERY processing:**
  A new optional keyword is added to the IEFSSNxx PARMLIB member to indicate whether or not OAM is to retrieve backup volume information when processing the OSREQ QUERY command.

- **A new value LOST added, for automatic backup copy retrieval of an object:**
  The new value LOST is added to the list of valid reasons for the MODIFY OAM,START,AB and MODIFY OAM,STOP,AB operator commands. The new LOST keyword value allows automatic retrieval of the backup copy of an object when the primary copy has been marked ‘lost’ or ‘not-defined.’

- **Automatic access to backup can be configured in the CBROAMxx PARMLIB:**
  In previous releases, automatic access to backup could only be turned on and off using operator commands. These commands had to be issued every time the system was IPLed. Configuring automatic access to backup facilities in the CBROAMxx PARMLIB member alleviates the need to repeatedly enter the same operator commands and the possibility of forgetting to issue one or more of the commands.

- **A new ONLYIF keyword in the CBROAMxx PARMLIB member allows statements in the PARMLIB member to be associated to a specific system:**
  This new keyword allows an installation to share a single CBROAMxx PARMLIB member across all the systems in a sysplex to improve usability. Prior to this release,
OAMplex installations were required to have a separate CBROAMxx member for each system in the OAMplex, to accommodate system-unique statements such as OAMXCF OAMMEMBERNAME(name). Additionally, in a mixed-level system environment, some statement keywords introduced in upper level systems might be invalid (and not tolerated) on lower level systems. The ONLYIF keyword allows these system specific statements and keywords to coexist in a single CBROAMxx PARMLIB member.

16.1.2 Migration considerations

When migrating to any new release of z/OS, you must run OAM DB2 BIND jobs if you are using OAM for object support. The BIND jobs update DB2 with new OAM DB2 code.

For more information about the software requirements for DFSMS, see z/OS V1.10 Migration All Supported Migration Paths, GA22-7499.

For additional information about z/OS coexistence and release migration information, refer to z/OS Planning for Installation, GA22-7504 at:

http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/E0Z2B119

16.2 OAM enhancement implementation

The following topics lead you through the implementation process:

> “IEFSSNxx PARMLIB changes” on page 185
> “Changes to CBROAMxx PARMLIB” on page 187
> “Changes to F OAM commands” on page 188
> “Changes to OSREQ API” on page 192

16.2.1 IEFSSNxx PARMLIB changes

In this section we discuss the changes to the IEFSSNxx member of PARMLIB.

> **Specifying object sizes up to 2000 MB:**

To implement the increased OAM object size, you must take the following steps:

– In the IEFSSNxx PARMLIB member, update the MOS= keyword on the OAM1 entry to specify the maximum object size that can be stored through the OSREQ programming interface. Valid values are 50-2048. If this keyword is omitted, the maximum supported object size defaults to 50 MB.

The maximum object size is checked when objects are initially stored through the OSREQ programming interface and is not checked on subsequent retrievals, in case the keyword was omitted or its value was changed.

**Note:** If the maximum value of 2048 is specified, the actual maximum is 1 byte less, because the maximum supported object size is 2000 MB (2,097,152,000 bytes).

> **Specifying whether to retrieve backup volume information about the OSREQ QUERY command:**

– Use the new optional keyword Q B= in the IEFSSNxx PARMLIB member to indicate whether or not OAM is to retrieve backup volume information when processing the OSREQ QUERY command. This specification is at the global level and pertains to all OSREQ QUERY processing.
QB=Y indicates that OSREQ QUERY requests result in a call into the OAM address space for each backup copy. The OSREQ QUERY will return a complete backup retrieval order key for each backup copy. If a backup copy does not exist, then the OAM address space will not be called and the backup retrieval order key will contain binary zeros. This is the default.

QB=N indicates that OSREQ QUERY requests will not result in a call into the OAM address space for each backup copy. The backup retrieval order key will contain binary zeros for each backup copy regardless of whether the backup copy exists or not.

Figure 16-1 shows the output from specifying the QB=Y and QB=N keyword.

QB=Y
OSREQ QUERY TOGRP1.COLSTD PETEST00.STORE02 NODATA
OSREQ QUERY successful. Return code = 00000000, reason code = 00000000.
OSREQ QUERY response time is 8 milliseconds.
**********************************************************************
Collection name = TOGRP1.COLSTD
Object name = PETEST00.STORE02
Object size = 15000
Creation date = 2008-02-26
Creation timestamp = 14.57.44.916764
Last referenced date = 2008-03-03
Expiration date = 0001-01-01
Management class = TOGRP1D1
Storage class = OBJTAPE
Primary retrieve key = 'E3C8E2F0F0F400000004'X
Backup retrieve key = 'E3C8E2F0F1F10000015F'X
Backup2 retrieve key = '00000000000000000000'X
Estimated retrieve time = 60000

QB=N
OSREQ QUERY TOGRP1.COLSTD PETEST00.STORE02 NODATA
OSREQ QUERY successful. Return code = 00000000, reason code = 00000000.
OSREQ QUERY response time is 194 milliseconds.
**********************************************************************
Collection name = TOGRP1.COLSTD
Object name = PETEST00.STORE02
Object size = 15000
Creation date = 2008-02-26
Creation timestamp = 14.57.44.916764
Last referenced date = 2008-03-03
Expiration date = 0001-01-01
Management class = TOGRP1D1
Storage class = OBJTAPE
Primary retrieve key = 'E3C8E2F0F0F400000004'X
Backup retrieve key = '00000000000000000000'X
Backup2 retrieve key = '00000000000000000000'X
Estimated retrieve time = 60000

Figure 16-1 Queries with QB=Y and QB=N
The attributes specified will have the following effects:

- Object directory time is based on GMT.
- The messages are in mixed case English.
- OTIS will start independently from JES.
- Update ODPENDDT and ODLREFDT fields on all OSREQ retrieves.
- Allow objects with sizes up to 1025 MB.
- Retrieve the backup retrieval order keys.

Example 16-1 shows the OAM specific portion of the IEFSSNxx PARMLIB member with QB=Y specified.

```
Example 16-1  OAM part of an IEFSSN00 member

SUBSYS SUBNAME(OAM1)
INITRTN(CBRINIT)
INITPARM('TIME=GMT,MSG=EM,OTIS=N,UPD=Y,MOS=1025,LOB=P,QB=Y')
```

### 16.2.2 Changes to CBROAMxx PARMLIB

In this section we discuss the changes to the CBROAMxx member of PARMLIB.

- **Configuring automatic access to backup for OAM:**
  
  In order to configure automatic access to backup for OAM using PARMLIB, you must code new keywords in a CBROAMxx member of SYS1.PARMLIB. These keywords are added to the SETOPT statement:
  
  - ABUNREAD
  - ABOFFLINE
  - ABNOTOPER
  - ABDB2ERROR
  - ABOOK
  - ABALL

  If one of the automatic access to backup reasons is not specified with an associated SETOPT automatic access to backup keyword, then OAM defaults to not setting automatic access to backup on for that specific reason. Figure 16-2 shows an example of the SETOPT ABALL statement. In this case, if a retrieve is attempted and the optical or tape volume on which the object resides is not available, the object is retrieved from the first backup copy.

```
SETOPT ABALL(BACKUP1)
```

*Figure 16-2  SETOPT ABALL syntax includes all named reasons*

- **Limiting CBROAMxx statements to one system, or all systems in an OAMplex:**

  Use the new ONLYIF keyword to associate statements in the CBROAMxx PARMLIB member to a specific system or to all systems in an OAMplex. The scope of the ONLYIF statement is in effect until the next ONLYIF is encountered. Until now, when using an OAMPLEX, you had different CBROAMxx PARMLIB members for your OAMPLEX joining systems. The ONLYIF keyword enables you to use only one CBROAMxx PARMLIB member for all joining systems.

---

1 If SETOPT automatic access to backup keywords are added to the CBROAMxx member of PARMLIB and the system is backed down to a previous level of DFSMS. The changes to PARMLIB must also be removed or OAM address space initialization will fail when the new SETOPT keywords are encountered.
Figure 16-3 displays the use of an ONLYIF statement. Only when the SYSNAME is SC70, the OAMXCF OAMMEMBERNAME is set to OAM70 and the SETOPT ABALL(BACKUP1) is in effect.

```plaintext
Figure 16-3  ONLYIF statement
```

Figure 16-4 displays the result of the ONLYIF statement when starting OAM.

```plaintext
Figure 16-4  OAM STC output when starting OAM using ONLYIF in the
```

### 16.2.3 Changes to F OAM commands

In this section we discuss the changes to F OAM commands introduced with DFSMS V1.10.

- **Marking tape and optical volumes as permanently full:**
  
  You can use the MODIFY OAM,UPDATE,VOLUME command to mark tape and optical volumes as permanently full, by specifying a value of "P" for the FULL field of the DB2 Tape Volume Table and the DB2 Volume Table (optical).

  Specify UPDATE,VOLUME,volser, with the FULL keyword set to P as shown in Example 16-2.

**Example 16-2  Display before and after updating the volume**

```plaintext
You can use D SMS,VOL(THS008) to display the volume:
CBR12401 Object tape vol status: 530 530
VOLUME STORAGE RD WR CM IN MED FREE-SPACE % FULL LOST REQ
GROUP USE TYPE FULL FLAG CT
THS008 TOGRP1 Y Y Y N 07 478515466 0 N N 0
XCF MEMBER NAME:
CAPACITY: 488281088 UNITNAME: 3590-1
```

---

188 DFSMS V1.10 and EAV Technical Guide
Chapter 16. DFSMSdfp OAM V1.10 enhancements

ERDS PHYSICAL ID: 0012
CREATION DATE: 2008-02-22 EXPiration date: 9999-12-31
LAST MOUNTED DATE: 2008-02-26 LAST WRITTEN DATE: 2008-02-26
DATACLASS: -------- SUBLEVEL: 1
CBR11801 OAM tape volume status: 531  531
VOLUME MEDIA STORAGE LIBRARY USE W C SOFTWARE LIBRARY
TYPE GROUP NAME ATR P P ERR STAT CATEGORY
THS008 MEDIA5 SGLIB2 LIB2 P N N NOERROR PRIVATE
----------------------------------------------------------------------------------
RECORDING TECH: EFMT2 COMPACTIoN: YES
SPECIAL ATTRIBUTE: NONE ENTER/EJECT DATE: 2008-01-24
LAST MOUNTED DATE: 2008-02-26 LAST WRITTEN DATE: 2008-02-26
SHELF LOCATION:
OWNER: STC

Now set the volume permanently full:
F OAM,UPDATE,VOLUME,THS008,FULL,P
CBR1651I Tape volume record for volume THS008 updated.
This example marks the volume specified by volser as permanently full, therefore its full status is not reevaluated during subsequent OAM initializations and is retained across OAM initializations.

Display the volume again:
CBR12401 Object tape vol status: 538  538
VOLUME STORAGE RD WR CM IN MED FREE-SPACE % FULL LOST REQ
GROUP USE TYPE FULL FLAG CT
THS008 TOGRP1 Y Y Y N 07 478515466 0 P N 0
XCF MEMBER NAME:
CAPACITY: 488281088 UNITNAME: 3590-1
ERDS PHYSICAL ID: 0012
CREATION DATE: 2008-02-22 EXPiration date: 9999-12-31
LAST MOUNTED DATE: 2008-02-26 LAST WRITTEN DATE: 2008-02-26
DATACLASS: -------- SUBLEVEL: 1
CBR11801 OAM tape volume status: 539  539
VOLUME MEDIA STORAGE LIBRARY USE W C SOFTWARE LIBRARY
TYPE GROUP NAME ATR P P ERR STAT CATEGORY
THS008 MEDIA5 SGLIB2 LIB2 P N N NOERROR PRIVATE
----------------------------------------------------------------------------------
RECORDING TECH: EFMT2 COMPACTIoN: YES
SPECIAL ATTRIBUTE: NONE ENTER/EJECT DATE: 2008-01-24
LAST MOUNTED DATE: 2008-02-26 LAST WRITTEN DATE: 2008-02-26
SHELF LOCATION:
OWNER: STC

You can also use a DB2 select (for MHLRES5.TAPEVOL use your table name):
SELECT VOLSER,UNITNAME,MEDIATYP,STORGRP,TYPE,FULL,READABLE,
       WRITABLE FROM MHLRES5.TAPEVOL WHERE VOLSER = 'THS008';
----------------------------------------------------------------------------------
VOLSER UNITNAME MEDIATYP STORGRP TYPE FULL READABLE WRITABLE
----------------------------------------------------------------------------------
THS008 3590-1 07 TOGRP1 G P Y Y
DSNE610I NUMBER OF ROWS DISPLAYED IS 1

You can also use a DB2 select (for MHLRES5.TAPEVOL use your table name):
SELECT VOLSER,UNITNAME,MEDIATYP,STORGRP,TYPE,FULL,READABLE,
       WRITABLE FROM MHLRES5.TAPEVOL WHERE VOLSER = 'THS008';
----------------------------------------------------------------------------------
VOLSER UNITNAME MEDIATYP STORGRP TYPE FULL READABLE WRITABLE
----------------------------------------------------------------------------------
THS008 3590-1 07 TOGRP1 G P Y Y
DSNE610I NUMBER OF ROWS DISPLAYED IS 1

You can also use a DB2 select (for MHLRES5.TAPEVOL use your table name):
SELECT VOLSER,UNITNAME,MEDIATYP,STORGRP,TYPE,FULL,READABLE,
       WRITABLE FROM MHLRES5.TAPEVOL WHERE VOLSER = 'THS008';
----------------------------------------------------------------------------------
VOLSER UNITNAME MEDIATYP STORGRP TYPE FULL READABLE WRITABLE
----------------------------------------------------------------------------------
THS008 3590-1 07 TOGRP1 G P Y Y
DSNE610I NUMBER OF ROWS DISPLAYED IS 1
Displaying the new automatic access to backup settings:

The F OAM,DISPLAY command shows the values of the new automatic access to backup keywords in the SETOPT statement in CBROAMxx.

Specify SETOPT with a scope of ALL.

Example 16-3  F OAM,DISPLAY,SETOPT,ALL command

```
F OAM,D,SETOPT,ALL
The output from this example includes the following keywords and their specified values if any in CBROAMxx: ABUNREAD, ABOFFLINE, ABNOTOPER, ABDB2ERROR, ABLOST, and ABALL.

CBR1075I GLOBAL value for OPREINIT is GROUP
CBR1075I GLOBAL value for OPDISDLY is 0
CBR1075I GLOBAL value for MWT is 5
CBR1075I GLOBAL value for UNLOADD is 1
CBR1075I GLOBAL value for UNLOADT is 9999
CBR1075I GLOBAL value for ABUNREAD is BACKUP1
CBR1075I GLOBAL value for ABOFFLINE is BACKUP1
CBR1075I GLOBAL value for ABNOTOPER is BACKUP1
CBR1075I GLOBAL value for ABDB2ERROR is BACKUP1
CBR1075I GLOBAL value for ABLOST is BACKUP1
CBR1075I TOAMBK1 value for OPREINIT is GROUP
CBR1075I TOGRP1 value for OPREINIT is GROUP
CBR1075I TOGRP1B1 value for OPREINIT is GROUP
CBR1075I TOGRP1B2 value for OPREINIT is GROUP
```

Retrieving backup copies of objects on volumes marked lost or not defined:

The command for automatic access to backup copies is expanded to include a new reason value of LOST, for objects that are on volumes which are marked as lost or not defined. If this value is specified, when a retrieve for an object is attempted and the optical or tape volume on which the object resides is marked lost or not-defined, the backup copy of the object is retrieved (see Example 16-4).

If your SETOPT is defined with a scope of “ABALL,” no action is needed. Else specify LOST on the MODIFY OAM,START,AB command.

Example 16-4  F OAM command with keyword LOST

```
F OAM,START,AB,LOST,BACKUP1 | BACKUP2
This command specifies that backup copies objects on volumes marked lost or not defined be retrieved when retrieval is attempted.

You lose your primary volume:
CBR2003I Tape volume THS008 not found. Volume has been marked lost.
Display all lost volumes:
F OAM,DISPLAY,LOSTVOL
CBR1155I Total number of OPTICAL volumes marked lost is 0.
CBR1154I TAPE volume THS008 is a lost volume.
CBR1155I Total number of TAPE volumes marked lost is 1.
Display the lost volume THS008:
CBR1240I Object tape vol status: 802 802
VOLUME STORAGE  RD WR CM IN MED FREE-SPACE % FULL LOST REQ
GROUP USE TYPE FULL FLAG CT
```
When you retrieve an object residing on that volume, the first backup volume will be automatically be used:

IEC01A M 0B22,THS011,SL,COMP,OAM,OAM,OAM.BACKUP.DATA.TOGRP1B1

Your job then returns following information:

OSREQ RETRIEVE successful. Return code = 00000004, reason code = 04030600

The third byte of the reason code can have a value of

06 for the first backup and a value of

07 for the second backup of the volume.

If you find the lost volume, you can reset the lostflag:

F OAM,UPDATE,VOL,THS008,LOSTFLAG,OFF

Again display the volume: (only a partition of the display)

CBR1240I Object tape vol status: 837 837

<table>
<thead>
<tr>
<th>VOLUME STORAGE</th>
<th>RD WR CM IN MED FREE-SPACE % FULL LOST REQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>GROUP USE TYPE</td>
<td>FULL FLAG CT</td>
</tr>
<tr>
<td>THS008 TOGRP1</td>
<td>Y Y N 07 478515466 0 P Y 0</td>
</tr>
</tbody>
</table>

▶ Stopping all OSMC processing:

The MODIFY OAM,STOP,OSMC command is updated to allow the operator to stop all OSMC processing quickly, using the new FORCE keyword.

Previously, the STOP,OSMC only canceled all unscheduled work. Now, with the FORCE keyword, OAM cancels all scheduled and unscheduled work.

Note: Work currently in progress must complete before OSMC will terminate.
To stop all OSMC processing more quickly, specify the FORCE keyword to the MODIFY OAM,STOP,OSMC command (Example 16-5).

Example 16-5  F OAM with FORCE keyword example

First we start a MOVEVOL:

First we start a MOVEVOL:

Then we enter the F OAM,STOP,OSMC,FORCE command:

Changes to OSREQ API

In this section we discuss the changes to the OSREQ API introduced with DFSMS V1.10.

Specifying storage for large objects in parts:

The following new OSREQ API functions are used to store large objects in separate parts:

- STOREBEG – begin a store operation for an object whose total size is greater than 256 MB.
- STOREPRT – store the next sequential contiguous part of an object whose total size is greater than 256 MB.
- STOREEND – end the storage of an object whose total size is greater than 256 MB either to complete the storage of the object or to effectively cancel the storage of the object.

Using these new functions in a sequence:

a. Start with the OSREQ STOREBEG to initiate the store operation and provide much of the information that would be provided on an OSREQ STORE.

b. Then, code one or more OSREQ STOREPRT function invocations to provide each individual part of the object from the beginning of the object to the end contiguously in sequence.

c. Finally, code a OSREQ STOREEND function invocation to complete the storage of the object after all of the individual parts have been provided to OAM.
Objects larger than 256 MB
When you plan to use objects larger than 256 MB, consider these recommendations:

- The maximum object size for the DASD layer of the OAM storage hierarchy is 2000 MB.
- The maximum object size for the tape and optical layers of the OAM storage hierarchy is 256 MB (268,435,456 bytes).
- It is impossible to create any backup copies of objects greater than 256M in size. The inability to create backup copies of objects greater than 256M means that the immediate backup copy function is disabled for any object greater than 256M. However, the functionality of the immediate backup copy function is not altered for objects 256M in size or smaller.
- Use Store for objects less than or equal to 256M. For storing objects greater than 256M see the Storebeg, Storeprt, and Storeend functions.
- To retrieve an object greater than 256MB, the object must be retrieved in pieces using multiple OSREQ Retrieves specifying the offset and length (maximum length allowed for each piece is 256MB).
- When using the security exit CBRUXSAE, the functions STOREBEG, STOREPRT, and STOREEND will be identified to the exit as STORE.
DFSMSrmm V1.10 enhancements

The DFSMSrmm enhancements in DFSMS V1R10 provide improvements in the areas of partitioning your tape volumes and system-managed tape libraries, EDGRMMnn PARMLIB commands and options, fast replication to copy the DFSMSrmm CDS, and enterprise system support.

In this chapter we discuss the following enhancements:

- Enabling use of DFSMSdss COPY Services
- SMF record changes
- Audit controls for release processing
- XREPTEXT tailoring via SYSIN
- Volume replacement policies
- Report generator enhancements
- System-managed library and tape volume partitioning
- Common Information Model (CIM) provider
17.1 Enabling use of DFSMSdss COPY Services

With DFSMSrmm release z/OS 1.10, you can request that DFSMSrmm back up the control data set, using the DFSMSdss COPY command.

17.1.1 Backup of DFSMSrmm control data set using DSS COPY

DFSMSrmm no longer relies on concurrent copy and virtual concurrent copy for fast and non-intrusive creation of copies/backups of the DFSMSrmm control data set. DFSMSrmm supports the use of DSS copy services and exploitation of Fast Replication services provided by DASD subsystems. This enables almost instantaneous copies of the control data set to be created enabling reduced recovery time objective and eliminates the impact that use of concurrent copy can have on HyperSwap®.

DFSMSrmm control data set backup can now additionally be performed using copy services. ADRDSSU is used with Fast Replication to enable a control data set “backup” to be created. New parameter options are provided to request this method for DFSMSrmm control data set backup.

Figure 17-1 shows the correct use of the EDGHSKP parameter options.

```
                  BACKUP(AMS) ———DSS———COPY——
```

Figure 17-1  EDGHSKP backup parameter options

The use of the EDGBKUP parameters options are shown in Figure 17-2.

```
                  restore ———BACKUP(AMS) ———DSS———COPY——
```

Figure 17-2  EDGBKUP restore parameter options

Where:

**RESTORE**

Specifies RESTORE to restore and forward recover the control data set from a backup copy. When you specify the BACKUP DD statement, DFSMSrmm restores the control data set. When you specify the JOURNAL DD or SMFIN DD statements, forward recovery is attempted. You can specify both the BACKUP DD statement and the JOURNAL or SMFIN DD statement in the same job step to restore and forward recover the control data set from journal backups and the journal data set or from SMF records.
COPY

Specifies BACKUP(COPY) to use DFSMSdss to create a logical data set copy of the control data set and optionally, a backup of the journal. DFSMSrmm uses DFSMSdss to create the copy of the control data set and IDCAMS to back up the journal. To allow updates to the control data set during backup processing, set up the DFSMSdss concurrent copy environment, virtual concurrent copy environment, or a fast replication with virtual concurrent copy environment. You must have the hardware and software required to establish a concurrent copy session, a virtual concurrent copy session, or fast replication with concurrent copy or virtual concurrent copy.

**Note:** When you use the access method services REPRO command or DFSMSdss without concurrent copy or without fast replication for backup, DFSMSrmm does not allow updates to the control data set during control data set backup.

### Using EDGBKUP to create a CDS backup copy

Create a copy of the control data set using fast replication use the sample JCL as shown in Figure 17-3. The copy of the control data set is created using a new data set name based on the current CDS dsname. The example shown ensures that fast replication is used so that an almost instant copy of the CDS can be created. No journal backup is created in this example. The example is run with DFSMSrmm active so that the MASTER DD is not required. The CDS data set name is obtained from the running DFSMSrmm subsystem. The data set name of the copy is created from the existing CDS data set name. The second qualifier of CDS is renamed to COPYCDS. The copy is almost instant, and this is a non-intrusive copy of the CDS. Updates to the CDS are allowed during backup.

```plaintext
//BACKUP   EXEC PGM=EDGBKUP,PARM='BACKUP(COPY)'
//SYSPRINT DD   SYSOUT=* 
//DSSOPT DD   *
   FR(PREF) CC(ANYREQ) RENAMEU((*.CONTROL.**,*.COPYCDS.**)) REPLACEU
/*
```

*Figure 17-3  Sample EDGBKUP JCL to copy the CDS only*

You have to use following DD statements if you would like to create a backup copy of the DFSMSrmm CDS and journal and to clear the journal:

**BACKUP**

Contains the backup copy of the DFSMSrmm control data set. You specify this data set to run backup processing for the control data set. You can back up directly to tape when you specify the BACKUP(DSS) parameter even when DFSMSdss concurrent copy is not available.

**BACKUP DD is optional when you specify BACKUP(COPY) and your environment is SMS managed.**

**JRNLBKUP**

Contains the backup copy of the DFSMSrmm journal. When you run backup, specify the BACKUP DD statement, the JRNLBKUP DD statement, or both statements. DFSMSrmm uses IDCAMS to back up the journal when you specify the BACKUP(AMS), BACKUP(COPY), or BACKUP(DSS) parameter. You can back up the journal directly to tape when you specify the BACKUP(DSS) parameter when DFSMSrmm is active.

**JRNLBKUP DD is optional.**
DSSOPT
Contains COPY, DUMP, or RESTORE command options used by DFSMSdss during backup processing.

Restriction: Do not specify FR(PREF) without the CONCURRENT keyword or the FR(REQ) keyword, because this causes processing to be intrusive.
If you using EDGBKUP and you have specified the optional JRNGBKUP DD statement to backing up the journal too the journal is not cleared at the end of the backup processing.

Where:

FR
FASTREPLICATION(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).

CC
CONCURRENT(ANYREQUIRED)
Specifies that data is to be processed with virtual concurrent copy.
If virtual concurrent copy cannot be used or fails, the data is processed with cache-based concurrent copy. If neither type of concurrent copy is possible, the data is not processed.

RENAMEU
RENAMEUNCONDITIONAL(on,nn)
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.

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.

REPLACEU
REPLACEUNCONDITIONAL
Specifies that DFSMSdss is to search the target volumes for usable preallocated data sets.
When used with the RENAMEUnconditional keyword, usable preallocated data sets are replaced.
If no preallocated target is found, DFSMSdss attempts to allocate a data set.
The REPLACE and REPLACEUnconditional keywords cannot be specified together.

Note: The parameters in the BACKUP DD statement are dependent on your SMS environment.
Figure 17-4 shows the messages the EDGBKUP and ADRDSSU utilities writes to the SYSPRINT DD.

```
COPY DATASET(INCLUDE(-
  RMM.CONTROL.DSET -
)) OUTDD(backup) SHARE -
  FR(PREF) CC(ANYREQ) RENAMEU((*.CONTROL.**,*.COPYCDS.**)) REPLACEU
ADR101I (R/I)-RI01 (01), TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'COPY'
ADR109I (R/I)-RI01 (01), 2008.073 21:31:09 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED
ADR050I (001)-PRIME(01), DFSMSDSS INVOKED VIA APPLICATION INTERFACE
ADR016I (001)-PRIME(01), RACF LOGGING OPTION IN EFFECT FOR THIS TASK
ADR006I (001)-STEND(01), 2008.073 21:31:09 EXECUTION BEGINS
ADR442I (001)-PREVS(01), DATA SET RMM.CONTROL.DSET PREALLOCATED WITH NEW NAME RMM.COPYCDS.DSET, IN CATLOG UCAT.VSBOX01, ON VOLUME(S): SBOX78
ADR730W (001)-CVSAM(01), CLUSTER RMM.CONTROL.DSET IS OPEN
ADR806I (001)-TOMI (03), DATA SET RMM.CONTROL.DSET COPIED USING A FAST REPLICATION FUNCTION
ADR801I (001)-DDDS (01), DATA SET FILTERING IS COMPLETE. 1 OF 1 DATA SETS WERE SELECTED: 0 FAILED SERIALIZATION AND 0 FAILED NOR OTHER REASONS
ADR734I (001)-DDDS (01), 2008.073 21:31:09 CONCURRENT COPY INITIALIZATION SUCCESSFUL FOR 1 OF 1 SELECTED DATA SETS. SERIALIZATION FOR THIS DATA IS RELEASED IF DFSMSDSS HELD IT. THE INTERMEDIATE RETURN CODE IS 0004
ADR454I (001)-DDDS (01), THE FOLLOWING DATA SETS WERE SUCCESSFULLY PROCESSED RMM.CONTROL.DSET
ADR006I (001)-STEND(02), 2008.073 21:31:10 EXECUTION ENDS
ADR013I (CLTSK(01), 2008.073 21:31:10 TASK COMPLETED WITH RETURN CODE 0004
ADR012I (SCH)-DSSU (01), 2008.073 21:31:10 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0004 FROM: TASK 001
EDG6401I CONTROL DATA SET BACKUP SUCCESSFUL
EDG6901I UTILITY EDGBKUP COMPLETED WITH RETURN CODE 0
```

**Figure 17-4  EDGBKUP utility messages**

**Using EDGHSKP to create a backup copy of your CDS and journal**

To take a backup copy of your DFSMSrmm control data set and journal and to clear the journal only after backup is completed successfully to avoid losing changes that were made since the last backup, you can use the EDGHSKP utility with the new COPY function of the backup function as shown in Figure 17-5.

```
//BACKUP EXEC PGM=EDGHSKP,PARM='BACKUP(COPY)'
//MESSAGE DD DISP=SHR,DSN=RMM.HSKP.MESSAGE
//SYSPRINT DD SYSOUT=*  
//JRNLRKBUP DD UNIT=SYSDA,VOL=SBOX78,DISP=(,CATLG),
//          DSN=RMM.COPYJRNLRKBUP.DSET,SPACE=(TRK,(100,100),RLSE),
//          RECFM=VB,LRECL=9248
//DSSOPT DD *
//FR(PREF) CC(ANYREQ) RENAMEU((*.CONTROL.**,*.COPYCDS.**)) REPLACEU
/*
```

**Figure 17-5  Sample EDGHSKP JCL to backing up the CDS and journal**
You have to use following DD statements if you would like to create a backup copy of the DFSMSrmm CDS and journal and to clear the journal:

**MESSAGE**
Lists the messages the DFSMSrmm subsystem issues during inventory management. This data set is required.

**BACKUP**
Contains the backup copy of the DFSMSrmm control data set. You specify this data set to run backup processing for the control data set. You can back up directly to tape when you specify the BACKUP(DSS) parameter even when DFSMSdss concurrent copy is not available.

**BACKUP DD is optional when you specify BACKUP(COPY) and your environment is SMS managed.**

**JRNLBKUP**
Contains the backup copy of the DFSMSrmm journal. You specify this data set to run backup processing for the journal. DFSMSrmm uses IDCAMS to back up the journal when you specify the BACKUP(AMS), BACKUP(COPY), or BACKUP(DSS) parameter. You can back up directly to tape when you specify the BACKUP(DSS) parameter even when DFSMSdss concurrent copy is not available.

**DSSOPT**
Contains COPY, DUMP, or RESTORE command options used by DFSMSdss during backup processing.

**Note:** In our test case we have not used the optional BACKUP DD because we have specified BACKUP(COPY) and our environment is SMS managed.

### 17.1.2 Restore the DFSMSrmm control data set

Now you can use the SYSIN file to select the target recovery point for forward recovery of the DFSMSrmm control data set. Use the end date and time syntax of the RESTORE command as shown in Figure 17-6.

![Figure 17-6  EDGBKUP RESTORE command syntax](image)

**Where:**

**RESTORE**
Use this command in the SYSIN file to select your chosen forward recovery point.
When you do not specify RESTORE, DFSMSrmm uses all the available complete sets of records to perform forward recovery to the latest available point in time.
TARGETDATE(Date, Time)

Use the TARGETDATE operand to specify the date and time you have selected to be your target forward recovery point in time. You must specify both a date and a time. The date and time are the local date and time as recorded in the journal or SMF records used for forward recovery. Any input records created at a time higher than the value you specify are skipped.

**Date**

Specifies the date in the format set by the DATEFORM operand. The format of the date values is specified by the DATEFORM operand or if DATEFORM is not specified, by the DATEFORM parameter value defined by the installation. For example, if your installation set DATEFORM(J), specify:

```
DATE(2002/123,2002/223)
```

**Time**

Specifies the target local time in the format hh:mm:ss.t The '.t' (tenths of a second) is optional.

**DATEFORM (A | E | I | J | D)**

Use this optional operand to specify the format for the date you enter in the TARGETDATE operand as shown in Table 17-1.

<table>
<thead>
<tr>
<th>Value</th>
<th>Language</th>
<th>Format</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>American</td>
<td>mm/dd/yyyy</td>
<td>12/15/1994</td>
</tr>
<tr>
<td>E</td>
<td>European</td>
<td>dd/mm/yyyy</td>
<td>15/12/1994</td>
</tr>
<tr>
<td>I</td>
<td>ISO</td>
<td>yyyy/mm/dd</td>
<td>1994/12/15</td>
</tr>
<tr>
<td>J</td>
<td>Julian</td>
<td>yyyy/ddd</td>
<td>1994/349</td>
</tr>
<tr>
<td>D</td>
<td>Default</td>
<td>Installation's default in EDGRMMxx</td>
<td>Initially set to Julian</td>
</tr>
</tbody>
</table>

Forward recovery starts at the first record in the JOURNAL or SMFIN file that matches the CDS forward recover start point, or if the start point cannot be found, the first record in sequence after that point. Forward recovery continues until the point in time you select or to the end of the input records, whichever occurs first.

**Using the control data set copy created with DSS COPY**

You can directly use the DFSMSrmm control data set created with the new COPY function or you can take an additional forward recovery using the EDGBKUP utility with the RESTORE parameter as shown in Figure 17-7.

```
//BACKUP EXEC PGM=EDGBKUP,PARM='RESTORE'
//SYSPRINT DD SYSOUT=*
//MASTER DD DISP=SHR,DSN=RMM.COPYCDS.DSET
//JOURNAL DD DISP=SHR,DSN=RMM.HSKP.JNLBKUP
//SYSIN DD *
/ RESTORE TARGETDATE(2008/074,23:59:59.9) DATEFORM(J)
/*
```

Figure 17-7  EDGBKUP with RESTORE parameter and a target forward recovery point in time
When you forward recover to a specific point in time, you have to restart DFRMM with an empty/new/cleared JOURNAL data set because DFSMSrmm will be unable to match your recovered control data set with your existing JOURNAL. If you do not ensure that the JOURNAL is new or empty, DFSMSrmm start-up will issue message EDG2106D and expect the operator to handle that WTOR.

After the completion of EDGBKUP, you should either empty the JOURNAL, or allocate a new JOURNAL prior to starting DFRMM. If you start DFSMSrmm without doing that, you will receive message EDG2106D. The correct action is to reply L for LOCK, and then run EDGHSKP BACKUP of CDS and JOURNAL to cause JOURNAL to be cleared. You can then continue with tape processing.

**Forward recovery the CDS based on SMF records**

To forward recover the CDS from SMF records, you must be using the standard IBM SMF record type for DFSMSrmm.

You can now avoid DFSMSrmm SMF records requiring record types from the user-written range. DFSMSrmm can now use the IBM assigned record type 42 subtypes 22 and 23.

Where:

**SMFIN DD**

Identifies the data set containing the SMF records to be used during restore processing. The only SMF records supported are the IBM SMF type 42 subtype 22. All other records are ignored. SMFIN is mutually exclusive with JOURNAL for RESTORE.

### 17.2 SMF record changes

RMM creates SMF records when requested, but can now optionally use the IBM assigned record type 42 instead of using SMF record types from the user-range. SMFAUD and SMFSEC options each use a record type unless the IBM record type is used, in which case subtypes are used within record type 42. The subtypes used by DFSMSrmm are 22 for audit records and 23 for security records.

To use the new SMF records, you specify the following parameters:

**SMFAUD(YES | nnn)**

Specifies the SMF record type to be used for audit records. Specify YES or a number between 128 and 255 that is different from the value for SMFSEC. The value must conform to standard SMF conventions.

IBM recommends that you do not use an SMF record type number nnn, but instead use the IBM assigned record number by specifying YES. The IBM assigned record number is type 42, and the subtype is 22. You cannot mix SMF record types. For example, you cannot specify YES for SMFAUD and a record type for SMFSEC.

If you do not specify either YES or a number, DFSMSrmm does not produce audit records.

Default: No audit records.

DFSMSrmm ignores SMFAUD on the client system.
**SMFSEC(YES | nnn)** Specifies the SMF record type to be used for security records. Specify YES or a number between 128 and 255 that is different from the value for SMFAUD. The value must conform to standard SMF conventions.

IBM recommends that you do not use an SMF record type number nnn, but instead use the IBM assigned record number by specifying YES. The IBM assigned record number is type 42, and the subtype is 23. You cannot mix SMF record types. For example, you cannot specify YES for SMFAUD and a record type for SMFSEC.

If you do not specify either YES or a number, DFSMSrmm does not produce security records.

Default: No security records.

Figure 17-8 shows the correct use of the EDGHSKP parameter options.

![Figure 17-8](image)

To check the current option settings, use the LISTCONTROL command as shown in Figure 17-9.

![Figure 17-9](image)
The result of the LISTCONTROL is shown in Figure 17-10.

| SMF audit = 42 | Means that you have specified SMSAUD(YES), and DFSMSrmm will write the audit records to SMF record number type 42, and the subtype is 22. |
| SMF security = 42 | Means that you have specified SMSSEC(YES), and DFSMSrmm will write the audit records to SMF record number type 42, and the subtype is 23. |

**Important:** It is still recommended to create SMF records, because these records can be very helpful in a case where recovery of your DFSMSrmm control data set is needed.

### 17.3 Audit controls for release processing

The existing PARMLIB options for ensuring that VRS managed retention is as expected include the VRSMIN and VRSCHANGE operands. If you want to control which volumes are
released, you have to exploit the NOTIFY release option of VLPOOL so that all volumes can be kept in pending release status until the list of volumes is validated against some installation based criteria. The latter is a manual action which is required, but which could be easily automated.

New PARMLIB options are added which can be used to ensure that data is being retained as expected and only released by DFSMSrmm processing if the numbers or percentages of volumes are within policy limits. EDGHSKP VRSEL and EXPROC processing counts the numbers of volumes that are:

- VRS retained; newly and initial count
- Newly assigned since the last VRSEL run
- Planned to be set pending release; Dropped from VRS and dropped by EXPDT
- EXPDT retained initial count

The processing then applies the new PARMLIB options to determine the action to be taken. New messages are added to the MESSAGE file, and are issued if VRSEL or EXPROC are run and the corresponding PARMLIB option has not disabled the function. The new messages list the numbers and percentages of volumes. Regardless of disablement some existing messages have a percentage added for completeness. In addition, based on the set limit and action, an existing message EDG2310I is issued if action is FAIL, and the EDGHSKP return code set if the limit threshold is exceeded. This processing for the new options is identical to the existing processing for VRSMIN.

**17.3.1 EXPDTDROP PARMLIB option**

The syntax for the EXPDTDROP PARMLIB option is:

```
EXPDTDROP(COUNT(count)/PERCENT(%age),action)
```

Use EXPDTDROP to specify a maximum number or percentage of existing expiration date retained volumes that can be dropped from retention, and the action to be taken by DFSMSrmm. DFSMSrmm counts the number of EXPDT retained volumes at the start of inventory management processing and the number of these to be set to pending release.

When you specify count, this is an absolute maximum number of volumes that can be released by a single run of EDGHSKP EXPROC processing. When you specify a percentage, this is a maximum percentage of the existing EXPDT retained volumes that can be released by a single run of EDGHSKP EXPROC processing. This processing occurs each time that you run inventory management EXPROC processing. When VRSEL and EXPROC are run together in a single EDGHSKP run, volumes which are dropped by VRSEL processing are counted only towards the VRSDROP limit, not to the EXPDTDROP limit. EXPDTDROP processing counts all volumes which are retained only by volume expiration date at the start of the run, and also which of these are set to pending release.

An EXPDT-retained volume is one that is not VRS-retained and is not newly assigned. When the EXPROC SYSIN command causes a subset of volumes to be processed, only those volumes are counted. Refer to “EDGHSKP EXPROC” on page 59 for more information. EXPDTDROP processing is intended to provide limited checking for volumes that are not VRS-retained. The volumes would previously have been through VRSRETAIN limit checking and if retained by VRS, the VRSDROP limit checking as well. It considers how many of the EXPDT-retained volumes expire during the EXPROC run. Those volumes that are not set pending release will be considered by EXPDTDROP limit processing on the next run of EXPROC.
Figure 17-12 shows the correct use of the EXPDTDROP parameter option.

<table>
<thead>
<tr>
<th>OPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>-PERCENT(-80--)</td>
</tr>
<tr>
<td>-EXPDTDROP( )</td>
</tr>
<tr>
<td>-COUNT(- count--)</td>
</tr>
<tr>
<td>-PERCENT(-%age--)</td>
</tr>
<tr>
<td>-OFF-</td>
</tr>
</tbody>
</table>

Figure 17-11  EDGRMMnn OPTION operand EXPDTDROP syntax

Where:

- **COUNT**
  Indicates the total number of volumes set pending release during EXPROC processing includes:
  - Those considered by VRSRETAIN limit checking, but not VRS-retained.
  - Those considered by VRSDROP limit checking that were dropped from VRS retention.
  - Those considered by EXPDTDROP limit checking.
  COUNT can be 0 to 2,147,483,647.

- **PERCENT**
  Indicates the percentage of volumes set pending release during EXPROC processing includes:
  - Those considered by VRSRETAIN limit checking, but not VRS-retained.
  - Those considered by VRSDROP limit checking that were dropped from VRS retention.
  - Those considered by EXPDTDROP limit checking.
  PERCENT can be 0 to 100.

- **action**
  Specifies action to control the action DFSMSrmm takes during processing and when the value is exceeded. Action can be FAIL, INFO, OFF, or WARN:

  - **FAIL**
    Issues messages EDG2427 and EDG2428I to the MESSAGE file. When the value is exceeded, DFSMSrmm stops inventory management processing prior to making any updates to volume records and in addition, message EDG2310I is issued, and EXPROC, and any other inventory management, processing ends with return code 12. Updates to data set records might have been made by VRSEL processing.

  - **INFO**
    Issues messages EDG2427 and EDG2428I to the MESSAGE file and processing continues.

  - **OFF**
    Processing of this function is turned off.

  - **WARN**
    Issues messages EDG2427 and EDG2428I to the MESSAGE file. When the value is exceeded, DFSMSrmm sets a minimum return code of 4 and processing continues.
17.3.2 VRSDROP PARMLIB option

VRSDROP processing is intended to provide limited checking for volumes that are already VRS-retained. It considers how many of these existing VRS-retained volumes are removed from VRS retention. Those volumes dropped from VRS-retention can be set pending release, depending on VRS release options and the volume expiration date.

They can also become EXPDT retained and on the next run of EXPROC, they will be considered by EXPDTRDROP limit processing.

You can use VRSDROP to specify a maximum number or percentage of existing VRS retained volumes that can be dropped from vital records retention, and the action to be taken by DFSMSrmm. DFSMSrmm counts the number of VRS retained volumes at the start of vital record selection processing and the number of these dropped by vital record processing. When you specify COUNT, this is an absolute maximum number of volumes that can be dropped by a single run of EDGHSKP VRSEL processing. When you specify PERCENT this is the maximum percentage of the existing VRS retained volumes that can be dropped by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing.

Figure 17-12 shows the correct use of the VRSDROP parameter option.

| OPTION | VRSDROP( | PERCENT(—80—) | INFO— | COUNT(—count—) | WARN— | PERCENT(—%age—) | FAIL— | OFF— |
|———|——|——|——|——|——|——|——|——|

Figure 17-12 EDGRMMnn OPTION operand VRSDROP syntax

Where:

**COUNT** Indicates the absolute maximum number of volumes that can be dropped by a single run of EDGHSKP VRSEL processing.

**PERCENT** Indicates the maximum percentage of the existing VRS-retained volumes that can be dropped by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing. Count can be 0 to 2,147,483,647. Percent can be 0 to 100. The default is PERCENT(10).

**action** Specifies action to control the action DFSMSrmm takes during processing and when the value is exceeded. Action can be FAIL, INFO, OFF, or WARN.

**FAIL** Issues messages EDG2242I and EDG2244I to the MESSAGE file. When the value is exceeded, DFSMSrmm stops inventory management processing prior to making CDS updates and in addition, message EDG2310I is issued, and processing ends with return code 12.
INFO

Issues messages EDG2242I and EDG2244I to the MESSAGE file and processing continues.

OFF

Processing of this function is turned off.

WARN

Issues messages EDG2242I and EDG2244I to the MESSAGE file. When the value is exceeded, DFSMSrmm sets a minimum return code of 4 and processing continues.

17.3.3 VRSRETAIN PARMLIB option

VRSRETAIN processing is intended to provide limited checking for volumes containing newly created data sets that have not yet been processed by inventory management VRSEL. The data sets on the volumes have been created since the start of the last completed VRSEL run. It considers how many of these volumes become VRS-retained during the new VRSEL run. Those volumes which become VRS-retained will be considered by the VRSDROP limit checking in future VRSEL runs.

Those volumes that are not retained by VRS can be set pending release, depending on VRS release options and the volume expiration date. They can also become EXPDT retained and on the next run of EXPROC, they will be considered by EXPDTDROP limit processing.

Use VRSRETAIN to specify a minimum number or percentage of newly assigned volumes that are to be retained by vital records retention, and the action to be taken by DFSMSrmm. A newly assigned volume is one which has a volume assignment date and time which is higher than the run date and time of the previous VRSEL processing. DFSMSrmm counts the number of newly assigned volumes at the start of vital record selection processing and the number of these that become VRS-retained by vital record processing.

When you specify a count, this is an absolute minimum number of volumes that are to be retained by a single run of EDGHSKP VRSEL processing. When you specify a percentage, this is the minimum percentage of the newly assigned volumes that are to be retained by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing.

Figure 17-13 shows the correct use of the VRSRETAIN parameter option.

```
<table>
<thead>
<tr>
<th>OPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>VRSRETAIN(</td>
</tr>
<tr>
<td>-COUNT( count )-</td>
</tr>
<tr>
<td>-PERCENT( %age )-</td>
</tr>
<tr>
<td>-INFO-</td>
</tr>
<tr>
<td>-WARN-</td>
</tr>
<tr>
<td>-FAIL-</td>
</tr>
<tr>
<td>-OFF-</td>
</tr>
</tbody>
</table>
```

Figure 17-13 EDGRMMnn OPTION operand VRSRETAIN syntax

Where:

COUNT

Indicates an absolute minimum number of volumes that are to be retained by a single run of EDGHSKP VRSEL processing.
A newly assigned volume is one that has a volume assignment date and time that is higher than the run date and time of the previous VRSEL processing and that is not VRS-retained. DFSMSrmm counts the number of newly assigned physical and logical volumes at the start of vital record selection processing and the number of these that become VRS retained by vital record processing.

**PERCENT**
Indicates the minimum percentage of the newly assigned volumes that are to be retained by a single run of EDGHSKP VRSEL processing. This processing occurs each time that you run inventory management VRSEL processing. Count can be 0 to 2,147,483,647. Percent can be 0 to 100. The default is PERCENT(80).

**action**
Specifies action to control the action DFSMSrmm takes during processing and when the value is exceeded. Action can be one of:

- **FAIL**
  Issues messages EDG2243I and EDG2245I to the MESSAGE file. When the value is exceeded, DFSMSrmm stops inventory management processing prior to making CDS updates and in addition, message EDG2310I is issued, and processing ends with return code 12.

- **INFO**
  Issues messages EDG2243I and EDG2245I to the MESSAGE file and processing continues.

- **OFF**
  Processing of this function is turned off.

- **WARN**
  Issues messages EDG2243I and EDG2245I to the MESSAGE file. When the value is exceeded, DFSMSrmm sets a minimum return code of 4 and processing continues.

### 17.3.4 Execute VRSEL and EXPROC processing

To check that the new setting is correct, use the RMM TSO LISTCONTROL subcommand and check the result as shown in Figure 17-14.

```
RMM LISTCONTROL OPTION
System options:
PARMLIB Suffix  = 70
Operating mode = P Retention period: Default = 0 Maximum = NOLIMIT
Catalog = 6 hours
....
VRS change = INFO
VRSMIN action = INFO VRSMIN count = 1
VRSDROP action = WARN VRSDROP count = 0 percent = 10
VRSRETAIN action= WARN VRSRETAIN count= 0 percent = 80
EXPDTDROP action= WARN EXPDROP count= 0 percent = 10
Disp DD name = DISPDD Disp msg ID = EDG4054I
....
```

*Figure 17-14  RMM LISTCONTROL showing controls for release processing*

Now we have checked the new parameter settings and ready we are ready to use the EDGHSKP to processing vital records (VRSEL) and performing expiration processing (EXPROC).
We used the JCL as shown in Figure 17-15.

```verbatim
//STEP01    EXEC PGM=EDGHSKP,
               //         PARM='VRSEL,EXPROC,RPTEXT,DSTORE,BACKUP(AMS)'
//SYSPRINT DD SYSPOUT=*  
//MESSAGE DD DISP=SHR,DSN=RMM.HSKP.MESSAGE  
//REPORT DD  DISP=SHR,DSN=RMM.HSKP.REPORT             
//ACTIVITY DD DISP=SHR,DSN=RMM.HSKP.ACTIVITY          
//XREPTEXT DD DISP=SHR,DSN=RMM.HSKP.EXTRACT            
//BACKUP DD   DISP=SHR,DSN=RMM.HSKP.CDSBKUP            
//JRNLBKUP DD DISP=SHR,DSN=RMM.HSKP.JNLBKUP           
//SYSIN DD    *  
RPTEXT RECORDS(X)  
/
/**
   ******************************************************
//STEP02    EXEC PGM=IEBGENER
//SYSPRINT DD DUMMY  
//SYSUT1 DD   DISP=SHR,DSN=RMM.HSKP.MESSAGE  
//SYSUT2 DD   SYSPOUT=*  
//SYSIN DD    DUMMY  
/**
   ******************************************************
```

Figure 17-15  JCL to process vital records (VRSEL) and performing expiration processing (EXPROC)

Where:

**STEP01**  Scheduling all DFSMSrmm Utilities except DSTORE at once.

- **VRSEL**  Processing vital records.
- **EXPROC**  Performing expiration processing.
- **RPTTEXT**  Creating an extract data set. Depending on the SYSIN DD, we are only creating extended extract records (X) at this time.
- **BACKUP**  Backing up the DFSMSrmm control data set and the DFSMSrmm journal.

- **MESSAGE DD**  Lists the messages the DFSMSrmm subsystem issues during inventory management. This data set is required.

- **REPORT DD**  Contains a detailed report of DFSMSrmm vital record specification processing. Specify if you want a report when you have specified the VRSEL parameter.

- **ACTIVITY DD**  Contains detailed information about data set related changes DFSMSrmm makes to the control data set during inventory management. This data set is required when you specify the VERIFY parameter.

- **XREPTEXT DD**  Contains the extract copy of the DFSMSrmm control data set. The extract copy is called the extract data set.

- **BACKUP DD**  Contains the backup copy of the DFSMSrmm control data set.

- **JRNLBKUP DD**  Contains the backup copy of the DFSMSrmm journal.

- **SYSIN DD**  Contains the control cards to tailoring the expiration processing (EXPROC) and creating an extract data set (RPTTEXT).

**STEP02**  Copies the information DFSMSrmm has stored in the message file to your job log so that you have all information in one place.
Figure 17-16 shows all the messages you get if you have used the foregoing JCL.

```
EDG6001I INVENTORY MANAGEMENT STARTING ON 2008/078 AT 14:45:55 - PARAMETERS IN USE ARE
   DATEFORM(J), VRSEL, EXPROC, RPTEXT
EDG6013I THE SYSIN OPTIONS CURRENTLY IN USE ARE
   EXPROC
   EDGSPCLCS(NO)
   VOLUMERANGES('THM000':THM0029')
   RPTEXT
   RECORDS(EXTENDED)
EDG2309I THE PARMLIB OPTIONS CURRENTLY IN USE ARE
   VRSEL(NEW)
   VRJOBNAME(2)
   VRMIN(1,INFO)
   VRSCHANGE(INFO)
   VRSNOP(PERCENT(10),INFO) VRSHELTON(PERCENT(80),INFO)
   EXPDTDROP(PERCENT(10),INFO)
   SMSTAPE(PURGE(ASIS) UPDATE(EXIT,SCRATCH,COMMAND))
   CATRETPD(6)
   UNCATALOG(Y)
   TPRACF(N)
   NOTIFY(N)
   SYSID(SC70)
   CATSYSID()
   RETAINBY(SET)
   MOVEBY(SET)
EDG2229I NUMBER OF VRS RECORDS READ IS 10
EDG2238I NUMBER OF UNUSED VRS RECORDS IS 7
EDG2242I INITIAL NUMBER OF VRS RETAINED VOLUMES = 103 57%
EDG2244I NUMBER OF UNUSED VRS RECORDS TO BE DROPPED = 0 0%
EDG2243I INITIAL NUMBER OF NEWLY ASSIGNED VOLUMES = 100 55%
EDG2245I NUMBER OF NEWLY ASSIGNED VOLUMES TO BE RETAINED = 100 100%
EDG2247I INITIAL NUMBER OF EXPDT RETAINED VOLUMES = 0 0%
EDG2248I NUMBER OF EXPDT RETAINED VOLUMES TO BE DROPPED = 0 0%
EDG2420I PHYSICAL VOLUMES READ = 182 100%
EDG2421I TOTAL VOLUMES READ = 182 100%
EDG2422I PHYSICAL VOLUMES UPDATED = 100 55%
EDG2423I TOTAL VOLUMES UPDATED = 100 55%
EDG2424I PHYSICAL VOLUMES, THIS RUN, KEPT FOR VRS = 100 55%
EDG2425I TOTAL VOLUMES, THIS RUN, KEPT FOR VRS = 100 55%
EDG2435I PHYSICAL VOLUMES SELECTED FOR EXPROC = 0 0%
EDG2436I TOTAL VOLUMES SELECTED FOR EXPROC = 0 0%
EDG2442I PHYSICAL VOLUMES, THIS RUN, SET PENDING RELEASE = 0 0%
EDG2451I TOTAL VOLUMES RETURNED TO SCRATCH = 0 0%
EDG2452I TOTAL NUMBER OF SCRATCH RECORDS WRITTEN = 0 0%
EDG2429I MAIN INVENTORY MANAGEMENT UPDATES HAVE COMPLETED SUCCESSFULLY
EDG2307I INVENTORY MANAGEMENT TASK VRSEL COMPLETED SUCCESSFULLY
EDG2307I INVENTORY MANAGEMENT TASK EXPROC COMPLETED SUCCESSFULLY
EDG2307I INVENTORY MANAGEMENT TASK RPTEXT COMPLETED SUCCESSFULLY
EDG6901I UTILITY EDGHSKP COMPLETED WITH RETURN CODE 0
```

Figure 17-16  EDGHSKP messages
Where:

**EDG2242I**  
**INITIAL NUMBER OF VRS RETAINED VOLUMES = number percent**

**Explanation:** DFSMSrmm issues this message to the MESSAGE file during VRSEL inventory management. This message is issued for information only.

**In the message text:**
- **number** Indicates the number of physical and logical volumes retained by VRS control when EDGHSKP VRSEL starts. This number includes volumes that are VRS retained only because of RETAINBY(SET) and another volume was VRS retained, and volumes that contain VRS retained data sets and those that are VRS retained.
- **percent** Indicates the percentage of the total number of physical and logical volumes read from the CDS.

**System action:** DFSMSrmm inventory management processing continues.

**Operator response:** None.

**System programmer response:** You can use this information to help determine a good setting for the VRSDROP PARMLIB option.

**Source:** DFSMSrmm

**Detecting Module:** EDGVRECM

**EDG2243I**  
**INITIAL NUMBER OF NEWLY ASSIGNED VOLUMES = number percent**

**Explanation:** DFSMSrmm issues this message to the MESSAGE file during VRSEL inventory management. This message is issued for information only.

**In the message text:**
- **number** Indicates the number of physical and logical volumes that are newly assigned when EDGHSKP VRSEL starts. A newly assigned volume is one that has a volume assignment date and time that is higher than the run date and time of the previous VRSEL processing and is not VRS retained.
- **percent** Indicates the percentage of the total number of physical and logical volumes read from the CDS.

**System action:** DFSMSrmm inventory management processing continues.

**Operator response:** None. System programmer response: You can use this information to help determine a good setting for the VRSRETAIN PARMLIB option.

**Source:** DFSMSrmm

**Detecting Module:** EDGVRECM

**Routing Code:** -

**Descriptor Code:** -

**EDG2244I**  
**NUMBER OF VRS RETAINED VOLUMES TO BE DROPPED = number percent**

**Explanation:** DFSMSrmm issues this message to the MESSAGE file during VRSEL inventory management. This message is issued for information only.
In the message text:

**number** Indicates the number of physical and logical volumes to be removed from VRS control. This number includes volumes that were VRS retained only because of RETAINBY(SET) and another volume was VRS retained, and volumes that contained VRS retained data sets or were VRS retained.

**percent** Indicates the number of physical and logical volumes removed from VRS control expressed as a percentage of the physical and logical volumes VRS retained at the start of VRSEL processing. See message EDG2242I for the initial count of VRS retained volumes.

**System action:** DFSMSrmm inventory management processing continues based on the action value specified for the VRSDROP PARMLIB option. If the VRSDROP limit is reached, the EDGHSKP return code is set based on the VRSDROP action:

- For FAIL, the return code is 12 and no volumes are dropped from VRS retention.
- For WARN, the return code is 4.
- For INFO, there is no effect on the return code.

**Operator response:** None.

**System programmer response:** You can use this information to determine a good setting for the VRSDROP PARMLIB option. If the limit is triggered, you should review the inventory management processing to determine if an error occurred and why the limit is triggered. If the limit is triggered incorrectly, you might have to adjust your limit.

**Source:** DFSMSrmm

**Detecting Module:** EDGVRECM

**EDG2245I**  
**NUMBER OF NEWLY ASSIGNED VOLUMES TO BE RETAINED = number**  
**percent**

**Explanation:** DFSMSrmm issues this message to the MESSAGE file during VRSEL inventory management. This message is issued for information only.

In the message text:

**number** Indicates the number of physical and logical volumes to be retained by VRS control. This number includes all newly assigned volumes that are now VRS retained. This might be because of RETAINBY(SET) and another volume was VRS retained, and includes all volumes that contain VRS retained data sets or are now VRS retained.

**percent** Indicates the number of physical and logical volumes retained by VRS control expressed as a percentage of the newly assigned physical and logical volumes VRS at the start of VRSEL processing. See message EDG2243I for the initial count of newly assigned volumes.

**System action:** DFSMSrmm inventory management processing continues based on the action value specified for the VRSRETAIN PARMLIB option. If the VRSRETAIN limit is reached, the EDGHSKP return code is set based on the VRSRETAIN action:
- For FAIL, the return code is 12 and DFSMSrmm makes no updates to the CDS as a result of VRSEL processing.
- For WARN, the return code is 4.
- For INFO, there is no effect on the return code.

**Operator response:** None.

**System programmer response:** You can use this information to determine a good setting for the VRSRETAIN PARMLIB option. If the limit is triggered, you should review the inventory management processing to determine if an error occurred and why the limit is triggered. If the limit is triggered incorrectly, you might have to adjust your limit.

**Source:** DFSMSrmm

**Detecting Module:** EDGVRECM

**EDG2427I**

**Explanation:** DFSMSrmm issues this message to the MESSAGE file during inventory management expiration processing. This message is issued for information only.

In the message text:

- **number**
  This is the number of processed physical and logical volumes that are retained by EXPDT retention. This number includes volumes that are expiration retained only because of RETAINBY(SET) and another volume was EXPDT retained, and volumes that are EXPDT retained. VRS retained and newly assigned volumes are excluded.

- **percent**
  This is the percentage of the total number of processed physical and logical volumes read from the CDS by EXPROC processing and as restricted by EXPROC SYSIN command. System action: DFSMSrmm inventory management processing continues.

**Operator response:** None.

**System programmer response:** You can use this information to help determine a good setting for the EXPDTDROP PARMLIB option.

**Source:** DFSMSrmm

**Detecting Module:** EDGMUPD

**EDG2428I**

**Explanation:** DFSMSrmm issues this message to the MESSAGE file during VRSEL inventory management. This message is issued for information only.

In the message text:

- **number**
  This is the number of physical and logical volumes to be removed from EXPDT retention. This number includes volumes that were expiration retained only because of RETAINBY(SET) and another volume was EXPDT retained, and volumes that were EXPDT retained. Volumes removed from VRS retention during this run and those newly assigned are excluded.
percent This is the number of physical and logical volumes removed from EXPDT retention expressed as a percentage of the physical and logical volumes retained by expiration date at the start of EXPROC processing.

System action: DFSMSrmm EXPROC inventory management processing continues based on the action value specified for the EXPDTDROP PARMLIB option. If the EXPDTDROP limit is reached, the EDGHSKP return code is set based on the EXPDTDROP action:

- For FAIL, the return code is 12 and no volumes are dropped from EXPDT retention and no volume updates occur as a result of VRSEL or any other inventory management processing. VRSEL data set updates have been completed.
- For WARN, the return code is 4.
- For INFO, there is no effect on the return code.

Operator response: None.

System programmer response: You can use this information to help determine a good setting for the EXPDTDROP PARMLIB option. If the limit is triggered, you should review the inventory management processing to determine if an error occurred and why the limit is triggered. If the limit is triggered incorrectly, you might have to adjust your limit.

Source: DFSMSrmm
Detecting Module: EDGMUPD

17.4 XREPTEXT tailoring via SYSIN

You can create an extract data set to use as input for creating reports. You can select the records to be extracted from the DFSMSrmm control data set. When you specify the RPTEXT command in the EDGHSKP SYSIN file, you can select which records are extracted or, you can use the DD name to determine if extended records are created. When you specify the XREPTEXT DD statement, the extract contains only extended records that are a combination of data set information and volume information that can be useful as input to DFSMSrmm reporting tools. When you code only the REPTEXT DD, all records other than the extended records are created.

Report extract processing now checks for the existence of the RPTEXT command in the SYSIN file, and if RPTEXT command is found writes the selected extract records to either REPTEXT or XREPTEXT DD. XREPTEXT DD is checked for and used in preference.

When there is no RPTEXT command in SYSIN or the RPTEXT command specifies no RECORDS operand:

<table>
<thead>
<tr>
<th>RPTEXT</th>
<th>RMM creates all records except extended records in the extract (unchanged)</th>
</tr>
</thead>
<tbody>
<tr>
<td>XREPTEXT</td>
<td>RMM creates only extended records (changed).</td>
</tr>
</tbody>
</table>

The EDG6013I message is used to list the RPTEXT command options from SYSIN. This is done in the same way that EXPROC is already listed.
17.4.1 Using the RPTEXT SYSIN command

The EDGHSKP parameters EXPROC and RPTEXT cause the SYSIN file to be checked for opened and supported EXPROC and RPTEXT commands processed. Processing of the SYSIN file allows any supported and correctly specified command to be accepted but ignored if not required. Only a single EXPROC command is supported. Only a single RPTEXT command is supported.

Example 17-1 shows the format of the EDGHSKP EXPROC SYSIN parameters.

Example 17-1   EDGHSKP RPTEXT - SYSIN parameters

Where:

RPTEXT  Use the RPTEXT command in the SYSIN file to select the records you want to be written to the report extract file. If you only specify RPTEXT or RPTEXT RECORDS(), the default of the REPTEXT or XREPTEXT DD records will be created. The following defaults are for using the different DD statements:

REPTEXT    BIN, DATASET, OWNER, PRODUCT, RACK, VOLUME and VRS

XREPTEXT   EXTENDED
RECORDS  Use this operand to select the records you want written by report extract processing. You can specify one or more of the following values for:

BIN     Creating extract data set Storage Location Shelf Location records: EDGRSEXT.
DATASET Creating extract data set Data Set Name records: EDGRDEXT.
EXTENDED Creating extract data set Extended Data Set Name records: EDGRXEXT
OWNER   Creating extract data set Owner records: EDGROEXT.
PRODUCT Creating extract data set Software Product records: EDGRPEXT
RACK     Creating extract data set Rack records: EDGRREXTStorage Location Shelf Location records: EDGRSEXT
VOLUME   Creating extract data set Volume report record: EDGRVEXT
VRS      Creating extract data set Vital Record Specification records: EDGRKEXT.

Note: Use the dash (-) at the end of a line if you need to specify more operands matching in one line and to be continued on the next line.

17.4.2 JCL for creating an extract data set

To create an extract data set, specify the RPTEXT parameter. To create an extract data set that contains extended records, uncomment the XREPTEXT DD statement or code the RPTEXT command with the RECORDS(X) operand in the SYSIN file. To create all records available, we used a job with JCL as shown in Figure 17-17.

```
//HSKP    EXEC PGM=EDGHSKP,
//        PARM='RPTEXT,DATEFORM(E)'
//MESSAGE DD   DISP=SHR,DSN=RMM.HSKP.MESSAGES
//REPTEXT DD   DISP=SHR,DSN=RMM.MASTER.EXTRACT
//XREPTEXT DD   DISP=SHR,DSN=RMM.MASTER.EXTENDED.EXTRACT
//SYSIN   DD   *
RPTEXT RECORDS(B,D,E,O,P,R,S,V)    -
/*
```

Figure 17-17  Sample JCL to create an extract data set

Note: When you specify both XREPTEXT and REPTEXT, DFSMSrmm uses the XREPTEXT DD statement.

Figure 17-18 shows the messages that you get if you are using the JCL shown previously for creating all available records at once.
17.5 Volume replacement policies

New limits and policies are used to determine if the REPLACE actions should be set:

- MEDINF commands in PARMLIB can now include REPLACE policy settings.
- One setting is used for each media type and recording format combination.

The place and timing for the checks for the limits is unchanged. In addition:

- DFSMSrmm now checks at OPEN time to see if a volume to be used for output processing has the REPLACE release action set and rejects the volume. Note that if the REPLACE action is pending, the volume must be pending release and already subject to rejection by RMM O/C/OV processing.
- REPLACE release action can be for scratch volumes, and an owner set for the volume and the volume set pending release, thus setting the release action pending.
- DFSMSrmm intercepts WTO messages that contain SARS MIM information and set the REPLACE release action for any volume where the recommended recovery action is to copy off the data or replace the volume. See WTO IEA486E. In addition, for scratch volumes, an owner is set for the volume and the volume set pending release, thus setting the release action pending.

17.5.1 USERMOD to EDGMUPD

Any installation that has installed a USERMOD to EDGMUPD to change the REPLACE release action checking must perform an action.

If you are using the USERMOD to EDGMUPD to set LIMIT=n, where n is shipped as x'00000001' you can no longer use the USERMOD. In the past you could use the USERMOD to override the shipped value. The USERMOD could disable volume replacement action setting by setting a value of x'00000000'. The USERMOD was first available with APAR OW43870 and might be called RMREPRL.

You must now use the function of MEDINF REPLACE in PARMLIB to implement any override or disablement. The USERMOD no longer fits to EDGMUPD, and DFSMSrmm processing is based solely on MEDINF REPLACE.

To disable replacement of IBM media, you can specify the new MEDINF command in the EDGRMMnn PARMLIB member as shown in Figure 17-19.

```
MEDINF NAME(IBM) REPLACE(PERM(0))
```

Figure 17-19  Disable tape replacement
To override the DFSMSrmm built-in default processing of PERM(1), use any of the available REPLACE operand values.

17.5.2 Overview

DFSMSrmm supports policies for volume replacement. You define them in the MEDINF commands in PARMLIB. This enables you to define different replacement policies based on media type and recording format. The replacement policies are implemented for all types of volume other than logical volumes, and for both private and scratch volumes, but not for those which are pending release. Until you define one or more MEDINF commands in PARMLIB with the REPLACE operand DFSMSrmm processing is unchanged. A hard-coded value of PERM(1) is used.

The policies are implemented during inventory management expiration processing. This enables the release action for private volumes that meet or exceed one or more of the threshold values to be changed from RETURN to REPLACE so that the volume is identified for replacement. Later, when the data expires, and the volume released, the volume pending actions are set from the release actions.

- It also enables scratch volumes to be processed by the replacement policies and if any threshold is met the release action is set to REPLACE and the scratch volume is changed to master status pending release.

When inventory management sets the release action, the volume last change userID is set to ‘*hkp’ (this is unchanged) and if the volume is scratch, the volume owner is set to ‘REPLACE’. Any other subcommand processing sets the last change user ID as normal and any other fields based on the subcommand operands and the command issuers ID.

You can report on volumes that need to be replaced:

- Use the RMM SEARCHVOLUME subcommand with the ACTION(REPLACE) operand; this lists volumes which are ready to be replaced. You can physically replace them and confirm the replace action (RMM CV volser CRLSE(REPLACE)) or you can delete them from DFSMSrmm (RMM DV volser REPLACE) and destroy the volumes. Alternatively, if you do not want to replace or destroy a volume marked for replacement you can turn off the replace action with (RMM CV volser REPLACE(NO)). To instead have the replacement policy re-applied you can reclaim the volume from pending release (RMM CV volser RETPD(1) RELEASEACTION(SCRATCH)); when the volume is next processed by EXPROC the REPLACE policy is used to determine if the REPLACE action is required.

- Use the SEARCHVOLUME subcommand with the RELEASEACTION(REPLACE) operand; this lists volumes which should be replaced. If the data does not expire soon you should consider copying the data to a new volume and then release the volume so that it can be replaced or destroyed. You can use a product such as IBM Tivoli® Tape Optimizer to copy the data to a newer volume.

- DFSMSrmm also provides sample reports in the Report Generator, which we discuss later.

EREP, and the IBM Service Agent (replaced Service Director™) provide reporting capability, based on LOGREC and SMF records, and can generate a report of volumes that should be replaced. There is no current link between this report and RMM. RMM tracks volume usage and I/O errors and now provides support for volume replacement policies. You could manually use EREP or Service Agent (with IBM Service support) reports as a base for setting the REPLACE release action.
17.5.3 Defining media information using MEDINF

You can use the MEDINF command to define media characteristics for OEM media products. The MEDINF values convert external values for media type and recording technology to internal values when supplied in the DFSMSrmm TSO subcommand input, as well as externalizes recorded values in reports and DFSMSrmm TSO subcommand output. To connect a volume to a specific media information entry, assign the media information name with the RMM TSO subcommands ADDVOLUME and CHANGEVOLUME using the MEDINF operand.

Example 17-1 shows the correct use of the MEDINF command used in the EDGRMMnn PARMLIB member.

Example 17-2  MEDINF command syntax

```
MEDINF NAME(medinf_name)

-MEDIATYPE(mediatype_id, mediatype_name)
-CAPACITY(capacity)

-RECORDINGFORMAT(recordingformat_id, recordingformat_name)

-REPLACE(replace_options)

replace_options:

-PERM(count)
-TEMP(count)
-AGE(years)
-WMC(count)
```
Where:

**NAME(medinf_name)** Specifies a name for the media information. You can specify a value between 1 and 8 alphanumeric characters. This value is used to match the recorded volume information to the MEDINF installation defined media information. The values defined for MEDIATYPE, RECORDINGFORMAT, and CAPACITY are taken if the assigned media information for a volume matches medinf_name.

Default: None.

**MEDIATYPE(mediatype_id,mediatype_name)**

Use this operand to identify the internally-used mediatype_id to be converted to the externally-used mediatype_name in reports and in the output of the DFSMSrmm TSO subcommands. Also, use this operand to identify the externally-used mediatype_name to be converted to the internally-used mediatype_id that is recorded in the control data set when used as input to RMM TSO subcommands.

When you specify multiple MEDINF entries that use the same mediatype_id, but with different mediatype_name values, the first such MEDINF entry is used for the internal-to-external conversion. The remaining values are synonyms used for external-to-internal conversions.

When you specify synonym values, do not code the CAPACITY or REPLACE operands.

`mediatype_id` Specifies a number in the range from 1 to 255.

`mediatype_name` Specifies from 1 to 8 alphanumeric or national characters.

Default: None.

**RECORDINGFORMAT(recordingformat_id,recordingformat_name)**

Use this operand to identify the internally-used recordingformat_id to be converted to the externally-used recordingformat_name in reports and in the output of the DFSMSrmm TSO subcommands. Also, use this operand to identify the externally-used recordingformat_name to be converted to the internally-used recordingformat_id that is recorded in the control data set when used as input to RMM TSO subcommands.

When you specify multiple MEDINF entries that use the same recordingformat_id, but with different recordingformat_name values, the first such MEDINF entry is used for the internal-to-external conversion. The remaining values are synonyms used for external-to-internal conversions.

When you specify synonym values, do not code the CAPACITY or REPLACE operands.

`recordingformat_id` Specifies a number in the range from 1 to 255.

`recordingformat_name` Specifies from 1 to 8 alphanumeric or national characters.

Default: None.

**CAPACITY(capacity)** Specifies the media capacity in MB available on the non-IBM media that has a media type mediatype_id and a recording format recordingformat_id. You can specify a value from 0 to 2147483647.
Do not specify the CAPACITY operand on a MEDINF command that specifies the same MEDINF NAME and MEDIATYPE mediatype_id and RECORDINGFORMAT recordingformat_id as an earlier MEDINF command. If you do, an information message EDG0243I is issued at start-up time, the CAPACITY operand is ignored, and the capacity from the earlier entry used instead. When you use synonyms, use the CAPACITY operand only when the combination of MEDINF NAME, MEDIATYPE mediatype_id, and RECORDINGFORMAT recordingformat_id are unique.

Default: None.

REPLACE(PERM | TEMP | AGE | WMC)

Use this operand to identify the policies for replacement of volumes based on such values as read and write errors, age, and numbers of times written. Volumes are identified for replacement when one or more of the policy values are met or exceeded.

Do not specify the REPLACE operand on a MEDINF command that specifies the same MEDINF NAME and MEDIATYPE mediatype_id and the RECORDINGFORMAT recordingformat_id as an earlier MEDINF command. If you do, an information message EDG0243I is issued at start-up time, the REPLACE operand is ignored, and the replace policy from the earlier entry is used instead. When you use synonyms, only use the REPLACE operand if the combination of MEDINF NAME, MEDIATYPE mediatype_id and RECORDINGFORMAT recordingformat_id are unique.

When you add or modify a volume replacement policy, DFSMSrmm does not implement it retrospectively. Volumes that are already set with the REPLACE release action are not reprocessed and the action reset. In order to have DFSMSrmm reconsider the policy for all non-pending release volumes, you first have to manually change the REPLACE release action to SCRATCH, and then run EXPROC processing.

When you do not specify replacement policies with MEDINF, DFSMSrmm uses the built-in value of REPLACE(PERM(1)).

You can specify a global REPLACE based on MEDINF NAME. For example:

MEDINF NAME(VEND1) REPLACE(PERM(1) AGE(20))

This policy is applied to all other MEDINF commands that specify the same NAME(VEND1), but do not specify the REPLACE operand. This is enough to override the built-in value of PERM(1) for all media with a MEDINF name of VEND1.

For IBM media, the hard-coded default of PERM(1) can be overridden by coding a single MEDINF in PARMLIB. For example:

MEDINF NAME(IBM) REPLACE(PERM(0))

This example disables replacement for IBM media. To implement your own chosen replacement policies, customize the command, and then add it to DFSMSrmm PARMLIB.

You can also specify a media-level policy for all media of a certain type. You do this by specifying the MEDIATYPE operand without the RECORDINGFORMAT operand, such as the following example:

MEDINF NAME(IBM) MEDIATYPE(1,CST) REPLACE(AGE(30))
This example sets a default value to be used for all media with a mediatype_id of 1.

You can override the global and media level policies by specifying the REPLACE operand for a specific media type and recording format combination.

No matter how you define the MEDINF commands in PARMLIB, DFSMSrmm LISTCONTROL always lists a REPLACE policy for each MEDINF entry. The way DFSMSrmm determines the values is:

- REPLACE is specified on the MEDINF command.
- A media level REPLACE policy was specified.
- A global REPLACE policy was specified.
- The default of REPLACE(PERM(1)) is used.

When inventory management determines the policy to use, it does so as follows:

- An extract An exact match to the combination of MEDINF NAME, MEDIATYPE mediatype_id, and RECORDINGFORMAT recordingformat_id.
- A match on the combination of MEDINF NAME and MEDIATYPE mediatype_id to a media-level policy.
- A match to a global policy based on MEDINF NAME alone.
- The default of REPLACE(PERM(1)) is used.

**PERM(count)**

Use this operand when your policy is to be based on permanent I/O errors. DFSMSrmm compares your value with the sum of permanent read and write errors over the life of the volume. The value range is 0 to 99999. Specify 0 to disable replacement based on permanent errors.

There is no default value.

**TEMP(count)**

Use this operand when your policy is to be based on temporary I/O errors. DFSMSrmm compares your value with the sum of temporary read and write errors over the life of the volume. The value range is 0 to 99999. Specify 0 to disable replacement based on temporary errors.

There is no default value.

**AGE(years)**

Use this operand when your policy is to be based on the age of the volume. DFSMSrmm compares your value with the volume creation, and current date and time. The value range is 0 to 99999. Specify 0 to disable replacement based on age.

There is no default value.

**WMC(count)**

Use this operand when your policy is to be based on how many times the volume is mounted for output and written to. DFSMSrmm compares your value with the write mount count for the volume. The value range is 0 to 99999. Specify 0 to disable replacement based on volume write mount count.

There is no default value.
17.5.4 Testing media information: MEDINF

We have added some MEDINF commands to our EDGRMMnn PARMLIB member as shown in Figure 17-20 to change the default settings for all of our known IBM 3592 cartridges. We have defined that this cartridges should be replaced:

**PERM** If there are 8 or more permanent write I/O errors reported

**TEMP** If there are 1000 or more temporary write I/O errors reported

**WMC** If the volume is 2500 or more times mounted for output and written to

**AGE** If the volume creation date is more than 5 years in the past

![Figure 17-20 MEDINF commands in EDGRMMnn to specify the IBM media](image)

DFSMSrmm provides sample MEDINF commands to support SUN/StorageTek media and the Open systems media. Refer to the members EDGMEDST and EDGMEDOP in the SAMPLIB library.
Before we re-started DFSMSrmm, we checked the current MEDINF settings using the RMM TSO LISTCONTROL subcommand and we got the information shown in Figure 17-21. As you can see, there is no information available and DFSMSrmm will use the hard coded values for all IBM medias.

If you have specified duplicate MEDINF operands, you get a message as you can see in Figure 17-22.

Where:

EDG0242E DUPLICATE MEDINF COMMAND FOR NAME name MEDIATYPE type_id,type_name RECORDINGFORMAT format_id,format_name

Explanation: This message is issued during initialization when you have specified multiple MEDINF commands and have duplicate commands.

In the message text:

- name: Is the value specified for the MEDINF NAME operand.
- type_id: Is a number in the range 0 to 255; it is the internal value specified for the MEDIATYPE operand.
- type_name: Is the external value specified for the MEDIATYPE operand.
- format_id: Is a number in the range 0 to 255; it is the internal value specified for the RECORDINGFORMAT operand.
- format_name: Is the external value specified for the RECORDINGFORMAT operand.

System action: DFSMSrmm initialization stops. This message is followed by message EDG0215D.

Operator response: Notify the system programmer. Reply to message EDG0215D as directed. Restart DFSMSrmm when the system programmer has corrected the error.

Note: If you specify the same retention criteria for all media matching the same MEDINF NAME(name), you can specify a global setting such as this:

MEDINF NAME(IBM) REPLACE(PERM(8) WMC(2500) TEMP(1000) AGE(5))
System programmer

Ensure that each MEDINF command specifies a unique combination of NAME, MEDIATYPE, and RECORDINFORMAT values. DFSMSrmm ignores the MEDINF command if initialization continues.

Source: DFSMSrmm

Detecting Module: EDGPARM

Routing Code: 3

Descriptor Code: 3

Figure 17-23 shows the result of the RMM TSO LISTCONTROL subcommand after DFSMSrmm is re-started using the new MEDINF definitions.

<table>
<thead>
<tr>
<th>Name</th>
<th>Media type</th>
<th>Recording format</th>
<th>Capacity (MB)</th>
<th>Perm</th>
<th>Temp</th>
<th>WMC</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM</td>
<td>5 3592JA</td>
<td>6 EFMT1</td>
<td>286</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>5 3592JA</td>
<td>7 EFMT2</td>
<td>500</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>5 3592JA</td>
<td>*</td>
<td>500</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>6 3592JW</td>
<td>6 EFMT1</td>
<td>286</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>6 3592JW</td>
<td>7 EFMT2</td>
<td>500</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>6 3592JW</td>
<td>*</td>
<td>500</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>7 3592JJ</td>
<td>6 EFMT1</td>
<td>100</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>7 3592JJ</td>
<td>7 EFMT2</td>
<td>60</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>7 3592JJ</td>
<td>*</td>
<td>60</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>8 3592JR</td>
<td>6 EFMT1</td>
<td>100</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>8 3592JR</td>
<td>7 EFMT2</td>
<td>60</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>8 3592JR</td>
<td>*</td>
<td>100</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>9 3592JB</td>
<td>7 EFMT2</td>
<td>660</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>9 3592JB</td>
<td>*</td>
<td>660</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>10 3592JX</td>
<td>7 EFMT2</td>
<td>660</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>IBM</td>
<td>10 3592JX</td>
<td>*</td>
<td>660</td>
<td>8</td>
<td>1000</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 17-23  RMM LISTCONTROL MEDINF result

After the change of the MEDINF for our IBM 3592 cartridges, we write two new data sets to an existing volume to see what will be changed, but DFSMSrmm will not change the capacity of an existing volume. The addition of a new volume in Figure 17-24 shows us the new settings that DFSMSrmm is using. In this case the capacity is only 286 MB as we have specified it in our test example of the MEDINF command.
After the EDGHSKP expiration processing, some of the volumes are now pending release, because we have reduced the write mount count (WMC) to a value of 3 and some of the volumes (for example, volume THS013 as shown in Figure 17-25), are used more often.

**Figure 17-24**  RMM LISTVOLUME statistics with new capacity usage

**Figure 17-25**  List of a volume must be replaced

```plaintext
TSO RMM ADDVOLUME THS029 STATUS(VOLCAT)
READY
TSO RMM LISTVOLUME THS029 STAT
Statistics:
Number of data sets = 0  Data set recording= ON
Volume usage(KB)= 0  Use count = 0
Capacity(MB) = 286  Percent full = 0
Date last read =  Date last written =
Drive last used =  Write mount count = 0
Volume sequence = 1  Media name = 3480
Previous volume =  Next volume =
Product number =  Level = V R M
Feature code =
Error counts:
Temporary read = 0  Temporary write = 0
Permanent read = 0  Permanent write = 0
```

```plaintext
RMM LISTVOLUME THS013 ALL
Volume information:
Volume = THS013  VOL1 =  Rack = THS013  Owner = MHLRES7
Type = PHYSICAL  Stacked count = 0  Jobname = MEDINF
Worldwide ID =  WORM = N
Creation:  Date = 2008/024  Time = 14:08:35  System ID = SC64
Assign:  Date = 2008/081  Time = 14:16:51  System ID = SC70
...
Status = USER  Availability = Pending Release  Label = SL
Current label version =  Required label version =
Media information: IBM
Density = IDRC  Type = 3592JA  Format = EFMT2  Compaction = YES
....
Action on release:
Scratch immediate = N  Expiry date ignore = N
Scratch = N  Replace = Y  Return = N  Init = N  Erase = N  Notify = N
Actions pending:
Scratch = N  Replace = Y  Return = N  Init = N  Erase = N  Notify = N
Storage group = SGLIB2
...
Statistics:
Number of data sets = 2  Data set recording= ON
Volume usage(KB)= 28  Use count = 7
Capacity(MB) = 476837  Percent full = 0
```

**Figure 17-25**  List of a volume must be replaced
17.5.5 TSO subcommand ADDVOLUME and CHANGEVOLUME updates

There are some new operands for the RMM TSO ADDVOLUME and CHANGEVOLUME subcommand as shown in Example 17-3. They can be used to set correct values when you are adding a new volume. These new operands are all introduced with the new volume replacement policies available in this release. All other operands not related to the new policies are not shown in this book. See 17.5, “Volume replacement policies” on page 218.

Example 17-3 New MEDINF operands for the ADDVOLUME subcommand

```
---ADDVOLUME---
     |---CHANGEVOLUME---|
     |---new_medinf_operands---|

new_medinf_operands:

---CAPACITY(--------nn-mb--------)--
     |---READ(--------count--------)|
     |---WRITE(--------count--------)|

---MEDIATYPE(--------IBM_defaults--------)
     |---medinf_mediatyp--------|

---OPENCOUNT(--------count--------)

---RECODINGFORMAT(--------IBM_defaults--------)
     |---medinf_recordingformat--------|

for CHANGEVOLUME only:

---REPLACE(--------1--------)
     |---NO--------|
     |---YES--------|
```
The option REPLACE can only be used if the volume status is currently “Pending release”. In all other cases you get the message EDG3287I RELEASE ACTION SCRATCH IS NOT PENDING.

Where:

**CAPACITY(nn-mb)** You can use this operand to specify the volume capacity in megabytes (MB). DFSMSrmm normally sets the capacity of a volume based on the media type and the recording format, or you can manually set the capacity if the capacity cannot be determined using the media type and recording format. The MEDINF PARMLIB commands define the capacity of different combinations of media type and recording formats. There are built-in capacity values for IBM media types. Specify a value between 0 and 2 147 483 647.

**ERROR(READ(count),WRITE(count))**
You can use this operand to specify new error count values for read and write errors. These are assumed to be the permanent errors for the volume.
You can specify an absolute value or an incremental value. Incremental values are specified as follows: WRITE(+23). If the value specified causes the count to exceed 65 535, the maximum value of 65 535 is set.

**MEDIATYPE(IBM defaults / medinf_mediatype)**
Specifies the volume’s physical media type.

- **IBM defaults** The IBM defaults are not listed in this book.
- **medinf_mediatype** Specifies a non-IBM media type if your installation definition contains media information for medinf_mediatype that matches the media information assigned to the volume. When you change the media type, DFSMSrmm sets the volume capacity based on the matching media information.

**MEDINF(medinf_name)**
Specifies the assigned installation-defined media information to the volume. This value is one-to-eight alphanumeric characters and must be defined once in your installation. You can use the LISTCONTROL subcommand with the MEDINF operand to display the media information. For more information about MEDINF, see z/OS DFSMSrmm Implementation and Customization Guide. When you change the installation-defined media information, DFSMSrmm sets the volume capacity based on the matching media information. Default: None.

**OPENCOUNT(count | +nn)**
Use this operand to specify how many times any data set on the volume has been opened. You can specify an absolute value or an incremental value. Incremental values are specified as follows: OPENCOUNT(+1). If the value specified causes the count to exceed 65 535, the maximum value of 65 535 is set.

**PERCENT(percent)**
Use this operand to specify how full the volume is. You can specify a value between 0 and 100.
RECORDINGFORMAT(*IBM defaults / medinf_recordingformat*)

Specifies the basic recording format for tape volumes.

**IBM defaults**

The IBM defaults are not listed in this book.

**medinf_recordingformat**

Specifies a non-IBM media recording format if your installation definition contains media information for *medinf_recordingformat* that matches the media information assigned to the volume. When you change the recording format, DFSMSrmm sets the volume capacity based on the matching media information.

REPLACE(YES | NO)

Use the REPLACE operand to change the setting of the replace action.

You specify YES to set on the REPLACE action when a volume is pending release with the SCRATCH release action, or if the volume is scratch. If a private volume is not pending release, you cannot change the REPLACE action (see instead the RELEASEACTION operand). DFSMSrmm processing sets off the pending SCRATCH action and sets the REPLACE action. DFSMSrmm processing for a scratch volume sets the volume to a master volume pending release with the REPLACE action pending. Any change from scratch to master for a system-managed volume requires access to the TCDB to update the volume status. Otherwise, the CV subcommand fails.

You specify NO to reset the replace action if it is a pending action. For pending release volumes, the SCRATCH action is set on by DFSMSrmm processing. This is not a confirmation that the REPLACE action is completed. Instead, use the CV subcommand with CRLSE(REPLACE) to confirm the REPLACE action. When you specify REPLACE(NO), you are avoiding the REPLACE action.

WMC(count | +nn)

Use this operand to set the write mount count for any volume. The write mount count reflects how many times the volume has been mounted for output and actually written to while mounted.

You can specify an absolute value or an incremental value. Incremental values are specified as follows: WMC(+1). If the value specified causes the count to exceed 65,535, the maximum value of 65,535 is set.

For WORM volumes mounted and processed under DFSMSrmm control on z/OS, the WMC value is obtained from the tape drive and can then no longer be changed via commands. The write mount count cannot be changed for OCE-recorded WORM volumes.

### Testing new subcommand operands

To use the new operands on existing volumes, you must specify FORCE if the volume was previously O/C/EOV recorded, as you can see in Figure 17-26. In this case you need UPDATE access to the resource STGADMIN.EDG.FORCE defined in RACF class FACILITY.

<table>
<thead>
<tr>
<th>EDG3224I UPDATE REQUEST DENIED AS THE VOLUME INFORMATION IS O/C/EOV RECORDED</th>
</tr>
</thead>
</table>

*Figure 17-26  Message EDG3224I that tell you the volume is O/C/EOV recorded*
In Figure 17-27 we show the most important fields of volume THS004 before we have updated them.

```
RMM LISTVOLUME THS004 STAT
Statistics:
Number of data sets = 1       Data set recording= ON
Volume usage(KB) = 951384    Use count = 7
Capacity(MB) = 476837       Percent full = 0
Date last read = 2008/073     Date last written = 2008/073
Drive last used = 0B23       Write mount count = 7
Volume sequence = 1           Media name = 3480
Previous volume =             Next volume =
Product number =              Level = V R M
Feature code =                Error counts:
Temporary read = 0             Temporary write = 0
Permanent read = 0             Permanent write = 0
```

**Figure 17-27  Volume statistics before update**

We have changed the volume CAPACITY, ERROR count, OPENCOUNT and the write mount count (WMC) of volume THS004 using the TSO RMM CHANGEVOLUME subcommand as shown in Figure 17-28. We have specified the FORCE operand too, because this volume is O/C/EOV recorded and the use of ERROR, OPENCOUNT and WMC required the FORCE operand in this case.

```
RMM CHANGEVOLUME THS004 -
    CAPACITY(200) /* SPECIFY THE VOLUMES CAPACITY */-
    ERROR(READ(100),WRITE(200)) /* PERM READ AND WRITE ERRORS */-
    OPENCOUNT(+100) /* ADD 100 TO THE ACTUAL OPEN CNT */-
    WMC(+500) /* ADD 500 TO THE WRITE MOUNT CNT */-
    FORCE /* DEPENT VOLUME IS O/C/EOV REC. */-
```

**Figure 17-28  Changes to a volume record**

In Figure 17-29 you can see the updated fields. Also you can see that DFSMSrmm will not recalculate the percent used information at this time.

```
RMM LISTVOLUME THS004 STAT
Statistics:
Number of data sets = 1       Data set recording= ON
Volume usage(KB) = 951384    Use count = 107
Capacity(MB) = 200           Percent full = 0
Date last read = 2008/073     Date last written = 2008/073
Drive last used = 0B23       Write mount count = 507
Volume sequence = 1           Media name = 3480
Previous volume =             Next volume =
Product number =              Level = V R M
Feature code =                Error counts:
Temporary read = 0             Temporary write = 0
Permanent read = 100           Permanent write = 200
```

**Figure 17-29  Volume statistics after update**
17.6 Report generator enhancements

The DFSMSrmm report generator is an Interactive System Productivity Facility (ISPF) application that you can use to create reports. The report generator:

- Provides reports that you can run as-is or that you can modify as you wish. You can use samples to create reports for volumes, data sets, racks, owners, and the retention and movement policies that are established for your installation. You can modify these samples to create tailored reports. DFSMSrmm ships samples in SYS1.SAMPLIB.
- Generates job control language (JCL) that is based on specifications that you use to submit the report jobs. The generation of JCL depends on the report type and therefore the macros that map the data records. The generation knows, based on the macro name and keyword options used, whether to generate a DCOLLECT jobstep, a DFSMShsm FSR reformat, a DFSMSrmm extract, or a copy of SMF records.
- Includes samples for reporting from DCOLLECT and DFSMShsm data.

The DFSMSrmm Report Generator is updated to support keywords for assembler macros from which report types are derived, and to add new built-in data extract steps. This is in support of new SMF record types from DFSMSrmm, and for DFSMShsm and DCOLLECT reporting.

Any new report type or report definitions created on z/OS V1R10 that include macro keyword information can be used by a lower level release, however, the macro keywords are ignored and are removed if the report type or report definition is changed or updated. The original input report type or report definition is unaffected and can be reused later on a supporting release to exploit the macro keywords.

**Recommendation:** Use z/OS V1R10 or later release to update/customize report types and reports that require macro keywords to be specified. After the report JCL is generated, you can run that JCL on any supported release.

17.6.1 Setting up the report generator for your installation

The following steps are necessary for setting up the DFSMSrmm report generator before you can use it. If you have defined the DFSMSrmm Report Generation settings, you can skip this section and go directly to “Creating a report” on page 238. Otherwise, follow these steps:

1. Customize the EDGRMAIN EXEC. The default name of the library you can find this REXX Exec SYS1.SEDGEXE1. The REXX variable names that you can customize all start with the characters ‘cedggrdl’. a.
   a. Define the installation library name and optionally customize the product library name in EXEC EDGRMAIN. There is no installation library name in the EXEC, so you must add the name.
   b. Update the default naming convention in EXEC EDGRMAIN for the user library name and the JCL library name, if necessary.

   Figure 17-30 shows the section of the EDGRMAIN EXEC that you can customize. For example, to allow a shared library that is used by all users, you must remove the 4 lines that are strikethrough and add the highlighted one. In our example, we use the two libraries:
   - RMM.REPORT.LIB
   - RMM.REPORT.JCL
Figure 17-30  Change prefix in EDGRMAIN REXX exec

/* Initialise Report library names */                           /*§09A*/
address "ISPEXEC" "VGET ZPREFIX"                                /*§09A*/
if length(zprefix) = 0 then                                     /*§10C*/
    edggpref = sysvar('SYSUID')                                   /*§09A*/
else                                                            /*§10C*/
    edggpref = zprefix                                            /*§10C*/

    edggpref = "RMM"                                              /*NSCH*/

cedggrdlu = """"edggpref!!"""".REPORT.LIB""  /* User Library §10C*/
cedggrdlj = """"edggpref!!"""".REPORT.JCL""  /* User JCL Library §10C*/
cedggrdlp = """"SYS1.SAMPLIB""             /* Product Library §10C*/
cedggrdlr = ""                           /* Installation Library §10A*/

2. Starting the RMM ISPF dialog from the ISMF PRIMARY OPTION MENU by entering the
selection "R" to call the Removable Media Manager as shown in Figure 17-31. You can
also select of option 'G' in the ISMF primary menu: Report Generation.

There is a selection available in the ISMF PRIMARY OPTION MENU based on the “User
Mode Selection” that can be changed by setting up your ISMF profile. Use the selection
“0” to change your profile if there is a need.

Figure 17-31  Selecting the Removable Media Manager on the ISMF Primary Option Menu panel
3. From the **REMOVABLE MEDIA MANAGER (DFSMSrmm)**, ISPF primary menu, as shown in Figure 17-32, choose 5 to select the **DFSMSrmm Command Menu**. You can also type REPORT on the option line to go directly to the primary **DFSMSrmm Report Generator ISPF selection panel**.

![Panel Help](image)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>5 COMMANDS</strong></td>
<td>Full DFSMSrmm structured dialog</td>
</tr>
<tr>
<td><strong>X EXIT</strong></td>
<td>Exit DFSMSrmm Dialog</td>
</tr>
</tbody>
</table>

Figure 17-32  Selecting the **COMMANDS** option on the DFSMSrmm Primary Menu panel

4. Select **R** from the **DFSMSrmm Command Menu** as shown in Figure 17-33 to get the primary **DFSMSrmm Report Generator ISPF selection panel**.

![Panel Help](image)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>R REPORT</strong></td>
<td>Report generator</td>
</tr>
</tbody>
</table>

Figure 17-33  Selecting the **REPORT** option on the DFSMSrmm Command Menu panel
5. If you are using the DFSMSrmm Report Generator for the first time, you can set up your user options by selecting "0" for Specify dialog options and defaults as shown in Figure 17-34.

```
Panel  Help

DFSMSrmm Report Generator

Option ===> 0

0 OPTIONS - Specify dialog options and defaults
1 REPORT - Work with reports
2 REPORT TYPE - Work with report types
3 REPORTING TOOL - Work with reporting tools

Enter selected option or END command. For more info., enter HELP or PF1.
```

*Figure 17-34 Selecting the Options option on the DFSMSrmm Report Generator panel*

6. In the DFSMSrmm Dialog Options Menu panel, select option 1 - Specify processing options as shown in Figure 17-35.

```
Panel  Help

DFSMSrmm Dialog Options Menu

Option ===> 1

1 USER - Specify processing options
2 SORT - Specify list sort options
3 REPORT - Specify report options

Enter selected option or END command. For more info., enter HELP or PF1.
```

*Figure 17-35 Selecting the USER option on the DFSMSrmm Dialog Options Menu panel*
7. Figure 17-36 shows the DFSMSrmm Dialog User Options panel to specify the JOBCARDs that should be used for all jobs generated by DFSMSrmm (If you do not provide a JOBCARD, the dialog attempts to use the one that you have defined for use within ISPF.)

8. After you are back to the DFSMSrmm Dialog Options Menu panel, select the option 3 for Specify report options as shown in Figure 17-37.
9. In the *DFSMSrmm Report Options* panel, you have to specify the installation library that you want to use as your user library as show in Figure 17-38. If you do not allocate the library, DFSMSrmm automatically allocates the library by using a primary space and secondary space of 10 tracks and 50 directory blocks.

![Panel Help](image)

**Report Definition Libraries**

These fields enable you to specify or change the names of the libraries containing the report definitions, report types and reporting tools.

The format of the report definition library members is fixed length, 80 byte records. You can use partitioned data sets or libraries. The DFSMSrmm Product Library is normally SYS1.SAMPLIB and contains report definitions and samples shipped with the product.

Your installation defines the Installation library. You can name a User library. If you do not create the User library, DFSMSrmm automatically creates it for you. The information in the Report Definition Libraries is used in an hierarchical order; first the User library, then the Installation library and finally the Product library. When there are duplicate entries the first occurrence in the hierarchy is used and others are ignored. Report type definitions are stored in the EDGGRRTD member of each library and the reporting tool definitions are stored in the EDGGGTOOL member.

Possible values:

- Any data set name
- Default values are library names defined in the EDGRMAIN exec:
  - User . . . . . . . . . . 'MHLRES7.REPORT.LIB'
  - Installation . . . .
  - Product . . . . . . . . 'SYS1.SAMPLIB'

**User Report JCL Library**

This field enables you to specify or change the name of the library containing the report JCL generated by DFSMSrmm. The format of the user report JCL library is fixed length, 80 byte records. You can use partitioned data sets or libraries.

The User Report JCL Library is required for you to generate and submit reports.

*Figure 17-38  Specifying the user library on the DFSMSrmm Dialog Options panel*
You must name a User Report JCL Library. If you do not create the User Report JCL Library, DFSMSrmm automatically creates it for you.

Possible values:
- Any data set name

Default value is the library name defined in the EDGRMAIN exec:
- ‘MHLRES7.REPORT.JCL’

Return to the DFSMSrmm Dialog Options Menu using one of the following options:
- Enter END command to save changes.
- CANCEL to end without saving.

Now you have defined all needed options and you can leave the DFSMSrmm Dialog Options Menu to come back to the DFSMSrmm Report Generator primary option menu.

**Note:** Set up the access lists for the libraries. Provide READ authority to the users of the installation libraries and the product libraries.

10. At the end we will show the current supported tools can by used to create reports. Figure 17-37. shows that today the ICETOOL and SYNCTOOL are the only tools are supported. You can ADD, CHANGE or DELETE a reporting tool at any time.

![Figure 17-39 Selecting the preferred tool on the DFSMSrmm Reporting Tools panel](image)

### 17.6.2 Creating a report

1. Select the Report Definition panel to customize report definitions that are shipped with the product. The report definition is a report file that contains all of the information that is needed to run a report. Each report definition in the product library, installation library, or user library contains the report type information, reporting tool information, the data fields that are used in the report, and the sort order of the records. The report selection criteria specify the subset of records that are used for a report. The reporting tool is a REXX EXEC that builds control statements to create reports that use a reporting utility, such as DFSORT’s ICETOOL. You can change the reporting tool at any time.
Select “1” from the *DFSMSrmm Report Generator* primary option menu as shown in Figure 17-40 to create a new report or work with existing reports.

![Panel Help](image)

**DFSMSrmm Report Generator**

Option ===> 1

0 OPTIONS - Specify dialog options and defaults
1 REPORT - Work with reports
2 REPORT TYPE - Work with report types
3 REPORTING TOOL - Work with reporting tools

Enter selected option or END command. For more info., enter HELP or PF1.

*Figure 17-40  Selecting the REPORT Option on the DFSMSrmm Report Generator menu panel*

2. From the *DFSMSrmm Report Definition Search* panel shown in Figure 17-41, you can specify that all existing reports are listed or only reports generated by a specific user ID. Also you can specify which library should be searched for. If you do not specify any selection DFSMSrmm will search for all available reports in the User and Product libraries when you press Enter.

![Panel Help](image)

**DFSMSrmm Report Definition Search**

Command ===> 

Report name . . May be generic. Leave blank for all reports.

User id . . . Leave blank for all user ids.

Libraries (enter S): Select one or more library.

- S User Default are all defined libraries.
- Installation
- S Product

The following line commands will be available when the list is displayed:

- A - Add a new report definition
- D - Delete a report definition
- G - Generate and save the JCL
- J - Edit and manually submit the JCL
- L - List macro assembly results
- M - Macros for report type are browsed
- N - Copy a report definition
- S - Display or change the report definition
- T - Select a reporting tool

*Figure 17-41  Searching for defined reports*
Where:

**Report name**  Use the Report Name field to search for report definitions by name. The report definition name is the name of a member in one of the report definition libraries (User, Installation or Product). The least generic report name that is used for the search in the Product library is EDGG*.

Possible values:
- 1 to 8 alphanumeric or national characters for a specific report name
- 1 to 7 alphanumeric or national characters, followed by a *, for a generic report name
- * for all report names, which is the default
- The first character must be an alphabetic or national character.

**User ID**  Use the User ID field to specify the creation or last update user id of report definitions.

Possible values:
- 1 to 8 alphanumeric or national characters for a specific user id
- * for all user ids, which is the default
- The first character must be an alphabetic or national character.

**Libraries:**  Select one or more of the User, Installation or Product report definition libraries to specify which libraries are searched for report definitions. DFSMSrmm searches the libraries in the hierarchy shown so that duplicate report definition names in different libraries are skipped. If you add any members to the product library, use member names that start with the EDGG prefix.

The default is to search all defined libraries.

3. You get a list of all available reports shown in the *DFSMSrmm Report Definitions* panel in form of a table as shown in Figure 17-42. There are now 16 DFSMShsm related reports available. To show the details for the DCOLLECT BACKUP DATA report, use the “S” line command in front of this report.
Panel Help

DFSMSrmm Report Definitions
Row 1 to 23 of 40

Command ===> Scroll ===> PAGE

The following line commands are valid: A,D,G,J,L,M,N,S, and T

<table>
<thead>
<tr>
<th>S</th>
<th>Name</th>
<th>Report title</th>
<th>Report type</th>
<th>User id</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ARCGAB01</td>
<td>ABARS ABACKUP Statistics</td>
<td>DFSMShsm ABARS Report</td>
<td>HSM</td>
</tr>
<tr>
<td>A</td>
<td>ARCGAR01</td>
<td>ABARS ARECOVER Statistics</td>
<td>DFSMShsm ABARS Report</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGDB01</td>
<td>DCOLLECT BACKUP DATA</td>
<td>DFSMShsm DCOLLECT BACKUP</td>
<td>HSM</td>
</tr>
<tr>
<td>L</td>
<td>ARCGD001</td>
<td>DCOLLECT DASD CAPACITY PLANNING</td>
<td>DFSMShsm DCOLLECT DASD CAP</td>
<td>HSM</td>
</tr>
<tr>
<td>M</td>
<td>ARCGD001</td>
<td>DCOLLECT MIGRATION DATA</td>
<td>DFSMShsm DCOLLECT MIGRATION</td>
<td>HSM</td>
</tr>
<tr>
<td>T</td>
<td>ARCGD001</td>
<td>DCOLLECT TAPE CAPACITY PLANNING</td>
<td>DFSMShsm DCOLLECT TAPE CAP</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS001</td>
<td>Statistics for DFSMShsm</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS002</td>
<td>Statistics for Backup</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS003</td>
<td>Statistics for Migration</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS004</td>
<td>Statistics for Recall</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS005</td>
<td>Statistics for Recovery</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS006</td>
<td>Statistics for Volume Dump</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS007</td>
<td>Statistics for Restore from Dump</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS008</td>
<td>Statistics for FRRestore</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS009</td>
<td>Statistics for FRRestore</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>S</td>
<td>ARCGS010</td>
<td>DFSMShsm Thrashing Report</td>
<td>DFSMShsm FSR-SMF Records</td>
<td>HSM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGAUD1</td>
<td>SMF Audit of Volumes by Volser</td>
<td>SMF Records for Volumes</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGAUD2</td>
<td>SMF Audit of Volume by Rack</td>
<td>SMF Records for Volumes</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGAUD3</td>
<td>SMF42 Audit of Volumes by Vols</td>
<td>SMF42 Records for Volumes</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGAUD4</td>
<td>SMF42 Audit of Volume by Rack</td>
<td>SMF42 Records for Volumes</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGDCCS</td>
<td>Data Sets by Storage Group</td>
<td>DFSMS DCOLLECT for Data Sets</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGDSNM</td>
<td>Mixed Case data sets Retained</td>
<td>Extended Extract Records</td>
<td>DFRRM1</td>
</tr>
<tr>
<td>D</td>
<td>EDGGREPL</td>
<td>Volumes to be replaced</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGREPV</td>
<td>Volumes to be replaced based on</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR01</td>
<td>Scratch tapes by volume serial</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR02</td>
<td>List of SCRATCH Volumes by Dat</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR03</td>
<td>Inventory List by Volume Serial</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR04</td>
<td>Inventory List by Dataset Name</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR06</td>
<td>Inventory of Volumes by Locati</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR07</td>
<td>Inventory of Dataset by Locati</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR08</td>
<td>Inventory of Bin by Location</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR09</td>
<td>Datasets in Loan Location</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR10</td>
<td>Volumes in Loan Location</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR11</td>
<td>List MultiVolume and MultiFile</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR12</td>
<td>Movement Report by Dataset</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR13</td>
<td>Movement Report by Bin</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR14</td>
<td>Movement Report by Volume Serial</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGR15</td>
<td>Volume Inventory Including Vol</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGSEC1</td>
<td>Report of Accesses to Secure V SMF</td>
<td>SMF Security Records</td>
<td>RMM</td>
</tr>
<tr>
<td>D</td>
<td>EDGGSEC2</td>
<td>Report of Accesses to Se SMF Security</td>
<td>Extended Extract Records</td>
<td>RMM</td>
</tr>
</tbody>
</table>

Figure 17-42 DFSMSrmm Report Definitions panel showing all available default reports
Notes:

- You can also add new report types for data other than data that is created by DFSMSrmm. For example, the report types shipped with the report generator include types for DCOLLECT and DFSMShsm reporting.
- The report type contains information about a specific type of record in an input data set, the Assembler language macro that defines the record format, and basic record selection criteria. For example, the report type, `Extract Records for Data Sets`, in the product library contains information about the data set record in the extract data set, the `EDGRDEXT` mapping macro, and the minimum subset definition of records that are used in the report.
- Report types contain only the base information from which report definitions are created.

4. To modify an existing report, use the “N” Copy a report definition line selection in front of the report you would like to modify. In the pop-up panel, you must define a new name for this report as shown Figure 17-43.

```
Panel Help

DFSMSrmm Report Definitions         Row 1 to 33 of 41
Command ===>                                                  Scroll ===> PAGE

The following line commands are valid: A,D,G,J,L,M,N,S, and T

S Name     Report title                   Report type                  User id
- --------+---------------------------------++-------+
| ARCGAB01 | !                                 | HSM   |
| ARCGAR01 | !                                 | HSM   |
| N ARCGDB01 | ! Enter the report name . . . . MYOWNREP | MHLRES7 |
| ARCGDO01 | !                                 | CAP   |
| ARGDMO01 | !                                 | TION  |
| ARGDTO01 | !                                 | CAP   |
| ARCGS001 | s                                 | HSM   |
| ARCGS002 | Statistics for Backup            | DFSMShsm FSR-SMF Records HSM |
```

Figure 17-43 Specify a new report name

5. You can select one of the default reports or you can create your own report. To demonstrate how you can create your own report, we show an existing report and give you some information what can be changed. We have selected the `MYOWNREP DCOLLECT BACKUP DATA DFSMShsm DCOLLECT BACKUP` report as shown in Figure 17-42 on page 241 to get the detailed information in the panel DFSMSrmm Report Definition - `MYOWNREP` as shown in Figure 17-44 on page 243.
The MYOWNREP report have 10 data columns and is sorted by date first, data set name and time next.

Where:

**S**
Selection Column:

Enter S in this selection column to list all fields with existing selection criteria, and to optionally add the newly selected field to the list to enable criteria to be entered.

Fields for which a selection criterion already exists, are marked with an asterisk in this column.

**CO**
Column Order:

Use this data column to specify the order and the grouping of the columns in your report. The column order is left to right across the report page.

Enter G to specify fields you want to group. You must then specify the sort order for the fields within the group.

Duplicate numbers are automatically resolved and renumbered from the bottom to the top of the list. Grouped field names appear on the top of the list by sort order. The grouped field names are the primary sort key of the record.
Grouped field names are listed in the report header. A page break is forced each time the contents of a grouping field name changes. Grouped field names are typically fields that have the same contents over several report pages. (such as the location and destination name in a movement report).

Possible values: A decimal number between 1 and 99 or G.

**SO**

**Sort Order:**

Use this data column to specify the Sort Order and the direction for the selected field names to change the sequence in which records are presented in the report.

Grouped field names are the primary sort key of the record. The sort order for grouped field names can only be specified within the range of the number of grouped field names. The default sort direction for grouped field names is ascending.

Possible values: Decimal number between 1 and 99, followed by: A - Ascending or D - Descending.

**Field Name**

The Field Name data column displays the name of a field in the mapping macro specified in the report type definition.

**Column header text**

Use the Column Header Text data column to specify the text used as the column header.

The column header text, the column width and the field length are dependent on each other. Changing the column header text sets the column width to the maximum of the column width, the field length and the column header text.

Changing the column width sets the length of the column header text to the column width. If the column width is less than the length of the field definition the output in the report is truncated to the column width. Truncation removes trailing characters for character fields, and leading characters for numeric fields.

Possible values: 1 to 37 characters.

**CW**

**Column Width**

Use the Column Width data column to specify the length of the column in the report.

The column header text, the column width and the field length are dependent on each other. Changing the column header text will set the column width to the maximum of the column width, the field length and the column header text. Changing the column width will set the length of the column header text to the column width. If the column width is less than the length of the field definition the output in the report will be truncated to the column width.

Possible values: Decimal number between 1 and 99.

**Len**

**The Field Length data column displays the length of the field definition.**

If the column width is less than the length of the field definition, the output in the report will be truncated to the column width.
Type

The Field Type data column displays the type of the field name. Use this information when specifying compare values.

Possible values:
- **C** - Character
- **N** - Numeric
- **B** - Bitstring

6. To add a new column to the report, you can specify an **S** in the selection column and the position the column should be written in the report. In our example we have added the **DS size compressed in KB** information to the existing columns and this information should be shown as column 11 at the end of each row in the report. You can see our selection in Figure 17-45.

<table>
<thead>
<tr>
<th>Type</th>
<th>Field name</th>
<th>Column header text</th>
<th>CW</th>
<th>Len</th>
<th>Typ</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>UBBKLNK</td>
<td>BLOCK LENGTH</td>
<td>12</td>
<td>2</td>
<td>C</td>
</tr>
<tr>
<td>B</td>
<td>UBTFLAG2</td>
<td>INFORMATION FLAG #2</td>
<td>19</td>
<td>1</td>
<td>B</td>
</tr>
<tr>
<td>N</td>
<td>UBRRECSP</td>
<td>RECOVER SPACE ESTIMATE(KB)</td>
<td>26</td>
<td>4</td>
<td>N</td>
</tr>
<tr>
<td>N</td>
<td>UB_USER_DATASIZE</td>
<td>DS size if not compressed in KB</td>
<td>31</td>
<td>4</td>
<td>N</td>
</tr>
<tr>
<td>S 11</td>
<td>UB_COMP_DATASIZE</td>
<td>DS size compressed in KB</td>
<td>24</td>
<td>4</td>
<td>N</td>
</tr>
<tr>
<td>C</td>
<td>UBBDSIE</td>
<td>END OF DCUBCDS</td>
<td>14</td>
<td>1</td>
<td>C</td>
</tr>
</tbody>
</table>

**Figure 17-45**  DFSMSrmm Report Definition panel to updating a column

7. Use END to save changes and return to the **DFSMSrmm Report Definition** panel or press Enter to tailoring your selection **DFSMSrmm Report Criteria** panel as shown in Figure 17-46.

If you have used END, skip the next information and go directly to step 15.
Use the S data column to specify the logical order of the criteria, or to request the Details panel.

Possible values:
- **B** Bottom - Move this entry to the bottom.
- **D** Delete - Delete this entry.
- **I** Detail - Add or change information details, for example, substring parameters.
- **N** Next - Move this entry down by one.
- **P** Previous - Move this entry up by one.
- **R** Repeat - Repeat this entry.
- **T** Top - Move this entry to the top.

8. To generate the job control for this report, return to the DFSMSrmm Report Definitions panel and use G in front of the report to get the DFSMSrmm Report Generation panel as shown in Figure 17-47 to change some variables before the file tailoring is running to create and store the job control.
Figure 17-47 DFSMSrmm Report Generation panel

Where:

**Input Data Set**

Specifies the name of the report input file. This field is optional for a report type definition, but required for the generation of the report JCL. If an extract step is included into the reporting JCL, obviously the output file name of this step is also represented by this Input Data Set parameter. In case of an RMM extract step the file must be preallocated.

Possible values: Any sequential file name

Specify GDG data sets as follows: ‘RMM.EXTRACT(0)’

**Date Format**

Specify the Date Format if you use variable dates with &TODAY ... in the selection criteria. For RMM extract data use the date format specified when creating the data with the RMM report extract inventory management step. When you compare dates, remember that a format such as ISO or JULIAN enables easy comparison because of the year, month, day hierarchy.

Possible values:

- **AMERICAN**
  - dates in format MM/DD/YYYY
- **EUROPEAN**
  - dates in format DD/MM/YYYY
- **ISO**
  - dates in format YYYY/MM/DD
- **JULIAN**
  - dates in format YYYY/DDD
- **free form**
  - The free form has a maximum length of 20 bytes and contains DD and MM (alternatively DDD), and YY or YYYY or CYY. These key characters can be surrounded by fill-in characters.
Examples:

- **YYYYDDD**: This form would fit for packed decimal data.
- **CYYDDD**: The C (century) is set to 1 for years after 2000.
- **DD.MM.YY**: Dot as fill-in character.
- **'   YY''MM''DD****: Three leading blanks and ' as fill-in character.

The default value JUlian is specified in the DFSMSrmm Dialog User Option (option 0.1).

**Create Report Data**

If you want to include the step which creates the Report Data, then choose Y. Additional skeleton variables might be needed for the extract JCL. Possible values:

- **N**: The Create Report Data step is not included.
- **Y**: The Create Report Data step is included.

**Skeleton Variables**

Use the Skeleton Variable fields to help with customization of the optional job step which creates the report data. Unless the skeleton variables are already defined in the report definition, the system tries to determine some skeleton variables based on the applied macro name.

For example: if an RMM report extract macro is applied, then:

- **Skeleton Variable_1**: Will contain DATEFORM(x)
- **Skeleton Variable_2**: Will contain a suggestion for the RMM reportextractmessage dataset name.

Possible values: 1 to 50 characters

9. Use END to save your changes and return to the DFSMSrmm Report Definition panel. You get the information where the JCL is stored as shown in Figure 17-48.

**Panel Help**

---

**DFSMSrmm Report Definitions**

---

**DFSMSrmm Report Definitions**

---

**Row 1 to 22 of 40**

---

**Scroll ====> PAGE**

---

The following line commands are valid: A,D,G,J,L,M,N,S, and T

---

<table>
<thead>
<tr>
<th>S Name</th>
<th>Report title</th>
<th>Report type</th>
<th>User id</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARCGAB01</td>
<td>ABARS ABACKUP Statistics</td>
<td>DFSMShsm ABARS Report</td>
<td>HSM</td>
</tr>
<tr>
<td>ARCGAR01</td>
<td>ABARS ARECOVER Statistics</td>
<td>DFSMShsm ABARS Report</td>
<td>HSM</td>
</tr>
<tr>
<td>MYOWNREP</td>
<td>DCOLLECT BACKUP DATA</td>
<td>DFSMShsm DCOLLECT BACKUP</td>
<td>MHLRES7</td>
</tr>
<tr>
<td>ARCGDD01</td>
<td>DCOLLECT DASD CAPACITY PLANNIN</td>
<td>DFSMShsm DCOLLECT DASD CAP</td>
<td>HSM</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EDGGAUD1</td>
<td>SMF Audit of Volumes by Volser</td>
<td>SMF Records for Volumes</td>
<td>RMM</td>
</tr>
<tr>
<td>EDGGAUD2</td>
<td>SMF Audit of Volume by Rack</td>
<td>SMF Records for Volumes</td>
<td>RMM</td>
</tr>
<tr>
<td>EDGGAUD3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EDGGAUD4</td>
<td>! Report JCL MYOWNREP stored on 'MHLRES7.REPORT.JCL' !</td>
<td>RMM</td>
<td></td>
</tr>
<tr>
<td>EDGGDCDS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EDGDSNM</td>
<td>Mixed Case data sets Retained</td>
<td>Extended Extract Records</td>
<td>DFRMM1</td>
</tr>
</tbody>
</table>

---

**Figure 17-48 DFSMSrmm Report Definitions panel with stored message**
Now you can see the stored report as member ARCDB01 in the library MHLRES7.REPORT.JCL as shown in Figure 17-49. This is the library you have defined in Report definition libraries: USER as described in “Setting up the report generator for your installation” on page 232, Step 8.

Now you can see the stored report as member ARCDB01 in the library MHLRES7.REPORT.JCL as shown in Figure 17-49. This is the library you have defined in Report definition libraries: USER as described in “Setting up the report generator for your installation” on page 232, Step 8.

Figure 17-49 User Report definition library

10. Use the “J” Edit and manually submit a report JCL line command in front of the new generated DCOLLECT BACKUP DATA report to edit the JCL member as shown in Example 17-4. You can see that the DS size compressed in KB column is added to this report. Specify SUBMIT and press enter to submitting the job.

Example 17-4 Sample DCOLLECT BACKUP DATA report JCL

```
/*JOBPARM SYSAFF=SC70
 //*/
//**    SKELETON MEMBER EDGSGICE
//**    TAILORED BY THE RMM REPORT GENERATOR
//**************************************************************************
//**  DATE CALCULATION STEP
//**  COMPARE VALUES CONTAINING  ARE CALCULATED BASED ON RUN DATE
//**  COMPARE VALUES CONTAINING  ARE CALCULATED BASED ON RUN DATE
//DATECONV EXEC PGM=IKJEFT01,PARM='%EDGRGDAT'
//SYSPROC DD DISP=SHR,DSN=SYS1.SEDGEXE1
//SYSTSPRT DD SYSOUT=* 
//DATEPATTERN:YYYYDDD
//INCLIN DD *
//OPTION VLSHRT,VLSCMP
//INCLUDE COND=((9,2,CH,EQ,C'B'), 
// AND, 
// (85,4,PD,LE,&TODAY-000D))
//INREC FIELDS=(1,4,
// 85,4,C' ',
// 81,4,C' ',
// 29,44,C' ',
// 213,6,C' ',
```

Panel Help

<table>
<thead>
<tr>
<th>EDIT</th>
<th>MHLRES7.REPORT.JCL</th>
<th>Row 00001 of 00002</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command ==&gt;</td>
<td>SUBMIT</td>
<td>Scroll ==&gt; PAGE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Prompt</th>
<th>Size</th>
<th>Created</th>
<th>Changed</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>MYOWNREP</td>
<td>89</td>
<td>2008/04/02 07:20:08</td>
<td>MHLRES7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EDGGSEC1</td>
<td>89</td>
<td>2008/03/12 21:26:16</td>
<td>MHLRES7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**End**
91,30,C' ',
155,30,C' ',
123,30,C' ',
193,4,C' ',
197,4,C' ',
77,4,C' ',
209,4,C' ')
SORT FIELDS=(5,4,PD,A,
15,44,CH,A,
10,4,CH,A)
//INCLOUT DD DSN=&INCL,UNIT=SYSALLDA,SPACE=(TRK,(1,1)),DISP=(,PASS),
//
DCB=(RECFM=FB,LRECL=80)
//SYSTSIN DD DUMMY
//*********************************************************************
//**
//**
following the new positions in record after inrec
//**
(variable record length needs a rdw field in column 1 to 4)
//**
//*********************************************************************
//* UBDATE
: orig pos: 00081. position after inrec:
5
//* UBTIME
: orig pos: 00077. position after inrec:
10
//* UBDSNAM
: orig pos: 00025. position after inrec:
15
//* UBFRVOL
: orig pos: 00209. position after inrec:
60
//* UBDATCL
: orig pos: 00087. position after inrec:
67
//* UBMGTCL
: orig pos: 00151. position after inrec:
98
//* UBSTGCL
: orig pos: 00119. position after inrec:
129
//* UBALLSP
: orig pos: 00189. position after inrec:
160
//* UBUSESP
: orig pos: 00193. position after inrec:
165
//* UBDSIZE
: orig pos: 00073. position after inrec:
170
//* UB_COMP_DATASIZE
: orig pos: 00205. position after inrec:
175
//**
//WRITE1
EXEC PGM=ICETOOL,REGION=0M
//SYSPRINT DD SYSOUT=*
//TOOLMSG DD SYSOUT=*
//DFSMSG
DD SYSOUT=*
//INDD
DD DSN=MHLRES5.PEDCOL.EAVTST.G0003V00,
//
DISP=SHR
//OUTDD
DD SYSOUT=*
//TEMP
DD UNIT=SYSALLDA,SPACE=(TRK,(5,25))
//TOOLIN
DD *
SORT FROM(INDD) TO(TEMP) USING(INCL)
DISPLAY FROM(TEMP) LIST(OUTDD) TITLE('DCOLLECT BACKUP DATA') PAGE DATE(4MD/) TIME HEADER('LAST BU DATE') ON(5,4,PD,A0) HEADER('LAST BU TIME') ON(10,4,HEX) HEADER('DSN') ON(15,44,CH) HEADER('1st SRC VOL') ON(60,6,CH) HEADER('DC NAME') ON(67,30,CH) HEADER('MC NAME') ON(98,30,CH) HEADER('SC NAME') ON(129,30,CH) HEADER('ORIG ALLOC (KB)') ON(160,4,FI,A0) HEADER('USER DATA (KB)') ON(165,4,FI,A0) HEADER('BACKUP DS (KB)') ON(170,4,FI,A0) HEADER('DS size compressed in KB') ON(175,4,FI,A0) -

250

DFSMS V1.10 and EAV Technical Guide


BLANK -
TOTAL(' ')
//**                             /* &INCL CONTAINS THE SORT §02A*/
//**                             /* INCLUDE STATEMENTS, MODIFIED §02A*/
//**                             /* IN STEP DATECONV §02A*/
//INCLCNTL DD DSN=&INCL,DISP=(OLD,PASS) /*§02C*/

11. Submit and run the job to get the result as shown in Figure 17-50. The column width can be changed in panel DFSMSrmm Report Definition - MYOWNREP as described in step 4 in this section.

<table>
<thead>
<tr>
<th>LAST BU DATE</th>
<th>LAST BU TIME</th>
<th>DSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007047</td>
<td>18054049</td>
<td>SYS1.SMS.ACDS.GW</td>
</tr>
<tr>
<td>2007053</td>
<td>12395692</td>
<td>YYY.MHLRES2.MVOL.DATA</td>
</tr>
<tr>
<td>2007053</td>
<td>12405017</td>
<td>YYY.SRCHFOR.LIST</td>
</tr>
<tr>
<td>2007053</td>
<td>12405453</td>
<td>YYY.SUPERC.LIST</td>
</tr>
<tr>
<td>2007054</td>
<td>18561271</td>
<td>MHLRES2.DCOLLECT.D987</td>
</tr>
<tr>
<td>2007058</td>
<td>15290805</td>
<td>YYY.DCOLLECT.ESDS0</td>
</tr>
<tr>
<td>2007058</td>
<td>15294019</td>
<td>YYY.DCOLLECT.ESDS01</td>
</tr>
<tr>
<td>2007058</td>
<td>15301437</td>
<td>YYY.DCOLLECT.ESDS1</td>
</tr>
<tr>
<td>2007061</td>
<td>13334149</td>
<td>TESTFR.SUPERC.LIST</td>
</tr>
<tr>
<td>2007071</td>
<td>17392521</td>
<td>TESTFR.CNTL.JCL</td>
</tr>
<tr>
<td>2007071</td>
<td>17531956</td>
<td>TESTFR.CNTL.JCL</td>
</tr>
</tbody>
</table>

Figure 17-50   Sample DCOLLECT BACKUP DATA report

12. You can change your settings to reduce the column width and the sequence in which the columns are shown in the report. In the DFSMSrmm Report Definitions panel, use the S line command in front of the report to get the DFSMSrmm Report Definition - MYOWNREP detail panel. In the example in Figure 17-51, we have changed the column width of the LAST BU DATE, LAST BU TIME and DS size compressed in KB. Also we have changed the column sequence so we have moved the DS size compressed in KB from column 11 to column 3 and have added one to all columns between 3 and 10. That means the column 3 is now column 4, the column 4 is now column 5, and so on.
13. Use END to save your changes and return to the DFSMSrmm Report Definitions panel. Select “G” Generate and save the JCL, than “J” Edit and manually submit the JCL as described in step 7 to step 9 in this section and submit the job. In Figure 17-52 you can see the changes. For example, the DS size column is now before the DSN column.

**Note:** The column header text cannot exceed the column width you have specified in Len.
17.7 System-managed library and tape volume partitioning

Library partitioning can happen for both system-managed and non-system-managed libraries and volumes.

In z/OS V1.10 new PARMLIB controls are provided to enable simplified and more powerful partitioning of system-managed libraries and controlling of application use of tape volumes. Now you can use PRTITION and OPENRULE commands in the EDGRMM PARMLIB member to control library partitioning and the use of volumes by applications.

**Attention:** If you use REJECT commands, you have to convert from the use of REJECT commands in order to use the PRTITION and OPENRULE commands.

You can partition a system-managed library including a VTS by performing these tasks:

- Specify the USE operand value on the RMM ADDVOLUME or RMM CHANGEVOLUME subcommands. You can set this value to MVS or VM or both. If you do not specify MVS for a volume, DFSMSrmm prevents the volume from being defined in the volume catalog on this system.

- Through the use of the PRTITION and OPENRULE PARMLIB commands, you can simplify the maintenance of the PARMLIB members as your libraries and volume ranges change. Operands on the OPENRULE and PRTITION commands allow global actions to be set. You can use one or more specific overrides based on volume sets that have different requirements. Typically, you could add a new range of volumes for use by a single partition and only that one system would need to be updated. The OPENRULE and PRTITION commands allow you to define whether they apply to volumes defined to DFSMSrmm or not. You can use operands on the OPENRULE command to automatically ignore volumes, rather than using EXPDT=98000, ACCODE=xCANORES or a customized EDGUX100.

- Define PARMLIB member EDGRMMxx REJECT prefixes. You can use REJECT to prevent a volume not defined to DFSMSrmm from being defined in a system-managed tape library. The REJECT ANYUSE(prefix) operand prevents a volume from being defined in the system-managed tape library on the current system. The REJECT OUTPUT(prefix) operand allows you to define the volume to the system-managed tape library but only use the volume for input processing.

In this section we only describe the use of the new PRTITION and OPENRULE PARMLIB commands.

**Important:** You can use operands on the OPENRULE command to automatically ignore volumes rather than using EXPDT=98000, ACCODE=xCANORES or a customized EDGUX100.

17.7.1 Sequence of processing during open

The processing path taken depends on whether PRTITION or OPENRULE or REJECT PARMLIB commands are used. If REJECT is used but neither PRTITION nor OPENRULE are used, processing is unchanged. If you have specified PRTITION and OPENRULE PARMLIB commands the sequence of processing is:

1. IGNORE processing determines if this request will be ignored or processed by DFSMSrmm.
2. Do partitioning checking using PRTITION.
3. REJECT processing based on OPENRULE.
17.7.2 Reasons to partition an SMS managed library

An IBM Automated Tape Library can be shared by different systems in two ways:

- By logically dividing it into different partitions (partitioning)
- By allowing all attached systems to sequentially access the tape volumes in the library (sharing).

In this section we are explain the concepts to partitioning only.

**Partitioned library with shared CDS and TCDB**

Our first example in Figure 17-53 shows a system-managed tape library partitioned:

- By using CBRUXENT installation exits on each system
- By using EDGUX200 installation exit on each system
- By using different scratch categories for each system in the library manager data base

The EDGHSKP expiration processing is running by system, but all volumes in the PRIVATE category are shared between the two systems.

**Partitioned library with shared TCDB and different CDSs**

In the example in Figure 17-54 we show a system-managed tape library partitioned:

- By using a different DFSMSrmm control data set for each system
- By using REJECT ANYUSE in the EDGRMMnn PARMLIB member
- By using different scratch categories for each system in the library manager data base

The EDGHSKP expiration processing is running by CDS, but all volumes in the PRIVATE category are shared between the two systems. To access these volumes, you have to specify one of the two JCL parameters EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 installation exit is needed.
Partitioned library with unshared CDS and TCDB

In the third example shown in Figure 17-55 you can see a system-managed tape library partitioned:

- By system
- By using REJECT ANYUSE in EDGRMMnn PARMLIB member
- By using different scratch categories for each system in the library manager data base

The EDGHSKP expiration processing is running by system. All volumes are not shared, and in order to share private volumes like this, you have to define a volume in private status in the TCDB first. Then to access these volumes you have to specify one of the two JCL parameters EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 installation exit is needed.

All volumes are not shared, so you have to define a volume in private status in the TCDB first before you can use it.
Figure 17-56 shows how you can add a volume in the TCDB using AMS commands. To access these volumes, you have to specify one of the two JCL parameters EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 installation exit is needed.

```
//STEP0000 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
   CREATE VOLUMEENTRY (NAME(VLZTB82) - 
        LIBRARYNAME(ATV3494J) - 
        LOCATION(LIBRARY) - 
        USEATTRIBUTE(PRIVATE) - 
        STORAGEGROUP(ATV3494J) - 
        MEDIATYPE(MEDIA2)
/*
```

**Figure 17-56  Add and delete a volume entry to the TCDB**

After you have successfully processed the volume you have previously defined in the TCDB, you should delete the entry from the TCDB using the commands shown in Figure 17-57.

```
//STEP0000 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
   DELETE ('VLZTB82') VOLUMEENTRY PURGE
/*
```

**Figure 17-57  Delete a volume entry from the TCDB**

**Partitioned library with shared CDS and unshared TCDB**

In our last example in Figure 17-58, we show a system-managed library partitioned:

- By system
- By using a customized version of CBRUXENT installation exits on each system
- By using different scratch categories for each system in the library manager data base

The EDGHSKP expiration processing is running by system based on the TCDB. All volumes are not shared so you have to define a volume in private status in the TCDB first before you can use it.
17.7.3 Partitioning support

Each system in the RMMplex uses its own PARMLIB options to determine the processing required. For releases prior to z/OS V1R10, the REJECT commands are used for both partitioning and for Open/Close/EOV volume use decisions. On z/OS V1R10 and later releases, the REJECT commands are only used if specified and if neither PRTITION nor OPENRULE commands are defined. When you define REJECT commands and PRTITION or OPENRULE commands, the REJECT commands cause an DFSMSrmm start-up error and you must reply to EDG0239E to continue the start-up or restart DFSMSrmm with a corrected PARMLIB. If there are no REJECT commands, the PRTITION and OPENRULE command defaults are used:

**EDG0239E**  
**REJECT COMMAND FOUND IN PARMLIB AND NO LONGER SUPPORTED**

**Explanation:** This message is issued during initialization when a REJECT command has been found, but one or more PRTITION and OPENRULE commands are also specified. You must use either REJECT commands or OPENRULE and PRTITION commands.

**System action:** DFSMSrmm initialization stops. This message is followed by message EDG0215D on page 42.

**Operator response:** Notify the system programmer. Reply to message EDG0215D as directed. Restart DFSMSrmm when the system programmer has corrected the error.

**System programmer response:** Do not attempt to use both REJECT commands and PRTITION or OPENRULE commands in PARMLIB. If you are implementing PRTITION and OPENRULE commands, you must remove the REJECT commands from PARMLIB. **You can recommend the operator reply “Y” to message EDG0215D to continue and ignore the REJECT commands.**

**Source:** DFSMSrmm  
**Detecting Module:** EDGPARM  
**Routing Code:** 3  
**Descriptor Code:** 3

17.7.4 Using PRTITION and OPENRULE commands

Using the PRTITION commands, you can control partitioning during entry/insert, export/import, eject, and CUA processing for system-managed volumes:

- DFSMSrmm partitioning is based on a global setting that you can change. The default is that all system-managed and non-system-managed volumes are accepted. This can be represented by the command shown in Figure 17-59.

```
PRTITION VOLUME(*) TYPE(ALL) SMT(ACCEPT) NOSMT(ACCEPT LOCATION(SHELF))
```

*Figure 17-59  Partitioning based on global setting*
You can change the global setting so that all volumes are ignored by using the command shown Figure 17-60.

```
PRTITION VOLUME(*) TYPE(ALL) SMT(IGNORE) NOSMT(IGNORE)
```

*Figure 17-60  Partitioning based on global setting ignoring all volumes*

Using a global command results in the example shown in Figure 17-61.

```
PRTITION VOLUME(*) TYPE(NORMM) SMT(IGNORE) NOSMT(IGNORE)
```

*Figure 17-61  Global rejecting all undefined volumes*

This global command will do the following tasks:

- All system-managed volumes undefined in the DFSMSrmm CDS are left in the insert category to be accepted by another system, and non-system-managed volumes are not added automatically to the CDS.

  **Note:** This command is almost equivalent to the REJECT ANYUSE(*) definition.

- The remainder of the default command
  
  [PRTITION VOLUME(*) TYPE(RMM) SMT(ACCEPT) NOSMT(ACCEPT)]
  
  that is not overridden by your global commands in PARMLIB is used to handle TYPE(RMM) volumes.

- With this approach, you must redefine system-managed volumes to DFSMSrmm to enable ownership of volumes during entry/insert processing.

- **No PRTITION commands** - When REJECT commands are used, processing is as for earlier releases unless any OPENRULE statements are defined. In the latter case and when no REJECT commands are defined, the defaults are used for partitioning.

- **Using selective PRTITION statements on top of a global command,** you can be very specific about which volumes are to become owned by the current system/partition.

- **All ACCEPTed volumes that are not defined to DFSMSrmm are added automatically to**
  
  the DFSMSrmm CDS and owned by this system/partition, unless it is eject processing.

  - For system-managed volumes, ‘Owned by this system/partition’ means that the volume is defined to DFSMSrmm, has an entry in the TCDB, and the scratch category used by the Library Manager (LM) is set based on your values in this system’s DEVSUPxx.

  - The ISMF library default entry status is used for added volumes.

- **Pre-defined volumes are only considered when the PRTITION command specifies**
  
  TYPE(RMM) or TYPE(ALL). The volume status for the TCDB entry is set by DFSMSrmm from the volume information.

  **Note:** As a result of this flexibility, you should no longer need to customize the CBRUXENT exit.

**Defining PRTITION commands**

Example 17-5 shows the syntax of the EDGRMMnn PARMLIB member PRTITION command.
Example 17-5  EDGRMMnn PRTITION command

```
Example 17-5  EDGRMMnn PRTITION command

If there are no PRTITION or OPENRULE commands, or there are no REJECT commands, found in EDGRMMxx PARMLIB, default entries are created from the command shown in Figure 17-62 to cover any volumes for which you do not define a PRTITION command.

```

PRTITION VOLUME(*) TYPE(ALL) SMT(ACCEPT) NOSMT(ACCEPT LOCATION(SHELF))

Figure 17-62  Default values of the PRTITION command

Where:

**SMT/NOSMT(action)** Use the SMT/NOSMT operands to select the action you want based on whether the volume is system-managed or not. You can specify a different action for NOSMT and SMT volumes. System-managed volumes are identified as follows:

- Any request made through the EDGLCSUX programming interface that is used from the OAM installation exits: CBRUXCUA, CBRUXENT, CBRUXEJC, and CBRUXVNL.
- During OPEN processing, the tape drive is identified as being in a system-managed library.
- During EXPROC processing, the volume is in the TCDB, or the current location, or the home location is a system-managed library.
- In any other case, the volume is treated as non-system-managed.
**ACCEPT/IGNORE**

Use this operand to identify the action to be taken by DFSMSrmm. The action applies to library entry/insert/import, eject/export, and CUA processing, OPEN processing, and to EXPROC processing. During OPEN processing, after the decision is taken about IGNOREing the volume, the PRTITION checking occurs first, and the action taken before the OPENRULE REJECT/ACCEPT processing occurs. As a result, the action taken by PRTITION can influence the TYPE processing by OPENRULE.

**ACCEPT**

DFSMSrmm processes the volume. During entry/insert import/export, CUA, and OPEN processing, the volume is added to the DFSMSrmm CDS if not yet defined. For NOSMT volumes, the location from the LOCATION operand is used to set the volume current and home locations. During EXPROC processing, the volume is processed for return to scratch.

Note that a volume identified as TYPE(NORMM) with an action of ACCEPT, that is added to the CDS, now becomes a TYPE(RMM) volume for subsequent processing including OPENRULE processing. By default, all volumes are processed during EXPROC, and all volumes are accepted during library entry/insert/import or OPEN processing.

During OPEN processing and during library entry/insert/import processing, any undefined volumes are automatically added to the DFSMSrmm CDS using the library default volume entry status. During OPEN processing, volumes are added to the CDS only if the current OPMODE is either WARN or PROTECT.

**IGNORE**

DFSMSrmm does not process the volume. During entry/insert/import processing, DFSMSrmm requests that OAM ignore the volume and leave it in the 'insert' category. During CUA and eject/export processing, the request for the volume is ignored. During EXPROC processing, the return to scratch processing is skipped. During OPEN processing:

- For NORMM volumes, the volume is not added to the CDS.
- For RMM volumes, processing depends on the OPENRULE entries.

ACCEPT is the default value.

**LOCATION(SHELF | LocdefHome)**

Use this operand for NOSMT volumes that are of TYPE(NORMM) with action ACCEPT. DFSMSrmm's add volume processing uses the defined location value as the volume's home location and current location.

You can specify either SHELF or the name of a LOCDEF defined location that has a specified TYPE(STORAGE,HOME). DFSMSrmm validates this value against the LOCDEF commands.

Although the operand is only used for NORMM volumes, DFSMSrmm does not prevent you from specifying it on a PRTITION command for volumes of TYPE RMM or ALL.

The default value is SHELF.
TYPE(ALL | RMM | NORMM)

Use to identify the type of volumes selected by the PRTITION command. The volser is used to determine whether the volume is defined in the RMM CDS, and, depending on the function, the VOL1 label and HDR1 can be used. Therefore, DFSMSrmm cannot always determine if the volume is an existing one defined to DFSMSrmm. TYPE is determined as follows:

- For system-managed library functions such as import/export, insert/entry, CUA, and eject, only the volser is used to determine whether the volume is defined in the RMM CDS.
- For OPEN processing, the same rules as volume ignore processing are used and the TYPE that is assigned is based on the following rules:
  - TYPE(RMM)
    1. The volume serial number is defined in the control data set, and there is no HDR1 tape label for the volume.
    2. The volume serial number is defined in the control data set, and no labels are used (NL, NSL, or BLP with an NL tape).
    3. The 17 characters read from the HDR1 label of the mounted volume match the last 17 characters of the data set name in the control data set.
  - TYPE(NORMM)
    1. The volume serial number is not defined in the control data set.
    2. The volume is defined, but the 17 characters read from the HDR1 label of the mounted volume do not match the last 17 characters of the data set name in the control data set.

- For EXPROC processing, TYPE(RMM) always applies because EXPROC is driven only by volumes defined to DFSMSrmm.

For each set of volumes, you can code one command with RMM and another with NORMM, but only a single command if ALL is used.

RMM
The volume must already be defined to DFSMSrmm.

NORMM
The volume is not defined to DFSMSrmm.

ALL
Applies to all volumes regardless of whether they are defined to DFSMSrmm. ALL is the default value.

VOLUME/VOLUMERANGE

Use these operands to select volumes that are to be managed by this command. You must specify either VOLUME or VOLUMERANGE, and each command defines a set of one or more volumes. Sets cannot overlap within a TYPE, but a set can be a subset of another and such a subset is more specific. When you code a set with TYPE(RMM) and repeat that set with TYPE(NORMM), you can specify different operands for each set. If you want to use the same operands for the sets, you can do so by coding a type of ALL.
You can specify the volume as fully qualified or a volser prefix ending in ". A fully qualified volume is one to six alphanumeric, national, or special characters. A volser prefix is zero to five alphanumeric, national, or special characters ending in an asterisk. Quotes are required for special characters, and the first character must not be blank. Any value ending in ", even if enclosed in quotes, is considered to be a volser prefix.

Use to select a subset of volumes based on starting and ending volser. One to six characters for beginning and end of range are required to be processed, and they can be alphanumeric, national, and special characters. Quotes are required for each value regardless of the use of special characters, and the first character must not be blank. The end of range must not be lower than the start of the range.

We take the PRTITION definitions as shown in Figure 17-63 in our EDGRMMnn PARMLIB member.

<table>
<thead>
<tr>
<th>PRTITION VOLUME(*)</th>
<th>/* added 2008/03/20 NS */-</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE(NORMM)</td>
<td>/* undefined volumes in CDS only */-</td>
</tr>
<tr>
<td>SMT(IGNORE)</td>
<td>/* ignore sms managed tapes */-</td>
</tr>
<tr>
<td>NO SMT(IGNORE)</td>
<td>/* ignore no-nsms managed tapes */</td>
</tr>
<tr>
<td>PRTITION VOLUMERANGE('THM000':'THM029')</td>
<td>/* added 2008/03/20 NS */-</td>
</tr>
<tr>
<td>TYPE(ALL)</td>
<td>/* defined and undefined volumes */-</td>
</tr>
<tr>
<td>SMT(IGNORE)</td>
<td>/* accept sms managed tapes */-</td>
</tr>
<tr>
<td>NO SMT(IGNORE)</td>
<td>/* ignore non-sms managed tapes */</td>
</tr>
<tr>
<td>PRTITION VOLUME(DB*)</td>
<td>/* added 2008/03/20 NS */-</td>
</tr>
<tr>
<td>TYPE(ALL)</td>
<td>/* defined and undefined volumes */-</td>
</tr>
<tr>
<td>SMT(IGNORE)</td>
<td>/* ignore sms managed tapes */-</td>
</tr>
<tr>
<td>NO SMT(ACCEPT)</td>
<td>/* accept non-sms managed tapes */</td>
</tr>
<tr>
<td>PRTITION VOLUME(HE00*)</td>
<td>/* added 2008/03/20 NS */-</td>
</tr>
<tr>
<td>TYPE(ALL)</td>
<td>/* defined and undefined volumes */-</td>
</tr>
<tr>
<td>SMT(IGNORE)</td>
<td>/* ignore sms managed tapes */-</td>
</tr>
<tr>
<td>NO SMT(ACCEPT)</td>
<td>/* accept non-sms managed tapes */</td>
</tr>
<tr>
<td>PRTITION VOLUME(J5T*)</td>
<td>/* added 2008/03/20 NS */-</td>
</tr>
<tr>
<td>TYPE(ALL)</td>
<td>/* defined and undefined volumes */-</td>
</tr>
<tr>
<td>SMT(IGNORE)</td>
<td>/* ignore sms managed tapes */-</td>
</tr>
<tr>
<td>NO SMT(ACCEPT)</td>
<td>/* accept non-sms managed tapes */</td>
</tr>
<tr>
<td>PRTITION VOLUME(MXX*)</td>
<td>/* added 2008/03/20 NS */-</td>
</tr>
<tr>
<td>TYPE(ALL)</td>
<td>/* defined and undefined volumes */-</td>
</tr>
<tr>
<td>SMT(IGNORE)</td>
<td>/* ignore sms managed tapes */-</td>
</tr>
<tr>
<td>NO SMT(ACCEPT)</td>
<td>/* accept non-sms managed tapes */</td>
</tr>
<tr>
<td>PRTITION VOLUME(S0024*)</td>
<td>/* added 2008/03/20 NS */-</td>
</tr>
<tr>
<td>TYPE(ALL)</td>
<td>/* defined and undefined volumes */-</td>
</tr>
<tr>
<td>SMT(IGNORE)</td>
<td>/* ignore sms managed tapes */-</td>
</tr>
<tr>
<td>NO SMT(ACCEPT)</td>
<td>/* accept non-sms managed tapes */</td>
</tr>
<tr>
<td>PRTITION VOLUME(S020*)</td>
<td>/* added 2008/03/20 NS */-</td>
</tr>
<tr>
<td>TYPE(ALL)</td>
<td>/* defined and undefined volumes */-</td>
</tr>
<tr>
<td>SMT(IGNORE)</td>
<td>/* ignore sms managed tapes */-</td>
</tr>
<tr>
<td>NO SMT(ACCEPT)</td>
<td>/* accept non-sms managed tapes */</td>
</tr>
</tbody>
</table>

Figure 17-63  EDGRMMnn PRTITION definitions
To list the current PRTITION settings you can use the RMM TSO subcommand LISTCONTROL with the new option PRTITION as shown in Figure 17-64.

```
RMM LISTCONTROL PRTITION
or
RMM LC P
```

*Figure 17-64  LISTCONTROL with option PRTITION*

Figure 17-65 shows the result of the RMM TSO subcommand LISTCONTROL. We used the option PRTITION to get only this information. Each definition is listed twice, one time to show the type RMM settings and the second time for the NORMM settings.

```
Partition Entries:

<table>
<thead>
<tr>
<th>Volume or Range Type</th>
<th>SMT</th>
<th>NOSMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>Action Location</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>RMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>IGNORE ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
<tr>
<td>NORMM</td>
<td>ACCEPT ACCEPT SHELF</td>
<td></td>
</tr>
</tbody>
</table>
```

*Figure 17-65  Result of the LISTCONTROL PRTITION*

Be sure that you have not specified any definition twice, because in this case the DFSMSrmm initialization process is stopped and message EDG0238E following by message EDG0215D is written as shown in Figure 17-66.

```
EDG0204I  DFSMSrmm BEING INITIALIZED FROM MEMBER EDGRMM71 IN RMM.PARMLIB
EDG0238E  OVERLAPPING VOLUME SET DEFINED FOR PRTITION TYPE(ALL). THM* OVERLAPS THM*
EDG0215D  ERRORS DETECTED IN INITIALIZATION PARAMETERS - ENTER 'Y' TO CONTINUE OR 'N' TO CANCEL
```

*Figure 17-66  Error message until DFSMSrmm initialization*

Where:

EDG0238E OVERLAPPING VOLUME SET DEFINED FOR command TYPE(type). value1 OVERLAPS value2

**Explanation:** A PRTITION or OPENERULE command has specified volume selection operands that conflict with another command.
In the message text:

**command** One of these values: PRTITION, or OPENRULE.

**type** One of these values: ALL, RMM or NORMM.

**value1** One of these values: A volume serial number, or a volume prefix specified through the VOLUME operand, or a volume range specified with the VOLUMERANGE operand.

**value2** One of these values: A volume serial number, or a volume prefix specified through the VOLUME operand, or a volume range specified with the VOLUMERANGE operand.

**System action:** DFSMSrmm initialization stops. This message is followed by message EDG0215D on page 42.

**Operator response:** Notify the system programmer. Reply to message EDG0215D on page 42 as directed. Restart DFSMSrmm when the system programmer has corrected the error.

**System programmer response:**
Correct the VOLUME and VOLUMERANGE operands so that overlaps do not exist. Volume sets must be unique within command and type; for example, PRTITION TYPE(RMM) with VOLUME(A*) and PRTITION TYPE(RMM) with VOLUMERANGE(A:B99999) is an overlap because the volume range A:B99999 begins within the A* set but extends beyond it. When the message includes TYPE(ALL) the overlap could be with another command specifying a type of ALL or it could be with RMM or NORMM. DFSMSrmm ignores the overlapping values if initialization continues.

**Source:** DFSMSrmm

**Detecting Module:** EDGSUSE

**Routing Code:** 3

**Descriptor Code:** 3

**Defining OPENRULE commands**
Example 17-6 shows the syntax of the EDGRMnn PARMLIB member OPENRULE command.

**Example 17-6  EDGRMnn OPENRULE command**

```
| OPENRULE -- VOLUME -- (Volser_or_prefix) -- |
| VOLUMERANGE -- (Volser_or_prefix) -- |

| TYPE(ALL) | ANYUSE(ACCEPT) |

| Selection | Intent |
```

**Selection:**
If there are no OPENRULE or PARTITION commands found in EDGRMMxx PARMLIB, or there are no REJECT commands, default entries are created from the command shown in Figure 17-67 to cover any volumes for which you do not define an OPENRULE command.

```
OPENRULE VOLUME(*) TYPE(ALL) ANYUSE(ACCEPT)
```

*Figure 17-67  Default values of the OPENRULE command*

Where:

**TYPE(ALL | RMM | NORMM)**

Use to identify the type of volumes selected by the OPENRULE command. The volser is used to determine whether the volume is defined in the RMM CDS, and, depending on the function, the VOL1 label and HDR1 can be used. Therefore, DFSMSrmm cannot always determine if the volume is an existing one defined to DFSMSrmm.
TYPE is determined as follows:

- For system-managed library functions such as import/export, insert/entry, CUA, and eject, only the volser is used to determine whether the volume is defined in the RMM CDS.

- For OPEN processing, the same rules as volume ignore processing are used, and the TYPE that is assigned is based on the following rules:
  - TYPE(RMM)
    1. The volume serial number is defined in the control data set, and there is no HDR1 tape label for the volume.
    2. The volume serial number is defined in the control data set, and no labels are used (NL, NSL, or BLP with an NL tape).
    3. The 17 characters read from the HDR1 label of the mounted volume match the last 17 characters of the data set name in the control data set.
  - TYPE(NORMM)
    1. The volume serial number is not defined in the control data set.
    2. The volume is defined, but the 17 characters read from the HDR1 label of the mounted volume do not match the last 17 characters of the data set name in the control data set.

- For EXPROC processing, TYPE(RMM) always applies because EXPROC is driven only by volumes defined to DFSMSrmm.

For each set of volumes, you can code one command with RMM and another with NORMM, but only a single command if ALL is used.

### RMM
The volume must already be defined to DFSMSrmm.

### NORMM
The volume is not defined to DFSMSrmm.

### ALL
Applies to all volumes regardless of whether they are defined to DFSMSrmm. ALL is the default value.

#### VOLUME/VOLUMERANGE

Use these operands to select volumes that are to be managed by this command. You must specify either VOLUME or VOLUMERANGE, and each command defines a set of one or more volumes. Sets cannot overlap within a TYPE, but a set can be a subset of another and such a subset is more specific. When you code a set with TYPE(RMM) and repeat that set with TYPE(NORMM), you can specify different operands for each set. If you want to use the same operands for the sets, you can do so by coding a type of ALL.

**VOLUME**
You can specify the volume as fully qualified or a volser prefix ending in ". A fully qualified volume is one to six alphanumeric, national, or special characters. A volser prefix is zero to five alphanumeric, national, or special characters ending in an asterisk. Quotes are required for special characters, and the first character must not be blank. Any value ending in ", even if enclosed in quotes, is considered to be a volser prefix.
VOLUMERANGE Use to select a subset of volumes based on starting and ending volser. One to six characters for beginning and end of range are required to be processed, and they can be alphanumeric, national, and special characters. Quotes are required for each value regardless of the use of special characters, and the first character must not be blank. The end of range must not be lower than the start of the range.

ANYUSE/INPUT/OUTPUT(action)
Use this operand to identify the type of OPEN request issued by the application. It reflects the applications intent towards the data.

INPUT Means that the application is attempting to read from the tape data set.

OUTPUT Means that the application is attempting to write to the tape data set.

ANYUSE Means that the application is attempting either to read from or write to the tape volume.

ACCEPT/IGNORE/REJECT(by)
Use this operand to identify the action to be taken by DFSMSrmm. The action applies to OPEN processing.

ACCEPT Means that DFSMSrmm processes the volume. This attempt to open a file on the tape volume is also subject to DFSMSrmm open-time volume validation and if allowed, the use of the volume and file is recorded by DFSMSrmm. ACCEPT means that the volume is accepted provided that the validation performed by DFSMSrmm at OPEN time allows the volume to be used. ACCEPT is the default value.

IGNORE Means that if requestor is authorized DFSMSrmm does not process the volume. When the mounted volume matches the requested volume, the use of the volume is ignored by DFSMSrmm; no validation of the volume is performed and there is no recording of the volume or file. The user must be authorized to ignore the volume. DFSMSrmm processing is as if the EDGUX100 exit requested the volume is ignored. Volume ignore processing is attempted if either IGNORE is specified or the EDGUX100 exit requests ignore. For ignore processing to be successful, the user must be authorized to ignore the volume. The ignore processing is just as if you coded EXPDT=98000 or ACCODE=xCANORES in the JCL and were using the sample EDGUX100 exit shipped with DFSMSrmm.

The use of action IGNORE enables you to ignore any specific or non-specific volume requests including all those for system-managed volumes.

REJECT Means that DFSMSrmm must prevent OPEN processing from allowing use of the volume. For a specific volume this results in the OPEN request failing. For a non-specific volume, the volume is dismounted by the system and another mount request issued.
BY(SPECIFIC/NONSPECIFIC/ANY)

Use this operand to identify for which requests the IGNORE action applies.

SPECIFIC

Means that the IGNORE action only applies, if a specific volume is requested. Otherwise, the ACCEPT action is used. NONSPECIFIC means that the IGNORE action only applies if a non-specific request is being processed; the volser in the mount message is either PRIVAT or SCRTCH. Otherwise the ACCEPT action is used.

ANY

Means that the IGNORE action applies to all types of request. ANY is the default value.

BY(SYSID, CATLG)

Use this operand for volumes of TYPE RMM to specify that the REJECT action only applies, if the volume is defined to DFSMSrmm, the request is for a specific volume, and the BY condition is not met. Otherwise, the ACCEPT action is used.

When specified for TYPE(NORMM) the operand is parsed successfully but is then ignored and never used by DFSMSrmm because volumes of TYPE(NORMM) cannot have existing tape data set records and are not defined to DFSMSrmm.

You can specify one or more of the values SYSID and CATLG. Use this operand to specify that the REJECT action applies if either applied by SYSID or by CATLG. Otherwise, the ACCEPT action is used.

There is no default value.

SYSID

Use this operand to specify that for non-scratch volumes existing tape data sets, the use of a volume is to be rejected if the creating SYSID of the first file does not match the current SYSID. This operand might help you if you have a common scratch pool across multiple systems, but after a volume is used, you only want the volume used/referenced on the system from which the data was created. When the creating SYSID matches the current SYSID, the ACCEPT action is used.

CATLG

Use this operand to specify that existing tape data sets must be referenced by their catalog entry. When the data set is referenced via its catalog entry the ACCEPT action is used.

DFSMSrmm checks to see if there is an existing data set record defined to RMM and applies this rule if the data set record exists regardless of the disposition specified for the data set. In cases where only the first file is being recorded by DFSMSrmm a data set record does not exist for file 2 onwards, but DFSMSrmm processing assumes that they exist.

Figure 17-68 shows the OPENRULE definitions we have inserted in our existing EDGRMMnn PARMLIB member. These rules will allow the normal use of all volumes defined to DFSMSrmm “TYPE(RMM)” and all other volumes not defined in the DFSMSrmm control data set can be used if you have the needed access to the STGADMIN.EDG.IGNORE.TAPE.* resource defined in RACF class FACILITY.
Chapter 17. DFSMSrmm V1.10 enhancements

17.7.5 Testing various OPENRULE settings

First of all, we have created a new data set on tape using the JCL as shown in Figure 17-70. In the next test cases we have used this data set for input but using different OPENRULE settings. The data set was created on volume THS013, an IBM 3592 volume inside an IBM automated tape library (ATL).

Figure 17-70  Sample JCL used to create a single data set on tape

Our conclusion of the test is, that in an unshared tape environment, it means only a single DFSMSrmm control data set, all tapes are defined in this CDS and all tape drives are shared, that only the both OPENRULEs shown in Figure 17-68 and both the RACF definitions shown in Figure 17-71 are required.
Test case 1

Function: Read an existing data set on an SMS managed tape volume using IEBGENER. The used volume serial number is THS013. That is normal tape processing. Figure 17-72 shows the JCL we used.

Result: The job ended without any errors, because the user MHLRES7 have access to the data set and the volume can based on the OPENRULE be used on this system.

OPENRULE: All DFSMSrmm defined and not-defined volumes will be accepted on this system.

RACF: The user has access to the data set profile protecting this data set. It is not necessary that the user have access to the special resources defined in RACF class FACILITY:

- STGADMIN.EDG.IGNORE.TAPE.RMM.*
- STGADMIN.EDG.IGNORE.TAPE.NORM.*

Figure 17-72 Test case 1 sample JCL

//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*   
//SYSTSIN DD *
RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.RMM.* -
   UACC(NONE) OWNER(SYS1 )
RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.NORM.* -
   UACC(NONE) OWNER(SYS1 )

//*
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*   
//SYSTSIN DD *
RMM TSO LISTCONTROL
//*
//TESTCASE EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*   
//SYSUT1 DD DISP=SHR,DSN=MHLRES7.TEST.OPENRULE.RULES,
//   UNIT=ATL3, VOL=SER=THS013
//SYSUT2 DD SYSOUT=*   
//SYSIN DD DUMMY

Where:

- **RACFCMDS** Shows the STGADMIN.EDG.* resources defined in RACF class FACILITY.
- **LISTCONT** Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command.
- **TESTCASE** Uses the IEBGENER utility to read the data set.
Figure 17-73 shows the result of the RACF SEARCH command showing that the resources STGADMIN.EDG.IGNORE.TAPE.RMM.* and STGADMIN.EDG.IGNORE.TAPE.RMM.* are not defined at this time.

```
READ
  SR   CLASS(FACILITY) MASK(STGADMIN.EDG)
STGADMIN.EDG.LISTCONTROL
STGADMIN.EDG.MASTER
READY
```

**Figure 17-73   Result of test case 1 RACF listing**

The result of the RMM TSO LISTCONTROL subcommand is shown in Figure 17-74. You can see that there is only one OPENRULE definition for each of the RMM and NORMM managed volumes, and that for both definitions, all tapes are accepted.

```
Figure 17-74   Result of test case 1 LISTCONTROL

Figure 17-75 we show the JES message log of this job. You can see that there were no errors.

```
| IEF403I OPENR002 - STARTED - TIME=14.18.28 - ASID=0042 - SC70 |
|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|
| JOBNAME              | STEPNAMt            | PROCSTEP             | RC | EXCP | CPU | SRB | CLOCK | SERV | PG |
| OPENR002             | RACFCMDS            | 00 | 28 | .00 | .00 | .00 | 340 | 0   |
| OPENR002             | LISTCONT            | 00 | 32 | .00 | .00 | .00 | 347 | 0   |

```

**Figure 17-75   JES message log of test case 1**

**Test case 2**

**Function:** Read an existing data set on an SMS managed tape volume using IEBGENER. The used volume serial number is THS013. That is normal tape processing. Figure 17-76 shows the JCL we used.

**Result:** The job abended. Although the user MHLRES7 has access to the data set residing on this volume, the volume cannot be used on this system because of the OPENRULE definition.
OPENRULE: All DFSMSrmm defined and not-defined volumes are ignored on this system.

RACF: The user has access to the data set profile protecting this data set. There is an additional need to have READ access to the special resource:
STGADMIN.EDG.IGNORE.TAPE.RMM.*
This is defined in RACF class facility, because the volume is ignored from DFSMSrmm.

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*    
//SYSTSIN DD   *   
SR CLASS(FACILITY) MASK(STGADMIN.EDG)
//*
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*    
//SYSTSIN DD   *   
RMM TSO LISTCONTROL
//*
//TESTCASE EXEC PGM=IEBGENER
//SYSPRINT DD   SYSOUT=*    
//SYSUT1 DD   DISP=SHR,DSN=MHLRES7.TEST.OPENRULE.RULES,   
//UNIT=ATL3, VOL=SER=THS013
//SYSUT2 DD   SYSOUT=*    
//SYSIN DD   DUMMY
```

**Figure 17-76 Test case 2 sample JCL**

Where:

RACFCMDS Shows the STGADMIN.EDG.* resources defined in RACF class FACILITY.

LISTCONT Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command.

TESTCASE Uses the IEBGENER utility to read the data set.

Figure 17-77 shows the result of the RACF SEARCH command to see that the resources STGADMIN.EDG.IGNORE.TAPE.RMM.* and STGADMIN.EDG.IGNORE.TAPE.RMM.* are not defined at this time.

```
READY
SR CLASS(FACILITY) MASK(STGADMIN.EDG)
STGADMIN.EDG.FORCE
STGADMIN.EDG.LISTCONTROL
STGADMIN.EDG.MASTER
READY
```

**Figure 17-77 Result of test case 2 RACF listing**
The result of the RMM TSO LISTCONTROL subcommand is shown in Figure 17-78. You can see that there is only one OPENRULE definition for RMM and NORMM managed volumes. This means that DFSMSrmm must depend on this definition to ignore all tape requests on this system.

```
<table>
<thead>
<tr>
<th>Volume or Range Type</th>
<th>Action Condition</th>
<th>Action Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>RMM</td>
<td>IGNORE ANY</td>
</tr>
<tr>
<td>*</td>
<td>NORMM</td>
<td>IGNORE ANY</td>
</tr>
</tbody>
</table>
```

Figure 17-78  Result of test case 2 LISTCONTROL

In Figure 17-79, we show the JES message log of this job. You can see that the job gets a security violation and an error opening the tape volume.

```
IEF403I OPENR003 - STARTED - TIME=14.39.26 - ASID=0042 - SC70
     --TIMINGS (MINS.)--
     -JOBNAME  STEPNAME PROCSTEP   RC  EXCP   CPU  SRB   CLOCK  SERV  PG
     -OPENR003  RACFCMDS          00  28  .00  .00  .00  1213  0
     -OPENR003  LISTCONT          00  32  .00  .00  .00   312   0

IEF233A M 0B23,THS013,,OPENR003,TESTCASE,MHLRES7.TEST.OPENRULE.RULES
ICH408I USER(MHLRES7 ) GROUP(SYS1    ) NAME(MARY LOVELACE - RESI)  306
STGADMIN.EDG.IGNORE.TAPE.RMM.THOS13 CL(FACILITY)
INSUFFICIENT ACCESS AUTHORITY
FROM STGADMIN.** (G)
ACCESS INTENT(READ   ) ACCESS ALLOWED(NONE   )
EDG4060I VOLUME THS013 REJECTED. OPENRULE ACTION IGNORE BUT USE OUTSIDE
EDG4060E VOLUME THS013 ON 0B23 REJECTED FOR USE BY OPENR003, TESTCASE, S
IEC518I DUMP SUPPRESSED - USER NOT AUTHORIZED BY SAF
IEF450I OPENR003 TESTCASE - ABEND=S413 U0000 REASON=00000008  313
     TIME=14.39.44
     -OPENR003  TESTCASE *S413  40  .00  .00  .30   526   0
IEF404I OPENR003 - ENDED - TIME=14.39.44 - ASID=0042 - SC70
```

Figure 17-79  JES message log of test case 2

Where:

**ICH408I**  The STGADMIN.** resource is used because there are no more specific RACF resource defined in RACF class FACILITY.
DFSMSrmm uses the normal RACF best matching processing to find a matching profile or resource. The user has currently no access to this definition.

**IEC145I**  The 413-08 is a result of the reject of the tape volume because of the OPENRULE definition.
EDG4060I  VOLUME volser REJECTED. OPENRULE ACTION IGNORE BUT USE OUTSIDE OF DFSMSrmm CONTROL NOT AUTHORIZED

Explanation: The volume matched to an OPENRULE entry that specified an action of IGNORE. The IGNORE action requests that DFSMSrmm ignore this volume. The user must be authorized to request that the specified volume be ignored. The user was not authorized and the request failed. Message EDG4060I can be accompanied by an ICH408I message that explains the reason for the authorization failure. If the ICH408I message is not issued, the most likely cause for the error is that there is no security profile defined for ignore processing. When there is no profile, DFSMSrmm does not allow a volume to be ignored. You should add one or more security profiles to ensure the correct authorization.

In the message text:

- volser: The volume serial number of the volume that the user is attempting to use.

System action: If DFSMSrmm is operating in warning mode, the volume specified in this message can be used and DFSMSrmm issues message EDG4004I. If DFSMSrmm is operating in protect mode, DFSMSrmm issues message EDG4005E or EDG4006E.

Operator response: None.

System programmer response:
If the volume should be ignored, define the RACF resource:
- STGADMIN.EDG.IGNORE.TAPE.volser,
- STGADMIN.EDG.IGNORE.TAPE.NORMALM.volser, or
- STGADMIN.EDG.IGNORE.TAPE.RMM.volser
Then authorize the user. Refer to the z/OS DFSMSrmm Implementation and Customization Guide for information about authorizing users.

Source: DFSMSrmm
Detecting Module: EDGOECM
Routing Code: 2,3
Descriptor Code: 3

EDG4006E  VOLUME volser ON rack_number REJECTED FOR USE BY jobname, stepname, ddname; OPEN REQUEST FAILED BY DFSMSrmm

Explanation: Neither the current volume nor any other tape volume can be used for this mount request.

In the message text:

- volser: Volume serial number
- rack_number: Volume shelf location identifier
- job_name: Name of a job identified to a system.
- stepname: Name of a step within a job
- ddname: Data definition name

System action: The tape is rejected, and the job abnormally ends.

Operator response: None.
System programmer response:

Check the specified DD statement for incorrect volume, density, or label parameters. If the DD statement appears correct, review the DFSMSrmm PARMLIB options to determine if this was a valid occurrence.

Source: DFSMSrmm

Detecting Module: EDGSOCE

Routing Code: 2,3,11

Descriptor Code: 11

Test case 3

Function: Read an existing data set on an SMS managed tape volume using IEBGENER. The used volume serial number is THS013. That is normal tape processing. Figure 17-80 shows the JCL we used.

Result: The job ended without an error, although the volume must be ignored by the OPENRULE setting. In this case, user MHLRES7 has READ access to the resource STGADMIN.EDG.IGNORE.TAPE.RMM.* that we now have defined in the RACF class facility, and has access to the data set residing on the volume.

OPENRULE: All DFSMSrmm defined and not-defined volumes are ignored on this system.

RACF: The user has access to the data set profile protecting this data set and have in the minimum READ access to the special resource:

STGADMIN.EDG.IGNORE.TAPE.RMM.*

This is defined in the RACF class facility, because the volume must be ignored by DFSMSrmm depending on the OPENRULE TYPE(RMM) definition for DFSMSrmm managed volumes.

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=* 
//SYSTIN DD *  
//SR CLASS(FACILITY) MASK(STGADMIN.EDG) 
//*  
//LISTCONT EXEC PGM=IKJEFT01 
//SYSTSPRT DD SYSOUT=* 
//SYSTIN DD *  
//RMM TSO LISTCONTROL 
//*  
//TESTCASE EXEC PGM=IEBGENER 
//SYSPRINT DD SYSOUT=* 
//SYSUT1 DD DISP=SHR,DSN=MHLRES7.TEST.OPENRULE.RULES, 
//UNIT=ATL3, VOL=SER=THS013 
//SYSUT2 DD SYSOUT=* 
//SYSSIN DD DUMMY
```

Figure 17-80  Test case 3 sample JCL
Where:

**RACFCMDS** Shows the STGADMIN.EDG.* resources defined in RACF class FACILITY.

**LISTCONT** Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command.

**TESTCASE** Uses the IEBGENER utility to read the data set.

Figure 17-81 shows the result of the RACF SEARCH command to see that the resource STGADMIN.EDG.IGNORE.TAPE.RMM.* now is defined on this system.

<table>
<thead>
<tr>
<th>RACFCMDS</th>
<th>Shows the STGADMIN.EDG.* resources defined in RACF class FACILITY.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISTCONT</td>
<td>Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command.</td>
</tr>
<tr>
<td>TESTCASE</td>
<td>Uses the IEBGENER utility to read the data set.</td>
</tr>
</tbody>
</table>

**Figure 17-81 Result of test case 3 RACF listing**

The result of the RMM TSO LISTCONTROL subcommand is shown in Figure 17-82. You can see that there is only one OPENRULE definition for RMM and NORMM managed volumes, and that for both definitions, all tapes must be ignored.

<table>
<thead>
<tr>
<th>READY$</th>
</tr>
</thead>
<tbody>
<tr>
<td>RDEFINE FACILITY STGADMIN.EDG.IGNORE.TAPE.RMM.* -</td>
</tr>
<tr>
<td>UACC(NONE) OWNER(SYS1 )</td>
</tr>
<tr>
<td>PERMIT STGADMIN.EDG.IGNORE.TAPE.RMM.* -</td>
</tr>
<tr>
<td>ID(MHLRES7) ACC(UPDATE) CLASS(FACILITY)</td>
</tr>
<tr>
<td>SETROPTS GENERIC(FACILITY) REFRESH</td>
</tr>
<tr>
<td>SETROPTS RACLIST(FACILITY) REFRESH</td>
</tr>
<tr>
<td>SR CLASS(FACILITY) MASK(STADMIN)</td>
</tr>
<tr>
<td>STADMIN.EDG.LISTCONTROL</td>
</tr>
<tr>
<td>STADMIN.EDG.MASTER</td>
</tr>
<tr>
<td>STADMIN.EDG.IGNORE.TAPE.RMM.* (G)</td>
</tr>
<tr>
<td>READY</td>
</tr>
</tbody>
</table>

**Figure 17-82 Result of test case 3 LISTCONTROL**

<table>
<thead>
<tr>
<th>Openrule Entries:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
</tr>
<tr>
<td>Volume or Range Type</td>
</tr>
<tr>
<td>*</td>
</tr>
<tr>
<td>*</td>
</tr>
</tbody>
</table>

| READY |

**Figure 17-82 Result of test case 3 LISTCONTROL**
In Figure 17-83 we show the JES message log of this job. You can see there were no errors, because now the resource STGADMIN.EDG.IGNORE.TAPE.RMM.* is defined and the user has a minimum READ access to this resource. If there is a need to write to a volume ignored by DFSMSrmm, the user needs a minimum UPDATE access to the resource.

```
IEF403I OPENR004 - STARTED - TIME=14.49.10 - ASID=0042 - SC70
-  --TIMINGS (MINS.)--
- JOBNAME  STEPNAME  PROCSTEP  RC  EXCP  CPU  SRB  CLOCK  SERV  PG
-OPENR004   RACFCMDS    00   127  .00  .00  .00  1477   0
-OPENR004   LISTCONT    00    32  .00  .00  .00    309   0
IEF233A M 0B22,THS013,,OPENR004,TESTCASE,MHLRES7.TEST.OPENRULE.RULES
EDG4061I VOLUME THS013 IGNORED. IGNORE REQUESTED BY OPENRULE ACTION IGN
IEF234E K 0B22,THS013,PVT,OPENR004,TESTCASE
-OPENR004   TESTCASE    00    76  .00  .00  .96    258   0
IEF404I OPENR004 - ENDED - TIME=14.50.09 - ASID=0042 - SC70
```

**Figure 17-83**  JES message log of test case 3

**Test case 4**

**Function:**  Read an existing data set on an SMS managed tape volume using IEBGENER. The used volume serial number is THS013. That is normal tape processing. Figure 17-84 shows the JCL we used.

**Result:**  The job ended without an error because the OPENRULE definition is changed to ACCEPT all DFSMSrmm defined volumes and IGNORE only all volumes not defined in the DFSMSrmm control data set.

**OPENRULE:**  All DFSMSrmm defined tape volumes are accepted and all not-defined volumes are ignored on this system.

**RACF:**  The user has access to the data set profile protecting this data set. There are no need to have access to the special resources defined in RACF class facility:

```
STGADMIN.EDG.IGNORE.TAPE.RMM.*
STGADMIN.EDG.IGNORE.TAPE.RMM.*
```
Figure 17-84  Test case 4 sample JCL

Where:

**RACFCMDS** Shows the STGADMIN.EDG.* resources defined in the RACF class FACILITY.

**LISTCONT** Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command.

**TESTCASE** Uses the IEBGENER utility to read the data set.

Figure 17-85 shows the result of the RACF SEARCH command showing that the resources STGADMIN.EDG.IGNORE.TAPE.RMM.* and STGADMIN.EDG.IGNORE.TAPE.RMM.* are not defined at this time.

Figure 17-85  Result of test case 4 RACF listing

The result of the RMM TSO LISTCONTROL subcommand is shown in Figure 17-86. You can see that there is an OPENRULE TYPE(RMM) definition for all DFSMSrmm defined tape volumes with the ACCEPT option that allows the use of volumes defined in the DFSMSrmm control data set. All other volumes not defined in the DFSMSrmm control data set can only be used when the user has access to the resource defined in RACF class facility:

STGADMIN.EDG.IGNORE.TAPE.NORMM.volser
Test case 5

Function: In the DFSMSrmm control data set, we change the data set name DFSMSrmm have automatically recorded at O/C/EOV time for tape volume THS013 to simulate a duplicate tape volume or a foreign tape volume, that means a tape coming from outside. The data set name used in the JCL does no longer matched the data set name DFSMSrmm has recorded for this volume. To read this data set, we used IEBGENER too. The volume serial number is THS013.

Result: The job abended, although the user MHLRES7 have access to the data set, but the volume cannot be used on this system depending on the OPENRULE TYPE(NORMM) definition. This rule is used by DFSMSrmm, because the data set name used in our JCL does not match the data set name DFSMSrmm recorded for this volume.

OPENRULE: All DFSMSrmm defined volumes are accepted by the OPENRULE TYPE(RMM) ACCEPT and can be used under normal conditions but all volumes not defined in the DFSMSrmm control data set are ignored on this system depending on the OPENRULE TYPE(NORMM) IGNORE definition.

RACF: The user has access to the data set profile protecting this data set. There is a need to have READ access to the special resource:

STGADMIN.EDG.IGNORE.TAPE.NORMM.*
This is defined in the RACF class facility, because the volume is ignored from DFSMSrmm based on the OPENRULE definition TYPE(NORMM), but this resource is not defined at this time.

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*  
//SYSTSIN DD *             
  SR CLASS(FACILITY) MASK(STGADMIN.EDG) 
//*     
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD SYSOUT=*    
//SYSTSIN DD *            
  RMM TSO LISTCONTROL 
//*     
//TESTCASE EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*    
//SYSUT1 DD DISP=SHR,DSN=MHLRES1.TEST.OPENRULE.RULES, 
//         UNIT=ATL3, VOL=SER=THS013 
//SYSUT2 DD SYSOUT=*      
//SYSIN DD DUMMY
```

Figure 17-88  Test case 5 sample JCL

Where:

**RACFCMDS**  Shows the STGADMIN.EDG.* resources defined in RACF class FACILITY.

**LISTCONT**  Shows the OPENRULE definitions using the RMM TSO LISTCONTROL command.

**TESTCASE**  Uses the IEBGENER utility to read the data set.

Figure 17-89 shows the result of the RACF SEARCH command showing that the resources STGADMIN.EDG.IGNORE.TAPE.RMM.* and STGADMIN.EDG.IGNORE.TAPE.RMM.* are defined at this time. The resource STGADMIN.EDG.FORCE is now present because we needed to be able to change the O/C/EOV recorded data set name for this volume using the RMM CHANGEVOLUME subcommand with the FORCE option.

```
READ  
  SR CLASS(FACILITY) MASK(STGADMIN.EDG) 
STGADMIN.EDG.FORCE 
STGADMIN.EDG.LISTCONTROL 
STGADMIN.EDG.MASTER 
READY
```

Figure 17-89  Result of test case 5 RACF listing

The results of the RMM TSO LISTCONTROL and SEARCHDATASET subcommands are shown in Figure 17-90. You can see that there is an OPENRULE definition TYPE(NORMM) for all not DFSMSrmm managed volumes and that the definition has an action of IGNORE for any request to these volumes. Also you can see that the data set name recorded for this volume in DFSMSrmm does not match the data set name we are using in our JCL. This volume must be a foreign volume from the DFSMSrmm point of view, and so the OPENRULE with TYPE(NORMM) is used.
In Figure 17-91 we show the JES message log of this job. You can see that the job abended and cannot open the volume based on the OPENRULE definition. Also you can see the missing access to the STGADMIN.EDG.IGNORE.TAPE.NORMM.volser resource defined in RACF class FACILITY.

Note: The STGADMIN.** resource is used because there are no more specific RACF resources defined in the RACF class FACILITY. DFSMSrmm uses the normal RACF best matching processing to find a matching profile or resource. The user currently has no access to this definition.
Refer to “Test case 2” on page 271. to see the description of the EDGnnnnnc messages.

Test case 6

Function: In the DFSMSrmm control data set, we change the data set name DFSMSrmm has automatically recorded at O/C/EOV time for tape volume THS013 to simulate a duplicate tape volume or a foreign tape volume, that means a tape coming from outside. The data set name used in the JCL no longer matches the data set name DFSMSrmm has recorded for this volume. To read this data set, we used IEBGENER too. The used volume serial number is THS013. Figure 17-92 shows the JCL we used.

Result: The job ended without an error, because the user MHLRES7 have access to the data set and have the needed access to the resource STGADMIN.EDG.IGNORE.TAPE.NORMM.* depending on the OPENRULE TYPE(NORMM) IGNORE definition used for all volumes not defined in the DFSMSrmm control data set.

OPENRULE: All DFSMSrmm defined volumes are accepted by the OPENRULE TYPE(RMM) ACCEPT and can be used under normal conditions but all volumes not defined in the DFSMSrmm control data set are ignored on this system depending on the OPENRULE TYPE(NORMM) IGNORE definition.

RACF: The user has access to the data set profile protecting this data set and also access to the special resources:
STGADMIN.EDG.IGNORE.TAPE.NORMM.*
This is defined in RACF class facility, because the volume is ignored by DFSMSrmm based on the OPENRULE TYPE(NORMM) definition.

```
//RACFCMDS EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=*  
//SYSTSIN DD   * 
  SR CLASS(FACILITY) MASK(STGADMIN.EDG)
  /*
//LISTCONT EXEC PGM=IKJEFT01
//SYSTSPRT DD   SYSOUT=* 
//SYSTSIN DD   * 
  RMM TSO LISTCONTROL
  /*
//TESTCASE EXEC PGM=IEBGENER 
//SYSPRINT DD   SYSOUT=* 
//SYSUT1 DD   DISP=SHR,DSN=MHLRES1.TEST.OPENRULE.RULES,
  //UNIT=ATL3, VOL=SER=THS013 
//SYSUT2 DD   SYSOUT=* 
//SYSIN DD   DUMMY
```

Figure 17-92  Test case 6 sample JCL

Where:

RACFCMDS Shows the STGADMIN.EDG.* resources defined in RACF class
FACILITY.

LISTCONT Shows the OPENRULE definitions using the RMM TSO
LISTCONTROL command.

TESTCASE Uses the IEBGENER utility to read the data set.
Figure 17-93 shows the result of the RACF commands we used:

- We have deleted and then added in RACF class FACILITY the resource STGADMIN.EDG.IGNORE.TAPE.NORMM.*. This resource is used if a volume is not defined in the DFSMSrmm control data set.
- The resource STGADMIN.EDG.IGNORE.TAPE.RMM.* will be used if a volume is defined in the DFSMSrmm control data set.
- The user MHLRES7 has UPDATE access to the resource STGADMIN.EDG.IGNORE.TAPE.NORMM.*, but only READ access is needed to read a volume. It is normal RACF processing that means UPDATE access is a higher authorization than READ access and solves the request.
- The resource STGADMIN.EDG.IGNORE.TAPE.RMM.* is defined, but this resource is not used at this time.
The result of the RMM TSO LISTCONTROL and SEARCHDATASET subcommands are shown in Figure 17-94. You can see that there is an OPENRULE TYPE(NORMM) definition for all not DFSMSrmm managed volumes and that definition have an action of IGNORE. Any request to this volumes must be ignored by DFSMSrmm. Also you can see that the data set name for this volume recorded in DFSMSrmm not matched the data set name we are using in our JCL. This volume must be a foreign volume to DFSMSrmm and so the OPENRULE TYPE(NORMM) is used.

![Figure 17-94](image)

In Figure 17-95 we show the JES message log of this job. You can see the job ended without an error because of the correct access to the STGADMIN.EDG.IGNORE.TAPE.NORMM.* resource defined in the RACF FACILITY class.

![Figure 17-95](image)

Where:

**EDG4061I** \ VOLUME volser IGNORED. IGNORE REQUESTED BY OPENRULE ACTION IGNORE

**Explanation:** The DFSMSrmm PARMLIB option for OPENRULE with an action of IGNORE has requested that this volume be ignored by DFSMSrmm. DFSMSrmm does not record any information about the specified volume and permits the volume's use. This is because the user is authorized to use a volume that is ignored or DFSMSrmm is running in record mode.
In the message text:

**volser**  
The volume serial number of the volume that the user is attempting to use.

**System action:**  
DFSM斯mm ignores this volume while it remains mounted. DFSMSrmm does not validate the mounted volume and does not record any information about the current tape usage in the DFSMSrmm control data set.

**Operator response:**  
None.

**Source:**  
DFSM斯mm

**Detecting Module:**  
EDGOECM

**Routing Code:**  
2,3

**Descriptor Code:**  
3

### 17.7.6 Converting REJECT commands

Before installing z/OS V1.10, you must look into your existing ANYUSE definitions in order to plan usage of the PRTITION and OPENRULE commands. Review your existing REJECT commands across all system images and identify the sets of volumes to be used on each system. Identify the PRTITION and OPENRULE commands required for each system to minimize PARMLIB maintenance as your storage requirements change. Until you define one or more PRTITION or OPENRULE commands, both partitioning and rejecting of volumes are controlled by REJECT commands. Any REJECT commands that specify ANYUSE are used for partitioning of undefined volumes, but all REJECT commands are used for rejecting volumes at OPEN time.

When you use either PRTITION or OPENRULE commands, the REJECT commands are no longer used so you must start using both PRTITION and OPENRULE at the same time to avoid loss of function. You have to remove the REJECT commands from PARMLIB, because they will fail when any PRTITION or OPENRULE commands are defined. When PARMLIB is processed, DFSMSrmm issues message **EDG0239E**, followed by message **EDG0215D** to provide an option to ignore the error or fail and then retry with message **EDG0107D** with another PARMLIB member.

When you have created your new commands, remove the REJECT commands. The PRTITION and OPENRULE commands can only be used on z/OS V1R10 and later releases. Lower level releases continue to use the REJECT commands. To give you more choice and flexibility, conversion from REJECT commands should not be done one to one and should not be automated. When each REJECT command is converted strictly to an equivalent PRTITION and OPENRULE command, you can end up with too much complexity and duplication.

The best approach is to start from scratch and list the basic rules you want to implement. For example, consider the changed function and whether you are impacted by it:

- For volumes not defined to DFSMSrmm, the REJECT command with ANYUSE causes CUA requests to fail. With the PRTITION command, you can choose between the actions of either ACCEPT or IGNORE. You no longer have the option to fail a CUA request. The ACCEPT action results in the volume being added to DFSMSrmm and the CUA continues. The IGNORE action causes DFSMSrmm to take no action for the CUA request.
- For non-system-managed volumes that are not defined to DFSMSrmm, the default processing during OPEN did not add the volume to the CDS. The default processing for the PRTITION command is that non-system-managed volumes are added to the CDS.
Examples of converting REJECT commands

This section provides examples that show the conversion of REJECT commands to OPENRULE and PRTITION commands.

The sample commands in Figure 17-96 shows REJECT being used by both partitioning and for open time rejects. A REJECT with a prefix, when used at open time, applies only to volumes defined to RMM so TYPE(RMM) is used for OPENRULE. However, partitioning REJECT ANYUSE applies only to volumes NOT defined to RMM, so TYPE(NORMM) is used for PRTITION.

```
REJECT ANYUSE(prefix)

OPENRULE VOLUME(prefix) -
   TYPE(RMM) -
   ANYUSE(REJECT)

PRTITION VOLUME(prefix) -
   TYPE(NORMM) -
   SMT(IGNORE) -
   NOSMT(IGNORE)
```

Figure 17-96 Conversion of REJECT ANYUSE

The sample commands in Figure 17-97 show that REJECT is used only for open time rejects. A REJECT with a prefix, when used at open time, applies only to volumes defined to RMM, so TYPE(RMM) is used for OPENRULE. The REJECT OUTPUT allows input processing, so code the INPUT option on the OPENRULE for completeness, although ACCEPT is the default.

```
REJECT OUTPUT(prefix)

OPENRULE VOLUME(prefix) -
   TYPE(RMM) -
   OUTPUT(REJECT) -
   INPUT(ACCEPT)
```

Figure 17-97 Conversion of REJECT OUTPUT

The sample commands in Figure 17-98 show that REJECT is used by both partitioning and for open time rejects. A REJECT with PREFIX(*) applies only to volumes not defined to RMM, so TYPE(NORMM) is used for both OPENRULE and for PRTITION.

```
REJECT ANYUSE(*)

OPENRULE VOLUME(*) -
   TYPE(NORMM) -
   ANYUSE(REJECT)

PRTITION VOLUME(*) -
   TYPE(NORMM) -
   SMT(IGNORE) -
   NOSMT(IGNORE)
```

Figure 17-98 Conversion of REJECT ANYUSE(*)
The sample commands in Figure 17-99 show the following details:

- Three systems, SYSA, SYSB, and SYSC, are each allowed to use only a single range of system-managed volumes (JT, JP, JX).
- Each system has its own CDS and TCDB.
- System-managed volumes are not to be automatically defined to DFSMSrmm.
- Private volumes can be shared.
- Undefined non-system-managed volumes can be used for input.
- EDGUX100 is customized to automate the ignoring of private system-managed volumes from other partitions, and the ignoring of undefined non-system-managed volumes. This is based on the RMM API checking if a volume is defined to RMM. The following RACF FACILITY class profiles are also required:
  - STGADMIN.EDG.IGNORE.TAPE.NORMM.JP*
  - STGADMIN.EDG.IGNORE.TAPE.NORMM.JX*

```plaintext
/* SYSA example */
REJECT ANYUSE(JP*)
REJECT ANYUSE(JX*)
REJECT ANYUSE(*)

/* Allow read of undefined volumes */
OPENERULE VOLUME(*) -
  TYPE(NORMM) -
  OUTPUT(REJECT)

/* Ignore for input selected system managed volumes if authorized */
OPENERULE VOLUME(JP*) -
  TYPE(NORMM) -
  INPUT(IGNORE) -
  OUTPUT(REJECT)

OPENERULE VOLUME(JX*) -
  TYPE(NORMM) -
  INPUT(IGNORE) -
  OUTPUT(REJECT)

/* Global partition rule - ignore all smt volumes */
PRTITION VOLUME(*) -
  TYPE(RMM) -
  SMT(IGNORE) -
  NOSMT(ACCEPT)

PRTITION VOLUME(*) -
  TYPE(NORMM) -
  SMT(IGNORE) NOSMT(IGNORE)

/* This partition owns JT* volumes if predefined */
PRTITION VOLUME(JT*) -
  TYPE(RMM) -
  SMT(ACCEPT)
```

Figure 17-99  Conversion of REJECT ANYUSE - Three Systems
17.7.7 Conversion from REJECT to PRTITION and OPENRULE commands

If you want to implement the OPENRULE and PRTITION commands but are already using REJECT commands in PARMLIB, you must plan for conversion.

Until you define one or more PRTITION or OPENRULE commands, both partitioning and rejecting of volumes are controlled by REJECT commands. Any REJECT commands that specify ANYUSE are used for partitioning of undefined volumes, but all REJECT commands are used for rejecting volumes at OPEN time.

When you use either PRTITION or OPENRULE commands, the REJECT commands are no longer used, so you must plan to start using both PRTITION and OPENRULE at the same time to avoid loss of function.

You do need to remove the REJECT commands from PARMLIB because they will fail when any PRTITION or OPENRULE commands are defined. When PARMLIB is processed DFSMSrmm issues EDG0239E and then EDG0215D to provide an option to ignore the error or fail and then retry via EDG0107D with another PARMLIB member. When you have created your new commands, remove the REJECT commands.

The PRTITION and OPENRULE commands can only be used on z/OS V1R10 and later releases. Lower level releases continue to use the REJECT commands.

Conversion from REJECT commands is not best done one to one and so is not something which should be automated. The reason for this is that you have much more choice and flexibility and when each REJECT command is converted strictly to an equivalent PRTITION and OPENRULE command you can end up with too much complexity and also duplication. The best approach is to start from scratch and list the basic rules you want to implement; such as these:

- For partitioning, all non-defined volumes are ignored.
- For open rules, all volumes defined to DFSMSrmm can be used for both input and output.

When you have the basic rules, you can next identify sets of volumes and how each is to be treated/managed. For example the only volumes to be used on a system/partition might be all those volumes with the ‘A’ prefix. You can create a command that uses either VOLUME(A*) or VOLUME RANGE(A*:A99999) and select the TYPE based on whether you will always pre-define the volumes or allow them to be defined during cartridge entry and OPEN processing.

For the PRTITION commands, you must consider the system-managed and non-system-managed volumes separately. Be careful not to specify TYPE(RMM/ALL) with NOSMT(IGNORE) unless you really want EXPROC processing to skip the volumes. However, TYPE(RMM/ALL) with SMT(IGNORE) makes sense because DFSMSrmm causes OAM to ignore the volume and leave for another system/partition, and it also causes DFSMSrmm to skip EXPROC without checking if the volume is in the TCDB.

Consider the new function provided with the PRTITION and OPENRULE commands and whether it makes sense to exploit it:

- You can have separate commands based on whether the volume is defined to DFSMSrmm.
- OPENRULE provides an automated way to ignore volumes at open time without the need to customize DGUX100 or use EXPDT=98000 in JCL.
- OPENRULE also allows you to control use of existing data sets such as enforcing cataloging and reference only from the creating system.
Also consider the changed function and whether you are impacted by it.

- For volumes that are not defined to DFSMSrmm, the REJECT command with ANYUSE caused CUA requests to be failed. With the new PRTITION command, you can choose between the actions of either ACCEPT or IGNORE. You no longer have the option to fail a CUA request. The ACCEPT action results in the volume being added to DFSMSrmm and the CUA continues. The IGNORE action causes DFSMSrmm to take no action for the CUA request.

- For non-system-managed volumes that are not defined to DFSMSrmm, the default processing during OPEN was not to add the volume to the CDS. The default processing for PRTITION is that non system-managed volumes are added to the CDS.

Look at your use of EDGUX100, EDGUX200, and CBRUXENT installation exits to determine if they have been customized and might no longer be needed. The PRTITION and OPENRULE function should allow almost any customization related to partitioning, rejecting volumes, ignoring volumes, to be removed.

Your installation exit customization need not be removed immediately, but can be run alongside the new function, and only removed when you are satisfied that it is no longer needed:

- CBRUXENT, EDGUX200: Any existing customization of these exits can very likely be removed and replaced with use of PRTITION PARMLIB commands.

- EDGUX100: Any existing function that controls use of a volume, that might fail open or reject a volume, or enable volumes to be ignored without coding EXPDT=98000, can likely be removed and replaced by use of the OPENRULE PARMLIB commands.

**Partitioned library Figure 17-100 with shared CDS and TCDB**

The first example in Figure 17-100 shows a system-managed tape library partitioned by:

- Using CBRUXENT installation exits on each system
- Using EDGUX200 installation exit on each system
- Different scratch categories for each system in the library manager data base

The EDGHSKP expiration processing is running by system, but all volumes in the PRIVATE category are shared between the two systems.

Figure 17-100 shows that the new OPENRULE commands must be defined in EDGRMMnn PARMLIB member to replace the CBRUXENT and EDGUX200 installation exit modifications.
Partitioned library with shared TCDB and different CDSs

The example in Figure 17-102 shows a system-managed tape library partitioned by:

- A different DFSMSrmm control data set for each system
- Use of REJECT ANYUSE in EDGRMMnn PARMLIB member
- Different scratch categories for each system in the library manager database

The EDGHSKP expiration processing is running by CDS, but all volumes in the PRIVATE category are shared between the two systems. To access these volumes you have to specify one of the two JCL parameters EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 installation exit is needed.

Partitioned library with unshared CDS and TCDB

In the third example shown in Figure 17-103, you can see a system-managed tape library partitioned:

- By system
- By using REJECT ANYUSE in EDGRMMnn PARMLIB member
- By using different scratch categories for each system in the library manager data base

The EDGHSKP expiration processing is running by system. All volumes are not shared, so you have to define a volume in private status in the TCDB first. To access these volumes, you have to specify one of the two JCL parameters EXPDT=98000 or ACCODE=XCANORES. For both parameters, the default shipped EDGUX100 installation exit is needed.
This is a case where REJECT is used by both partitioning and for open time rejects. A REJECT with PREFIX(*) applies only to volumes NOT defined to RMM, so TYPE(NORMM) is used for both OPENRULE and for PRTITION.

Figure 17-104 shows the new PRTITION and OPENRULE commands must be defined in EDGRMMMnn PARMLIB member to replace REJECT ANYUSE and optionally the EDGUX100 installation exit.

```
OPENRULE VOLUME(*) /* added 2008/03/21 NS */-
   TYPE(NORMM) /* volumes not defined in CDS */-
   ANYUSE(REJECT BY(ANY)) /* bypass RMM processing */-
      /* ignore all types of request */
OPENRULE VOLUME(*) /* added 2008/03/21 NS */-
   TYPE(RMM) /* all volumes defined in CDS */-
   ANYUSE(ACCEPT) /* allow OPEN processing */

PRTITION VOLUME(*) /* added 2008/03/21 NS */-
   TYPE(NORMM) /* not defined in DFSMSrmm */-
   SMT(IGNORE) /* ignore SMS managed volumes */-
   NOSMT(IGNORE) /* ignore volumes outside an ATL/VTS */

SYSA:
PRTITION VOLUME(A*) /* added 2008/03/21 NS */-
   TYPE(ALL) /* volumes defined/not defined in CDS*/-
   SMT(ACCEPT) /* process the volume */-
   NOSMT(ACCEPT) /* process the volume */

SYS1:
PRTITION VOLUME(1*) /* added 2008/03/21 NS */-
   TYPE(ALL) /* volumes defined/not defined in CDS*/-
   SMT(ACCEPT) /* process the volume */-
   NOSMT(IGNORE) /* process the volume */
```
Partitioned library with shared CDS and unshared TCDB

In our last example in Figure 17-105, we show a system-managed library partitioned:

- By system
- By using customized versions of CBRUXENT installation exits on each system
- By using different scratch categories for each system in the library manager data base

The EDGHSKP expiration processing is running by system based on the TCDB. All volumes are not shared, so you have to define a volume in private status in the TCDB first before you can use it.

![Figure 17-105  Partitioned library with shared CDS and unshared TCDB](image)

**Note:** Figure 17-56 on page 256 shows how you can add a volume in the TCDB using AMS commands. After you have successfully processed the volume you have previously defined in the TCDB, you should delete the entry from the TCDB using the command as shown in Figure 17-57 on page 256.

This is a case where REJECT is used by both partitioning and for open time rejects. A REJECT with PREFIX(*) applies only to volumes NOT defined to RMM, so TYPE(NORMM) is used for both OPENRULE and for PRTITION.

Figure 17-106 shows the new PRTITION and OPENRULE commands that must be defined in EDGRMMnn PARMLIB member to replace REJECT commands and the customized CBRUXENT installation exit. The conversion to these commands allows you to define new volumes in the library manager data base and DFSMSrmm will accept the add of this volumes in the DFSMSrmm control data set if the prefix is matching the prefix we have specified in the PRTITION command. Also you can add the volumes first in the DFSMSrmm control data set and either the add volume using command on the library manager or inserting the volume into the library later.
Chapter 17. DFSMSrmm V1.10 enhancements

17.8 Common Information Model (CIM) provider

The DFSMSrmm CIM agent now has an option to register itself using the SMI-S Storage Library profile so that storage management clients (and Tivoli Storage Productivity Center in particular) can use Service Location Protocol (SLP) to detect the CIM agent and determine which registered profiles it can support.

DFSMSrmm now ships the web service for installation either as an Enterprise Archive (EAR) under WebSphere or as a Web Archive under Tomcat (WAR) or other web service middleware. For details of externals, handling, installation and customization refer to the manual, DFSMSrmm Tomcat Web Service User Guide.

17.8.1 Support of the latest CIM schema

Support is provided for the latest CIM Schema level - 2.14. This has caused a change in the key processing of the existing CIM classes provided by RMM CIM Agent. The RMM CIM classes are now inherited from more specific classes of the CIM Schema 2.14. This, on the one hand, avoids use of RMM-specific super-classes, but, on the other hand, requires a change in the attributes processing. IBMRMM_LogicalVolume, IBMRMM_PhysicalVolume, IBMRMM_Dataset classes, and their associations now have the following place in the inheritance tree;

Figure 17-106  Convert two TCDBs with one CDS to OPENRULE and PRTITION commands
The IBM RMM Control class preserved its place as a subclass of CIM_Service. Likewise, all the other associations preserved their place between the associated classes. Just as the RMM CIM classes included in the DFSMSrmm implementation of the SMI-S Storage Library profile do not use the CDSID as a key value, all the remaining RMMplex-wide CIM classes no longer use CdsID as the key. In order to provide similar capability the other existing keys of the classes now have compound formats as shown in Table 17-2.

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Key with CDS ID</th>
<th>Key Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBMRMM_Dataset</td>
<td>Name</td>
<td>DatasetName+FileSeq+Volser+CdsID – 4 concatenated strings¹ representing the 44-byte data set name, the 5-byte physical file sequence number, the 6-byte volume serial number, and variable length up to 8 bytes CDS ID of the RMMplex.</td>
</tr>
<tr>
<td>IBMRMM_Location</td>
<td>Tag</td>
<td>LocationName+LocationType+CdsID – 3 concatenated strings¹ representing the 8-byte location name, the 1-byte location type, and variable length up to 8 bytes CDS ID of the RMMplex.</td>
</tr>
<tr>
<td>IBMRMM_LogicalVolume</td>
<td>DeviceID</td>
<td>Volser+CdsID – 2 concatenated strings¹ representing the 6-byte volume serial number and variable length up to 8 bytes CDS ID of the RMMplex.</td>
</tr>
<tr>
<td>IBMRMM_Owner</td>
<td>InstanceID</td>
<td>OrgId:OwnerID+CdsID, where OrgId is a variable length copyright, trademarked or other unique name, and OwnerID+CdsID – 2 concatenated strings¹ representing the 8-byte owner name and variable length up to 8 bytes CDS ID of the RMMplex.</td>
</tr>
<tr>
<td>IBMRMM_PhysicalVolume</td>
<td>Tag</td>
<td>Volser+CdsID – 2 concatenated strings¹ representing the 6-byte volume serial number and variable length up to 8 bytes CDS ID of the RMMplex.</td>
</tr>
<tr>
<td>IBMRMM_PolicyRule</td>
<td>PolicyRuleName</td>
<td>RuleType+RuleName+JobNameMask+CdsID – 4 concatenated strings¹ representing the 1-byte policy rule type, the 44-byte policy rule name, 8-byte job name mask, and variable length up to 8 bytes CDS ID of the RMMplex.</td>
</tr>
<tr>
<td>IBMRMM_Product</td>
<td>IdentifyingNumber</td>
<td>ProductNumber+CdsID – 2 concatenated strings¹ representing the 8-byte product number and variable length up to 8 bytes CDS ID of the RMMplex.</td>
</tr>
<tr>
<td>IBMRMM_ShelfLocation</td>
<td>Tag</td>
<td>LocationName+MediaName+Number+CdsID – 4 concatenated strings¹ representing the 8-byte location name, the 8-byte media name, the 6-byte rack/bin number, and variable length up to 8 bytes CDS ID of the RMMplex.</td>
</tr>
</tbody>
</table>

¹ Strings are delimited with a ‘+’ and spaces are included
In order to simplify processing of the compound keys, every changed class now have KeyWithCdsIdName attribute containing the name of its compound key. Additionally, xxxFormat and xxxMask attributes are provided, where xxx - is the name of the compound key, to specify the symbolic format and mask correspondingly. For example, TagFormat property of IBMRMM_PhysicalVolume is set to “Volser+CdsID” and TagMask string contains the consecutive concatenation of 6 blanks symbol ‘+’ and 8 blanks.

Before the first IPL
For backward compatibility, RMM CIM Agent for V1R9 is provided as rmmcim19.tar.Z compressed tar archive within the /usr/lpp/dfsms/rmm directory. In order to use it on z/OS, you should go to the “Miscellaneous functions” panel provided by the rmmutil.sh script and invoke “Unpack RMM CIM Agent for V1R9”. As a result, you will have the old Agent unpacked in the “v1r9” path within current directory. For example if you use /usr/lpp/dfsms/rmm deployment directory the new Agent will reside in /usr/lpp/dfsms/rmm/v1r9. If you want to use the V1R9 Agent on Linux system you should resend individual unpacked files the usual way described in the rmmcim.txt readme file.

If you have CIM clients, which have been working on V1R9 level and you want to use them against V1.10, and the clients have some hard coded or specific processing of the keys of RMM CIM classes, you have the choice either:
1. To use the backward compatible agent as unpacked above, or
2. To change the processing according to the latest changes in keys of RMM CIM classes.
   The following table shows the old keys of RMM CIM classes and new compound keys, which have formats of concatenated strings containing the values of the old keys delimited with a ‘+’ and appended with spaces by the fix length if needed. Refer to the section ‘After the first IPL for details of setting up the Cim agent for V1R10.

After the first IPL
You have to delete all the RMM CIM classes from the CIM repository and also unregister all the RMM CIM provider modules. Then you add all the RMM CIM classes to the repository and register all the RMM CIM provider modules. The rmmutil.sh utility is the helper to automate the process.

CIM provider API for RMM
In order to exploit the CIM provider API for RMM, you must be running the OpenPegasus Common Information Model Object Manager (CIMOM) implementation, and be using a CIM browser/client or any equivalent product that supports the standards as specified in the xmlCIM in support of the SNIA Storage Management Initiative - Specification (SMI-S). The CIM server can run either on a non-z/OS server or directly on z/OS V1R8 or higher. The CIM client/browser can run on any platform supported by the provider of that client/application.

Use of the DFSMSrmm web service now requires either a WebSphere Application Server or a Tomcat Server running on the same z/OS system as the DFSMSrmm system to which the API commands are being sent. The release of WAS supported is unchanged. The release of Apache Tomcat supported is 5.5.x.

17.8.2 Integration with IRMM
The DFSMSrmm CIM Agent is updated to support a web service target of either WebSphere Application Server or Apache Tomcat. A new startup option is used to select your configured target web server type.
DFSMSrmm support for IRMM includes work required to get the RMM web service to run without requiring the complete WebSphere Application Server to be installed. This, initially, is enabled through use of Tomcat as an alternative web server (this is open source).

The registration of the profile is performed via binding an instance of a subclass of CIM_RegisteredProfile class to all managed instances of CIM_ComputerSystem subclasses. OpenPegasus provides its own implementation of the SMI-S Server profile and the engine for the binding of the vendor CIM classes and the profiles they correspond to. For this purpose, RMM CIM Agent initialization includes a step of creating the instances of the PG_ProviderProfileCapabilities and PG_ProviderReferencedProfiles classes. The instances refer to the IBMRMM_TapeLibrary class representing the tape libraries managed by DFSMSrmm.

An instance of IBMRMM_TapeLibrary is the anchor object of all the managed resources. Every instance of the IBMRMM_TapeLibrary corresponds to an RMMplex and should have NameFormat attribute set to “HID” and the Name key of the form Vendor+Product+Serial – 3 concatenated strings representing the 12-byte vendor name – “IBM “, the 16-byte product name – “TapeLibrary “, and variable length up to 8 bytes serial number representing the CDSID of the RMMplex (for example, “CDS001”). Strings are delimited with a ‘+’ and spaces are included. For instance, Name = “IBM +TapeLibrary +CDS001”.

The link to all the library components is done via association to its hardware components packaging all the other units. Every instance of the IBMRMM_TapeLibrary class corresponds to the set of instances of the IBMRMM_Location class associated with the tape library. When traversing the instances of the IBMRMM_TapeLibraryLocation association class with the specific instance of IBMRMM_TapeLibrary, you get the set of the instances of the IBMRMM_Location class, corresponding to all the DFSMSrmm EDGRMMxx PARMLIB member LOCDEF commands of the systems within the RMMplex.

**Implementing the DFSMSrmm CIM provider**

The readme file, rmmcim.txt, describes the necessary steps for CIM provider installation and configuration. See z/OS Common Information Model User’s Guide, available from the IBM z/OS web library, for details on how to setup the OpenPegasus CIM server under z/OS. If running the OpenPegasus CIM server under Linux, see the documentation that comes with the Pegasus package on how to setup the OpenPegasus CIM server under Linux.

**DFSMSrmm CIM classes**

The DFSMSrmm Common Information Model (CIM) provider application programming interface are Java classes that implement the CIM-specified methods required of providers. The CIM-classes provided by DFSMSrmm and the providers for those classes are defined in a Managed Object Format (MOF) file. Each of the classes are subclasses of a corresponding class of the CIM schema version 2.14, respectively SMI-S 1.2.

The DFSMSrmm CIM agent now has an option to register itself using the SMI-S Storage Library profile so that storage management clients (and TPC in particular) can use Service Location Protocol (SLP) to detect the CIM agent and determine the registered profiles that it can support. The registration of the profile is performed by binding an instance of a subclass of the CIM_RegisteredProfile class to all managed instances of CIM_ComputerSystem subclasses. OpenPegasus provides its own implementation of the SMI-S Server profile and the engine for the binding of the vendor CIM classes and the profiles they correspond to.

The CIM server reads and interprets the MOF file and calls the providers, as required. The DFSMSrmm-provided classes extend those of the standard CIM object model and enable DFSMSrmm to provide information about removable media managed by DFSMSrmm in real time.
These DFSMSrmm CIM classes are supported:

- IBMRRM_PhysicalVolume
- IBMRRM_LogicalVolume
- IBMRRM_Dataset
- IBMRRM_Owner
- IBMRRM_Location
- IBMRRM_ShelfLocation
- IBMRRM_Product
- IBMRRM_PolicyRule
- IBMRRM_Control
- IBMRRM_ChangerDevice
- IBMRRM_Manufacturer
- IBMRRM_SCSIProtocolController
- IBMRRM_SoftwareIdentity
- IBMRRM_TapeDrive
- IBMRRM_TapeLibrary
- IBMRRM_PhysicalLogicalVolume (association 1:1)
- IBMRRM_LogicalVolumeDataset (association 1:N)
- IBMRRM_LogicalVolumeOwner (association N:1)
- IBMRRM_PhysicalVolumeCurrentLocation (association N:1)
- IBMRRM_PhysicalVolumeDestinationLocation (association N:1)
- IBMRRM_PhysicalVolumeHomeLocation (association N:1)
- IBMRRM_PhysicalVolumeLoanLocation (association N:1)
- IBMRRM_PhysicalVolumeOldLocation (association N:1)
- IBMRRM_PhysicalVolumeRequiredLocation (association N:1)
- IBMRRM_PhysicalVolumeCurrentShelfLocation (association 1:1)
- IBMRRM_PhysicalVolumeDestinationShelfLocation (association 1:1)
- IBMRRM_PhysicalVolumeOldShelfLocation (association 1:1)
- IBMRRM_DatasetOwner (association N:1)
- IBMRRM_LogicalVolumeChainedLogicalVolume (association 1:1)
- IBMRRM_LogicalVolumeLogicalVolumeInChain (association 1:N)
- IBMRRM_LocationShelfLocation (association 1:N)
- IBMRRM_ProductLogicalVolume (association 1:N)
- IBMRRM_PolicyRuleNextPolicyRule (association N:1)
- IBMRRM_PolicyRuleAndPolicyRule (association N:1)
- IBMRRM_PolicyRulePolicyRuleInChain (association 1:N)
- IBMRRM_PolicyRuleLocation (association N:1)
- IBMRRM_PolicyRuleOwner (association N:1)
- IBMRRM_ChangerDeviceController (association N:M)
- IBMRRM_ChangerDeviceSoftware (association N:M)
- IBMRRM_LocationManufacturer (association N:1)
- IBMRRM_TapeLibraryLocation (association 1:N)
- IBMRRM_TapeDriveController (association N:M)
- IBMRRM_TapeDriveSoftware (association N:M)
- IBMRRM_SearchOperands (aux class for search type operations)
- IBMRRM_DeleteOperands (aux class for delete type operations)

**WAR installation steps**

To install the Web Archive under Tomcat (WAR) using the following steps:

2. Unpack into z/OS UNIX System Services
3. Create environmental variable
4. export CATALINA_HOME=tomcat_dir
5. Copy WAR file into folder $CATALINA_HOME/webapps
6. Tailor to your needs:
   a. $CATALINA_HOME/conf/server.xml
   b. $CATALINA_HOME/conf/saf-roles.xml
7. Start server by: $CATALINA_HOME/bin/startup.sh
8. Stop server by: $CATALINA_HOME/bin/shutdown.sh

To ensure that authorization can be performed for each web service request a RACF userid is required and authentication and authorization via SAF/RACF must be possible. Dovetail provides an open source class called SafRealm, which makes this pretty easy (see http://dovetail.com/docs/saf.html). See Figure 17-107 for how this is architected.

**Note:** DFSMSrmm does not ship any code which is Open source. Any code which we require which is Open source must be downloaded and installed for use with DFSMSrmm.

17.8.3 Using the Apache Tomcat server

To set up the DFSMSrmm Web service for the management of DFSMSrmm tasks, complete the following tasks.

1. Install Apache Tomcat.
2. Start/Stop Tomcat.
3. Deploy the Web Service.
4. Enable authentication and authorization.

In the following sections we discuss these tasks in detail.
Installing Apache Tomcat

Follow these steps for the installation:

1. As a prerequisite, the Java Runtime Environment 1.5 must be installed under z/OS. Java version 1.4 could also be used, if the “JDK™ 1.4 Compatibility Package” is downloaded from the Tomcat site.

2. Download the binary Core package “apache-tomcat-5.5.x.zip” from tomcat.apache.org. If Java 1.4 is to be used, also download the JDK 1.4 Compatibility Package “apache-tomcat-5.5.x-compat.zip”.

3. Upload the zip files to the Tomcat installation directory within the UNIX system services on your target z/OS system. The Tomcat installation directory must be at least 36 MB large, therefore the hosting mount dataset must have the corresponding number of cylinders.

4. Logon to the UNIX system services and unpack the zip-file by:

   jar -xvf apache-tomcat-5.5.x.zip
   jar -xvf apache-tomcat-5.5.x-compat.zip

   if applicable

   This will create a “apache-tomcat-5.5.x” sub-directory.

5. Ensure that these environmental variables are set:

   $CATALINA_HOME=tomcat_directory/apache-tomcat-5.5.x _BPXK_AUTOCVT=ON

   (Replace tomcat_directory by your actual Tomcat installation directory, and 5.5.x by the actual Tomcat version.) You can make these settings by appropriate “export” statements within $HOME/.profile or /etc/profile. For example:

   export CATALINA_HOME=$HOME/tomcat/apache-tomcat-5.5.x
   export _BPXK_AUTOCVT=ON

   Also verify that the $JAVA_HOME variable is set to the actual Java home directory and that the $PATH variable contains $JAVA_HOME/bin. The $LIBPATH must contain $JAVA_HOME/bin and $JAVA_HOME/bin/classic.

6. To make the Tomcat script files executable, tag them as follows:

   chtag -t -c ISO8859-1 $CATALINA_HOME/bin/*.sh

7. The DFSMSrmm Web service uses a C++ DLL from the z/OS Link List. The program object is called EDGXHCLL. To make this DLL available for the DFSMSrmm Web service, install a link in the systems library path ($LIB_PATH). Go to the library path in the file system, for example: /usr/lib, and type “ln -e EDGXHCLL libEDGXHCLL.so”. This step establishes an external link to the DLL in LINKLST.
Starting and stopping Tomcat
You must re-login to the UNIX system services to activate the previously set environmental variables.

To start Tomcat, type:
$CATALINA_HOME/bin/startup.sh

To stop Tomcat, type:
$CATALINA_HOME/bin/shutdown.sh

After having successfully started Tomcat, you will see its start page in your browser:
http://domain_name:8080system).

Note: To navigate to the “Tomcat Manager” page, you must implement one of the security schemes, as described below. Additionally, a user must exist that is mapped to role “manager”, to login to the “Tomcat Manager” page.

See Figure 17-108 for an illustration of authentication and authorization concepts.

Configuration
- SAF/RACF based authentication &
  authorization

![Diagram showing the process of SAF/RACF based authentication and authorization]

Where:
1. The Tomcat web server receives a service request via the network.
2. It reads “web.xml” to obtain the role-name ("master") for the requested service.
3. It reads “server.xml” to obtain both:
   - The role database (saf-roles.xml)
   - The active realm (saf_realm.jar)
4. It gets the required SAF authority for that role from the role database.
5. SAF/RACF is connected by the realm class, to authorize the incoming userid/password versus the required authority level.

The service request is executed if the password is correct (authentication) and the user has the required authority (authorization.)
Part 3

Extended Address Volumes

This part of the book contains the information you need to plan and implement Extended Address Volumes (EAVs) introduced with DFSMS V1.10.
Extended Address Volume introduction

In this chapter we provide the concepts and architectural overview of Extended Address Volume (EAV). The concept of Dynamic Volume Expansion introduced in the IBM System Storage™ DS8000, and how it can be used to migrate to EAV, are also covered in this chapter.
18.1 EAV overview

The Extended Address Volume (EAV) is the next step in providing larger volumes for z/OS. This support will be provided in z/OS Version 1 Release 10 of the operating system. DS8000 Release 4.0 Licensed Internal Code is also required for EAV support.

Rapid data growth on the z/OS platform is leading to a critical problem for some installations. The compound growth rate (CGR) for disk storage expansion from 1996-2007 was 37%. Business Resilience solutions such as GDPS®, HyperSwap and PPRC, that provide continuous availability are also driving this constraint. Over the years, IBM has grown volumes by increasing the number of cylinders and thus GB capacity. However, the existing track addressing architecture has limited our growth to relatively small GB capacity volumes which has put pressure on the 4-digit device number limit. For example, prior to EAV, the largest available volume is one with 65,520 cylinders or approximately 54 GB.

With EAV, IBM is implementing architecture that eventually will provide capacities of 100’s of terabytes for a single volume—with an architectural maximum of 268,434,453 cylinders. The first release of EAV in z/OS V1.10 will be limited to a volume with a maximum of 223 GB.

18.2 Terms and abbreviations

We first provide an overview of the terms and abbreviations used in this chapter and throughout the book. We use two sets of terms when referencing an EAV. The first set of terms deal with how space is managed, and the second set with how the disk is addressed.

- Basic definitions

  **EAV**
  
  Extended address volume.
  
  A volume with more than 65520 cylinders.
  
  Only 3390 Model A devices can be an EAV.

  **EAV capable**
  
  In this book we use the term EAV capable to describe a 3390 Model A volume that is smaller than 65520 cylinders. This volume can be expanded dynamically to more than 65520 cylinders - at which time it becomes an EAV.

  **Extended attribute DSCBs**
  
  New DSCB types (format 8 and format 9). A format 8 DSCB is equivalent to a format 1 DSCB. It is designed to protect existing programs that read format 1 DSCBs and examine extent descriptors that can contain 28-bit cylinder addresses.

  Format 9 DSCBs are a new type that contain additional metadata.

- How space is managed

  **multicylinder unit**
  
  A multicylinder unit is a fixed unit of disk space that is larger than a cylinder. Data sets allocated on an EAV in storage above the 65520 cylinder address will have the space request rounded to a multiple of the multicylinder unit. Currently, on an EAV, a multicylinder unit is 21 cylinders.
cylinder-managed space

The space on the volume that is managed only in multicylinder units.

Cylinder-managed space begins at cylinder address 65,520. Each data set extent occupies an integral multiple of multicylinder unit. Space requests targeted for the cylinder-managed space will be rounded up to the next multicylinder unit.

The cylinder-managed space is unique to Extended Address Volumes.

track-managed space

The space on a volume that is managed in tracks and cylinders.

Track-managed space ends at cylinder address 65,519. Each data set extent occupies an integral of multiple of tracks or cylinders. Track-managed space applies to Extended Address Volumes and non-Extended Address Volumes.

BPV

breakpoint value.

The BPV is expressed in cylinders. When the size of a disk space request is equal to or greater than the BPV, the system prefers to use the cylinder-managed space for that extent. This applies to each request for primary or secondary space for data sets that are eligible for the cylinder-managed space. If not enough cylinder-managed space is available, then the system will use the track-managed space or will use both areas. When the size of a disk space request is less than the BPV, the system prefers to use the track-managed area and if enough space is not available there, then the system will use the cylinder-managed space or will use both areas.

How the disk is addressed

track address

A 32 bit number that identifies each track within a volume. It is in the format hexadecimal CCCCccccH, where CCCC is the low order 16 bits of the cylinder number, ccc is the high order 12 bits of the cylinder number, and H is the four-bit track number.

For compatibility with older programs, the ccc portion is hexadecimal 000 for tracks in the base addressing space.

EAS

extended addressing space

On an EAV, the cylinders whose addresses are equal to or greater than 65,536. These cylinder addresses are represented by 28-bit cylinder numbers.

base addressing space

On an EAV, the cylinders whose addresses are below 65,536. These cylinder addresses are represented by 16-bit cylinder numbers or by 28-bit cylinder numbers.
18.3 Key design points regarding the EAV space

An important design point is that IBM maintains its promise that the 3390 track format and image size, and tracks per cylinders, will remain the same on an EAV as on other 3390 model devices. An application using data sets on an EAV is comparable to how it runs today.

18.3.1 Track-managed space and cylinder-managed space

An EAV has two managed spaces, the track-managed space and the cylinder-managed space as shown in Figure 18-2.

The track-managed space is managed in tracks and cylinders increments. All volumes today have track-managed space. Track-managed space ends at cylinder address 65,519. Each data set occupies an integral multiple of tracks.

The cylinder-managed space is the space on the volume that is managed only in multicylinder units. An multicylinder unit is a fixed unit of disk space that is larger than a cylinder. Cylinder-managed space begins at cylinder address 65,520. Each data set occupies an integral multiple of multicylinder units. Space requests targeted for the cylinder-managed space will be rounded up to the next multicylinder unit. In V1.10 cylinder-managed space only exists on Extended Address Volumes. Currently, on an EAV, a multicylinder unit is 21 cylinders and the number of the first cylinder in each multicylinder unit is a multiple of 21.
The reason that a multicylinder unit is 21 cylinders is derived from being the smallest unit that can map out the largest possible Extended Address Volume and stay within the architecture of the VTOC index (with a block size of 8192 bytes). It is also a value that divides evenly into the 1 GB storage segments of a DS8000. These 1 GB segments are the allocation unit in the DS8000 and are equivalent to 1113 cylinders. Data sets allocated in cylinder-managed space can have their requested space quantity rounded up to the next multicylinder unit.

The track-managed space allows existing programs and physical migration products to continue to work. You can perform physical copies from a non-EAV to an EAV and have those data sets accessible.

**Note:** Track-managed space is compatible to same space on non-EAVs.

Extended Address Volume provides a method to protect existing programs from seeing how the tracks in cylinder-managed space are addressed. We do that with new Data Set Control Blocks (DSCBs) in the VTOC and a new track address format.

**EAV addressing**
We have discussed that cylinder-managed space and track-managed space describe how space is managed. Next we look at how the EAV is addressed through its new track address format.

The EAV has two addressing spaces, the base addressing space and the Extended Addressing Space (EAS) as shown in Figure 18-3.
**Base addressing space**

The base addressing space is the area on an EAV located within the first 65,535 cylinders. Tracks are addressed in this area with 16-bit cylinder numbers, described with the traditional CCHH notation. The CC represents 16-bits for a cylinder address. The HH represents 16-bits for a track address, which only the low order 4-bits are used. This is how all disks are addressed prior to z/OS V1.10. This area is similar to the track-managed space but has a larger set of tracks/cylinders.

**Note:** Base addressing space is compatible with addressing space on a non-EAV.

**Extended Addressing Space**

The Extended Addressing Space (EAS) is the area on an EAV located beyond the first 65,535 cylinders. Tracks are addressed in this area with 28-bit cylinder numbers, described with the CCCCcccH (showing the hex digits) notation. The CCCC represents the low order 16-bits of a 28-bit number. The ccc represents the high order 12-bits of a 28-bit number. The H represents a 4-bit track number. This addressing is compatible to all 16-bit cylinder addressing. We often interchange the terminology of EAS and cylinder-managed space because this area is similar to the cylinder-managed space.

To protect existing programs from incorrectly accessing tracks in the EAS, new DSCBs in the VTOC are created for EAS-eligible data sets.

**28-bit cylinder track address**

This 28-bit cylinder track addresses is a 32 bit number that addresses each track within a volume. Traditionally each cylinder and track number uses a 16-bit number. However, for the track number, only the low order 4-bits are used, where the high order 12-bits of the track number are not.
The maximum supported volume size today is 65,520 cylinders; this is near the 16-bit theoretical limit of 65,535 cylinders. To handle cylinder numbers greater than 65,520, a new format for the track address is required. The new track addressing method is described with CCCCcccH notation, showing hex digits of 4-bits each.

This track address is a 32-bit number that identifies each track within the base addressing space. It is in the format hexadecimal CCCCCcCcH, where CCCC is the low order 16 bits of the cylinder number, ccc is the high order 12 bits of the cylinder number, and H is the four-bit track number. For compatibility with older programs, the ccc portion is hexadecimal 000 for tracks in the base addressing space. This track address method is referred to as a 28-bit cylinder number.

Key points and considerations
In this section we discuss some key points for your consideration:

- The cylinder number is in a non-linear form, a program or person reading this track address needs to re-arrange the bits in the program or in their minds. This format preserves the 3390 track geometry.
- Track addresses for space in track-managed space will be compatible to today’s track addresses.
- Track addresses for space in cylinder-managed space will not be compatible with today’s track addresses.

Additional documentation:

- z/OS V1.10 DFSMS Using Data Sets, SC26-7410:
  - In this manual you will find information about data sets (types and attributes), allocation, access method, volumes, and so on.
- z/OS V1.10.0 DFSMSdfp Advanced Services, SC35-0428:
  - In this manual you will find information about VTOC usage and the VTOC DSCBs.

18.3.2 Terminology and EAV considerations
In this section we discuss track addressing.

Absolute track addresses
Absolute track addresses generally are represented as cchh or bbechh and the term generally is shortened to track addresses. Programs that reference these most likely will require changes if processing EAS-eligible data sets.

Track addresses in existing channel programs and extent descriptors in DSCBs and elsewhere are in the form of CCHH, where CC is the 16-bit cylinder number and HH is the 16-bit track number in that cylinder. If the volume is an EAV, the cylinder number in these four CCHH bytes will be 28 bits and the track number will be 4 bits. For compatibility reasons, the 32 bits in each track address on an EAV will be in the CCCCCcccH format:

The 12 high order bits of the cylinder number are in the high order 12 bits of the two old HH bytes. This format is written as CCCCCcCcH. You can compare two of these track addresses for equality but you cannot reliably make any calculation including using a simple comparison for greater than or less than. Any arithmetic computation must take this special format into consideration. IBM recommends using the TRKADDR macro for any track address calculations.
Absolute block address
An "absolute block addresses is of the form cchhr or mbbcchhr and includes an absolute track address. Often they are written in capitals in the documentation as CCHHR or MBBCHCHHR.

Relative block address
Relative block addresses are written as TTR or TTTR. The TT or TTTR bytes count tracks relative to something. If they are relative to the beginning of a data set and inside the data set, they are unaffected by this project. If they are relative to the beginning of the volume, they might be affected by this project. Relative block addresses often are referred to as being relative track addresses, although that term should be reserved for a 2-byte, 3-byte, or 4-byte number that counts tracks relative to the logical beginning of a data set and inside its extents or a number that is relative to the beginning of the volume.

Conversions of relative track addresses to and from absolute format (MBBCCHHR) are supported automatically in the existing routines pointed to by CVTPRLTV and CVTPCNVT

Track calculations using the TRKADDR macro
You can use the TRKADDR macro introduced in DFSMS V1.10 for all track address comparisons and calculations. Programs should not need to do their own 28-bit cylinder manipulation. Even if you have programs not directly affected by this support for VSAM data sets using EAS. Using this new macro for all track address computations prepares you for additional changes in this support. You can run programs that are assembled with the TRKADDR macro on previous releases of z/OS, but not necessarily on machines that do not support the z/Architecture®.

This new macro supports these functions:
- ABSTOREL calculates the relative track number on the volume from the passed ‘cchh’ track address.
- COMPARE compares two track addresses in the new CCCCcccH format.
- EXTRACTCYL extracts the 28-bit cylinder number from the passed ‘cchh’ track address.
- EXTRACTTRK extracts the 4-bit track number from the passed ‘cchh’ track address.
- NEXTTRACK increments the track address by one track and increments the cylinder number if necessary from the passed ‘cchh’ track address.
- NORMALIZE reverses the 16-bit and 12-bit portions of the cylinder number from the passed ‘cchh’ track address. The CCCCcccH becomes cccCCCCH. This could be used to subsequently perform unsigned comparisons of track addresses.
- NORMTOABS reverses the 12-bit and 16-bit portions of the cylinder from the track address. The cccCCCCH becomes CCCCcccH. Use this to convert a normalized track address to an absolute 28-bit cylinder address.
- RELTOABS converts a relative track number on the volume (counting across cylinder boundaries) to a 28-bit cylinder address.
- SETCYL converts a cylinder number to a 28-bit cylinder address and sets the track portion to zero. This macro is available only in z/OS 1.10 but the expansion will run equally well on lower level systems if the high order 12 bits of the track number are zero.

Track calculations using the IECTRKAD routine
Programs that are written in a high level language such as C, C++, COBOL or PL/I can call this routine for same functions in TRKADDR. This routine is available only on z/OS 1.10 but programs linked with it can run on lower level releases.
The normalized cylinder-track address

To assist in the understanding of 28-bit cylinder numbers, a normalized cylinder-track address can be used for printing. This is where the bits are rearranged to a more readable format using a linear 28-bit cylinder number. The presence of a colon before the last hex digit identifies the track address as being normalized. The normalized cylinder-track address should be used only for printing.

The normalized cylinder-track format is:

cccCCCC:H

Where cccCCCCC is a 28-bit cylinder number in a contiguous form:

- H is a 4-bit hex track number (decimal 0-14).
- The colon shows that it is a normalized address.
- For instance, 0010000:E describes Cylinder 65,536 track 14.

18.4 New extended attribute DSCBs

Data Set Control Blocks (DSCBs) are Volume Table of Content (VTOC) entries that describe information on a volume such as data set attributes and allocated extent information. For data set data related DSCBs, this extent information describes allocated space using beginning and ending track addresses.

Extended attribute DSCBs are new DSCB types that provide a method of protecting existing programs from seeing unexpected track addresses (28-bit cylinder numbers) in its extent descriptors and from seeing additional information that the program might not understand.

Format 8 DSCB replaces the format 1 DSCB for EAS eligible data sets. In DFSMS V1.10 they are allowed only on EAVs. It is essentially equivalent to a format 1 DSCB however, its chain pointer always points to a format 9 DSCB. In z/OS V1.10, the format 9 DSCB contains a list of pointers to possible format 3 DSCBs. In addition, it contains fields that are available for vendor programs to use. Its chain pointer contains zeroes or points to an additional format 9 or format 3 DSCB.

In order to access extended attribute DSCBs, the system will require the specification of a new permission keyword, EADSCB=OK, on system services that read DSCBs, opens a VTOC on an EAV, or opens a VSAM data set for EXCP access. By specifying EADSCB=OK, the invoking program is indicating to the system service that it understands extended attribute DSCBs and the 28-bit cylinder numbers that could be present in the data set's and volume's extent descriptors.

Not specifying EADSCB=OK will cause these services to fail if issued to a data set that has extended attribute DSCBs or, for volume level services, a volume that supports cylinder-managed space (for example CVAFDSM). It will not fail for data sets on EAVs that do not have Format 8 or Format 9 DSCBs. Code this keyword when your applications support EADSCB. It should be specified on all invocations of each service regardless or whether the application runs on pre-z/OS V1.10 systems or accesses volumes that do not support extended attribute DSCBs.

In z/OS V1.10, each data set that has a format 8 DSCB has exactly one format 9 DSCB. In a future release, IBM might define further subtypes of the format 9 DSCB. They would be chained after the format 9 DSCB described here. If your program follows the DSCB chain, it should allow for extra format 9 DSCBs.
18.5 New volume attributes

In z/OS V1.10, the following volume attributes are defined:

- Volume has extended addressing space (track addresses with cylinder numbers greater than 65535).
- Volume has cylinder-managed space (space always in units of 21 cylinders).
- Volume can have extended attribute DSCBs (format 8 and format 9).

An EAV has all three attributes, while a non-EAV has none of these attributes. Each of these attributes should be tested separately because, in a future release, they might be independent of each other. For example, in a future release it might be possible for a non-EAV to support format 8 and 9 DSCBs (extended attribute DSCBs).

Identifying volume attributes

You can identify the new volume attributes from the following three sources:

- The format 4 DSCB, which describes the extent and contents of the VTOC. It also provides volume and device characteristics. This DSCB is mapped by the IECSDSL1 macro. Each format 4 DSCB label begins with ‘DS4’.
- The Device Class Extension (DCE) of the UCB that is used to describe device characteristics and supported feature information for a given UCB. The DCE is mapped by IECDDCE. Each label begins with ‘DCE’.
- The DEVTYPE macro, that is used to obtain device characteristics information about I/O devices. Among others, the information includes the number of cylinders. To obtain the new information code INFO=DASD. DEVTYPE returns information that is mapped by the IHADVA macro. Each label begins with ‘DVA’.

These three sources contain the following information:

- Each xxxMCU label describes the minimum allocation size in cylinders for cylinder-managed space. The number of cylinders in each extent in this space must be a multiple of this value. This minimum allocation size is referred to as the multicylinder unit. It is the smallest unit of disk space in cylinders that can be allocated in cylinder-managed space. The label names are DS4MCU, DCEMCU, and DVAMCU.
- Each xxxCYLMG label indicates that cylinder-managed space exists on the volume. The label names are DS4CYLMG, DCECYLMG and DVACYLMG. Flag xxxEADSCB will also be set.
- The xxxLCYL label indicates the first cylinder address (divided by 4095) where space is managed in multicylinder units. Cylinder-managed space begins at this address. It is valid when xxxCYLMG is set. The label names are DS4LCYL, DCELNCYL and DVALCYL.
- The xxxEADSCB label indicates that extended attribute DSCBs (format 8 and format 9 DSCBs) are allowed on this volume. The label names are DS4EADSCB, DCEEADSCB and DVAEADSCB.
- The volume size information has been relocated to new four-byte fields labeled as DCEUDCY, DS4DSCYL and DVAICYL. These are set for all volumes. For an EAV, the number of user data cylinders in the DCE (16-bit DCEUDCYL) and format 4 DSCB (16-bit DS4DSCYL) will be set to x’FFFE’ (65534). This can be used to identify the volume as being an EAV. An EAV will also have the xxxCYLMG and xxxEADSCB flags set.

Be sure to check the correct attributes in your program. If your program needs to know if a volume supports extended attribute DSCBs then check the EADSCB flags and not the old volume size field containing x’FFFE’.
18.6 EAS eligible data sets

An EAS eligible data set is a data set on an EAV that is eligible to have extents in the EAS. Eligible EAS data sets can be created or extended anywhere on an EAV. Data sets that are not eligible for EAS processing can only be created or extended in the track-managed portions of the volume.

In z/OS V1.10, the following VSAM data sets are EAS eligible:
► All VSAM data types (KSDS, RRDS, ESDS, and Linear).
► See “EAS ineligible data sets” for exceptions.

This includes both SMS managed and non-SMS managed VSAM data sets.

DB2, CICS and zFS support EAS eligible VSAM data sets. IMS™ support is planned to be available shortly after general availability of EAV.

18.7 EAS ineligible data sets

An EAS ineligible data set is a data set that can exist on an EAV but is not eligible to have extents (through Create or Extend) in the EAS. The following data sets are not EAS eligible:
► Catalogs
► VTOC (continues to be restricted to within first 64K-1 tracks)
► VTOC index
► VVDS
► Page data sets
► VSAM data sets with imbed or keyrange attributes
► VSAM data sets with incompatible control area sizes
► Non-VSAM data sets

In a future release, some of the foregoing data sets might become EAS eligible.

**Note:** All data set types can be allocated in the track-managed space of an EAV.

18.8 Free space statistics on EAV

Free space on the volume today is kept, reported and processed by the system on a total volume basis. For a device that supports cylinder-managed space, free space information is kept, reported and processed by the system using the total volume free space statistics and also free space statistics pertaining to the track-managed space. These two sets of free space statistics are maintained in the index data set for quick access from functions like LSPACE for volumes that have cylinder-managed space.

Because all volumes have track-managed space, free space statistics from track-managed space are reported on all volumes. The total volume and track-managed space statistics are the same for volumes that do not support cylinder-managed space.
The system maintains the following free space statistics for each volume:
- Total volume statistics (as reported prior to z/OS V1.10)
- Track-managed statistics (new with DFSMS V1.10)

The free space is computed by the system and kept in the VTOC index if available (new with DFSMS V1.10).

**Note:** Ensure that EAVs have an active index and that system management procedures are in place to detect and rebuild an index if it gets disabled. Non-indexed volumes with many data sets might perform poorly.

Listed here are the functions and products that report on the two sets of free space statistics:
- LSPACE, ISPF, ISMF, IEHLIST, DISKMAP, DITTO, File Manager
- SMF type 19 record
- DCOLLECT provides new volume statistics in the following records:
  - SMS Volume Definition Record (Record type ‘VL’).
  - VOLUME Record Field (Record Type ‘V’).

For non-EAV, track-managed statistics is equivalent to total volume statistics.

### 18.9 Breakpoint value

The breakpoint value (BPV) directs placement of EAS-eligible data sets on EAV. The BPV is a value in cylinders. If the requested space for a data set allocation is greater than or equal to the BPV (\( \geq \) BPV) then cylinder-managed space is preferred. If the requested space for allocation is less than the BPV (\(< \) BPV) then track-managed space is preferred.

In either case if the preferred space cannot satisfy the space request then the entire volume becomes eligible.

**Defining BPV**
The BPV value is defined as a Storage Group attribute for those specific volumes, or in the IGDSMSxx SMS PARMLIB member for all volumes in the system that do not have a storage group BPV attribute specified. If the BPV is not specified in the Storage Group or in the IGDSMSxx PARMLIB member, a system default of 10 cylinders is used. The BPV can be a value of 0-65520 cylinders. The value can be changed dynamically in the Storage Group definition or via the SETSMS operator command.

Following are additional BPV key points:
- Storage group BPV overrides system-level BPV.
- System BPV can be changed via SETSMS BreakPointValue(0-65520).
- A value of 0 means that cylinder managed space is always preferred.
- A value of 65520 means that track managed space is always preferred.

### 18.10 EAV space considerations

In cylinder-managed storage a space request is rounded up to next multicylinder unit if the entire extent is allocated in cylinder-managed space. Individual extents always start or end on a multicylinder unit boundary in cylinder-managed space.
This applies to the creation of data sets, extending them with more space and also releasing unused space (partial release).

A given extent is contained in a single managed space, which means that an extent cannot straddle the track-managed space and cylinder-managed space boundary. A given extent will always start and end in track-managed space or start and end in cylinder-managed space. No extent will be allowed to start in track-managed space and end in cylinder-managed space.

Keep in mind that the exact space requested will be obtained if some portion of the space is allocated in track-managed space.

**Algorithms used to satisfy space request**

Current algorithms in the search for space continue to apply with EAV. The system will select the first free space extent from the start of the preferred managed-space that can satisfy the requested quantity. When the space cannot be satisfied in one extent, additional extents can be used to satisfy the request where the unused extents are ranked from the largest to smallest in the preferred managed space.

When the request cannot be satisfied from the preferred managed space as determined by the BPV, the available free space extents from the entire volume are ranked and used. Space could be from the non-preferred area or from both track-managed and cylinder-managed spaces.

When the system extends an SMS VSAM data set and the secondary space must be returned in multiple extents due to volume fragmentation, prior to DFSMS V1.10 the system extends the last current extent only if the first new extent happens to be contiguous. In DFSMS V1.10 the system extends the current last extent if any of the new extents is contiguous with it.

**VSAM striping considerations**

Because VSAM striped data sets require all stripes to be the same size, the system will attempt to honor the exact requested space using a combination of cylinder-managed and track-managed spaces. Stripes will be allowed in both track-managed and cylinder-managed storage.

If the exact requested space cannot be obtained for all stripes, the system will round up the requested space to next multicylinder unit for all stripes.

**Note:** When the system attempts to get the exact space returned, it might return multiple extents, where it used to be one.

For VSAM striping, all stripes must have a common “release point”—a common Relative Block Address based on multicylinder unit or control area boundaries. For partial release, some or all of the unused space might not be released because extents must end on a multicylinder unit boundary.

### 18.11 Allocating data sets

The introduction and use by the system of an EAV is determined by a combination of adding the volumes to SMS storage groups and adding them to non-SMS managed storage pools (specific or esoteric volumes). Volumes could be in their dedicated storage group/pool or mixed with other volumes. Adding and enabling them to these pools allows the system to
allocate VSAM data sets in cylinder-managed space if the USEEAV PARMLIB setting is set to YES. When USEEAV is set to NO, no new data set creates (even non-VSAM) will be allowed on any EAV capable volume in the system that has more than 65520 cylinders.

As we discussed previously, VSAM and non-VSAM data set types can reside in the track-managed space. VSAM data sets, however, can be allocated anywhere on the volume. The preference on how the system determines which managed space should be used is based upon the derived BPV for the target volume of the allocation.

**USEEAV PARMLIB options**
The following option is for USEEAV PARMLIB:

- USEEAV(YES|NO)
  - This controls whether EAV is allowed for data set initial allocations and EOV extends to new volumes. It has no effect on EAVs coming online.
  - It is specified in SMS PARMLIB member IGDSMSxx.
  - It defaults to NO if SMS is active.
  - It defaults to YES if SMS is not active.
  - It can be changed via SETSMS USEEAV(YESINO) when SMS is active.
  - It has no effect for data set extend to same volume.
  - With USEEAV(NO), a new data set allocation that does not fit into the track-managed space on any available volume results in an IGD17279I message that indicates the USEEAV(NO) setting.

EAVs can be SMS-managed or non-SMS managed. SMS storage groups and non-SMS managed volume pools can be defined with all EAV, all non-EAV or a mix of EAVs and non-EAVs.

### 18.12 Allocation and migration thresholds

The allocation and migration thresholds in the system have an effect on how SMS selects volumes and DFSMShsm (HSM) conducts space management.

For each volume with cylinder-managed space, SMS and HSM use two sets of values, the first being the total volume free space threshold (as is used on non-EAVs), and a new one for track-managed space. They are specified in the storage group or, for non-SMS managed, as HSM parameters. These two thresholds are recorded in DCOLLECT’s storage group (record type SG).

**SMS volume selection regarding thresholds**

For SMS processing, two thresholds are used to determine whether a volume could be put on the list for selection.

The primary threshold is the more important threshold, however both the primary and secondary threshold must be met.

- **Primary threshold:**
  - When requested space >= BPV, the Extended Address Volume meets the primary threshold if the cylinder-managed space can remain below threshold.
  - When requested space < BPV, the Extended Address Volume meets the primary threshold if the track-managed space can remain below threshold.
- The non-Extended Address Volume meets the primary threshold if the entire volume has sufficient below-threshold space.

**Secondary threshold:**
- An Extended Address Volume or non-Extended Address Volume meets the secondary threshold if the entire volume has sufficient below-threshold space.

**HSM space management**
For HSM processing, track-managed threshold was added to ensure that track-managed space gets space managed even when overall volume has not exceeded the threshold value.

### 18.13 Volume selection

SMS requests will not use EAV if the USEEAV setting is set to NO. Specific allocation requests will be failed and for non-specific allocation request (UNIT=SYSDA) EAV will not be selected. No space messages will be returned when non-EAVs are not available.

For non-EAS eligible data sets, all volumes (EAV and non-EAV) are equally preferred (or they have no preference). This is the same as today, with the exception that Extended Address Volumes will be rejected when the USEEAV PARMLIB value is set to NO.

During the volume selection process, candidate volumes are each assessed for their capability with respect to attributes that are pertinent to volume selection. One of the key attributes, among others, is whether a volume can remain below its high threshold for free space after the requested space has been allocated. This is the THRESHOLD requirement. Volumes that cannot meet this requirement are less preferred than volumes that do meet this requirement. For VSAM allocations, the primary and secondary thresholds are applied as described previously and the volumes are ranked.

Extended Address Volumes are preferred over non-Extended Address Volumes when requested space is $\geq$ BPV. There is no EAV preference when requested space $< $ BPV. For VSAM striping allocations, SMS prefers EAVs that can provide the exact requested space for each stripe.

### 18.14 VSAM control area

VSAM data sets created in z/OS V1.10 (EAV and non-EAV) can have different CA sizes, from what would have been received on prior releases because the CA must be compatible with the multicylinder unit for cylinder-managed space. A migration consideration here is that VSAM data sets physically copied from a non-EAV to an EAV might have an incompatible CA and thus would not be EAS eligible. This means extends for additional space would not use cylinder-managed space:

- Compatible CA sizes are 1, 3, 5, 7, 9 and 15 tracks (they evenly divide into 21 cylinders which equals 315 tracks).
- The compatible CA size allows VSAM data sets to be EAS-eligible. VSAM data sets allocated with compatible CAs on a non-EAV are eligible to be extended to additional volumes that might be one that supports cylinder-managed space.
- For initial allocation, the system can adjust the CA size and the primary and secondary space amounts. For example, TRK(24,4) will have a CA of 5 tracks, with a primary and secondary amount of 25 and 5 tracks.
In this section we discuss the enhancements to DFSMSdss and EAV. Performance improvements have been made in these functions by replacing the method of maintaining the locations of data set extents and free space with a more efficient method.

**DEFRAG and CONSOLIDATE performance improvement**

The DEFRAG function consolidates the free space on a volume to prevent out-of-space conditions on new allocations. DEFRAG accomplishes this by relocating data set extents on a DASD volume to reduce or eliminate free space fragmentation, and prints a report about free space and other volume statistics.

The CONSOLIDATE function consolidates data set extents on a volume to improve performance when reading data sets. CONSOLIDATE accomplishes this by combining contiguous extents of a data set into one extent and by relocating multiple non-contiguous data set extents into contiguous space and prints a report of volume statistics.

As volumes have increased in size, the time that it takes to process them with DEFRAG has also increased. The larger volume size inherently requires more time to process the increased number and size of data sets. This increased processing time affects most customers because they have limited time in which to run maintenance tasks on their systems. CONSOLIDATE, being a function that is extracted from DEFRAG, shares the same performance problems.

**DEFRAG**

The DEFRAG Version 2, which is new for z/OS V1.10, brings performance enhancements for EAV and non-EAV:

- It no longer supports the CONSOLIDATE keyword (refer to “CONSOLIDATE command” on page 319).
- If CONSOLIDATE is specified, you will receive an informational message indicating that the keyword is no longer supported.

The following new keywords are provided:

- **MMOVPC(n,p)**
  
  Stops the DEFRAG run when n% contiguous tracks on the volume are assembled as free. If n% contiguous tracks already exist as free tracks, the DEFRAG function tries to further reduce the fragmentation of the volume but no more than n% tracks are relocated. If more than n% tracks must be relocated, no DEFRAG is performed.

  - **n** The percentage of tracks on the volume that DFSMSdss is to try to assemble as free tracks in a contiguous area.
  - **p** The number of passes DFSMSdss is to make in attempting to assemble the tracks.

  The MMOVPC keyword is recommended instead of MAXMOVE when running DEFRAG on an EAV. The MMOVPC will apply separately to the track-managed space and the cylinder-managed space.

  - MAXMOVE and MMOVPC are mutually exclusive.

- **MAXTIME(nummins)**

  Specifies the maximum number of minutes that the DEFRAG function should be allowed to process. This allows the user to control the time that the job will run. MAXTIME will be checked after processing each data set. If the MAXTIME has passed, the DEFRAG function will end.
nummins This is a decimal number (0-9999) that specifies the maximum number of minutes the DEFRAG function will run. A value of 0 is ignored.

- The elapsed time of the operation might be slightly longer than the MAXTIME specified because the MAXTIME is checked after each data set has been processed.
- If the CONSOLIDATE keyword is specified, MAXTIME will be ignored.

- DEBUG(TRACE):
  Specifies that DFSMSdss prints messages that indicate the relocated extents.
- VERSION1:
  Instructs DFSMSdss to execute Version 1, the pre-z/OS V1.10 version of DEFRAG:
  - This keyword does not support EAVs.
  - An error message is returned if specified for an EAV.

CONSOLIDATE command
CONSOLIDATE is a new command in z/OS V1.10 that performs both data set extent consolidation and extent reduction for data sets that occupy multiple extents on a single volume.

- It allows you to specify which data sets are to be processed.
- It tries to relocate multiple non-contiguous data set extents into contiguous space.
- Attempts to reduce the number of extents of a data set as much as possible even when the entire data set cannot be reduced to one extent.
- Supports the same type of data set filtering as COPY, DUMP, and RESTORE functions.

Note: This is the CONSOLIDATE function extracted from DEFRAG.

The CONSOLIDATE command will be restricted by the new STGADMIN.ADR.STGADMIN.CONSOLID RACF FACILITY-class profile. Access to this profile lets you perform a CONSOLIDATE operation without having READ access to the data sets that are moved.

See the manual DFSMSdss Storage Administration Reference, SC35-0423 for command syntax.

18.16 DFSMShsm

In this section we discuss how DFSMShsm handles space management, as well as important features such as the datamover.

Space management
Space management is performed using both the volume threshold (as it does today) and the track-managed threshold (new with DFSMS V1.10). Track-managed threshold was added to ensure that track-managed space gets space managed even when the overall volume has not exceeded the volume threshold value.

The track-managed threshold is set for SMS through the Storage Group and for non-SMS with the ADDVOL command. Table 18-1 shows the relationship between the track-managed threshold and the volume threshold.
Table 18-1  Space management

<table>
<thead>
<tr>
<th>Track-managed Threshold Exceeded</th>
<th>Volume Threshold Exceeded</th>
<th>Data Set Selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>YES</td>
<td>All Data Sets</td>
</tr>
<tr>
<td>NO</td>
<td>YES</td>
<td>All Data Sets</td>
</tr>
<tr>
<td>YES</td>
<td>NO</td>
<td>Only data sets with one or more of the first three extents in track-managed space</td>
</tr>
<tr>
<td>NO</td>
<td>NO</td>
<td>NONE</td>
</tr>
</tbody>
</table>

There are also considerations for DFSMShsm processing with Extended Address Volumes in the areas of ML1 Volumes, Recall and Recover and usage of Datamover.

**ML1 Volumes**
- Migration copies can only be written to track-managed space because they are non-VSAM data sets.
- SDSP data sets can be allocated in cylinder-managed space. If you choose, Small Data Set Packing (SDSP) data can be allocated in cylinder-managed space because they are VSAM, but note the threshold for ML1 volumes only applies to the track-managed space.

**Recall and Recover**
For non-SMS and non-VSAM data sets, DFSMShsm will use the track-managed free space for volume selection. For VSAM data sets, the entire volume free space is used.

**Datamover**
DFSMShsm has removed support for overriding DFSMSdss as the datamover. Any patch of the DMVST control block will fail.

### 18.17 VTOC index

The index record size for an EAV VTOC has increased to 8192 bytes versus 2048 bytes for non-EAV devices. This new value is recorded in the Format 1 DSCB for the index. This allows for scaling to the largest EAV size.

The VTOC index space map (VIXM) has a new bit, VIMXHDRV, to indicate that new fields exist in the new VIXM extension. The VIXM contains a new field, VIMXMAU for the “minimum allocation unit” (this is the multicylinder unit) in cylinders for the cylinder-managed space. Remember that each extent in the cylinder-managed space is a multiple of the multicylinder unit.

**Note:** If you omit the VTOC index size when formatting a volume with ICKDSF and do not preallocate the index, the default prior to DFSMS V1.10 has been 15 tracks. In DFSMS V1.10, the default size for all volumes (EAV and non-EAVs) will be calculated and might be different from prior releases.
18.18 VTOC access

Access to the VTOC records has been enhanced to allow a program to read the logical DSCB chain for a data set in one invocation of a service. OBTAIN and CAMLST have a new option to specify a number for the number of DSCBs you wish to read. CVAFDIR provides a new keyword to indicate to CVAFDIR processing to use the multiple buffers passed in the buffer list.

- **OBTAIN and CAMLST:**
  - NUMBERDSCB=number_dscbs
    - If number_dscbs > 0 and the DSCB is a Format 1 or Format 8 DSCB, then you need to read the logical chain of DSCBs that begin with that DSCB. They are chained with the DS1PTRDS field, which contains the CCHHR type of address of the next DSCB.
    - Consecutive 140-byte areas must be passed.

**Note:** Possible migration item if your program provides 101 bytes return area.

For SEARCH and SEEK requests, number_dscbs is an absolute expression with a value between 0 and 255 that designates the number of consecutive 140-byte return areas that are provided in wkarea_relexp. The system treats a value of 0 as a 1. Currently the system does not support a chain of more than 12 DSCBs for one data set but it is valid for you to provide an area that is longer than currently needed. The system verifies that the provided area is valid.

When you provide an area that is long enough to contain more than one DSCB, obtain processing will return DSCBs for the requested data set name in logical VTOC order until all the 140-byte return areas are used. The logical VTOC order is a format-1 DSCB, followed by zero or more format-3 DSCBs or a format-8 DSCB, followed by one or more format-9 DSCBs, followed by zero or more format-3 DSCBs. No absolute maximum number of DSCBs for a data set should be assumed.

For SEARCH requests, the actual number of DSCBs will be returned in a field located in the first 140-byte return area. For SEEK requests where the target of the seek operation is not a format-1 or format-8 DSCB, the NUMBERDSCB value will be treated as if it were 1 and only that single DSCB will be returned.

**Note:** For programs run on a level of the system prior to z/OS V1.10 that does not support this keyword, the NUMBERDSCB value will be treated as if it were 1.

- **CVAFDIR**
  - MULTIPLEDSCBS=YES

This specification indicates that the calling program requests to read/write multiple DSCBs to/from a buffer list that contains more than one buffer list entry.

This flag will resolve to a new indicator in the CVPL, CV4MULTD, to be set on.

Multiple DSCB processing for reads and writes is requested by specifying the MULTIPLEDSCBS=YES keyword and providing a buffer list that contains more than one buffer list entry (BFLHNOE>1).
MULTIPLEDSCBS=NO

This specification indicates that the calling program requests that only one DSCB should be processed. This is the default for MF=L and MF=I forms of the CVAFDIR macro.

When the MULTIPLEDSCBS keyword is not specified on the MF=E form, the existing setting of CV4MULTD is left unchanged.

When MULTIPLEDSCBS=NO is specified or defaulted, only the first available buffer list entry is processed.

Read considerations for MULTIPLEDSCBS

Only the first buffer list entry, seek or search, argument will be used. This will provide orientation to the data set from which the subsequent data set DSCBs will be read. As each one is read, the DSCB argument (BFLEARG) in each buffer list entry will be set in the format specified by the caller in each buffer list entry. The buffer list argument will be indicated as updated with the flag, BFLEAUPD, set on. Reading the data set DSCBs will be in the logical VTOC order and will continue as long as buffer list entries are available to return the DSCB.

A new field in the buffer list header (BFLHNOEN) will be set by CVAFDIR read processing to indicate the number of buffer list entries that are needed to read the entire set of DSCBs for the data set. This number is set in the header of the first buffer list and its setting is independent of the specification of the MULTIPLEDSCBS= keyword, the target volume type and whether the number of provided buffer list entries, BFLHNOE, is short or not.

The logical VTOC order would be either:
- Format-1, Format-3s
- Format-8, Format-9s, and Format-3s
- Buffer list entries other than the first must provide a 140-byte buffer. The first buffer list entry buffer size will follow the same rules as today. That is, with the seek option, provide either a 96-byte or 140-byte buffer and for the search option provide a 96-byte buffer. All other buffer list entry processing flags as described in the buffer list entry flag byte (BFLEFL) should continue to be supported. They include data in buffer modified, skip, I/O error, no key verify and argument format qualifiers. The DSCB argument (BFLEARG) returned in each buffer list entry is in the format determined by the argument format qualifiers (BFLECHR or BFLETTR) in each buffer list entry.

Write considerations for MULTIPLEDSCBS

The buffer list header must indicate the number of buffer list entries passed. Only buffer list entries without the skip flag on will be processed. The order in which the DSCBs are passed in the buffer list entries must correspond to the logical VTOC order as described previously. A new flag in the buffer list header can be set by the caller to indicate that the logical order in which the DSCBs appear in the buffer list must be written in reverse order.

For write processing, the first buffer list entry, like today, can be a 96-byte buffer if the DSCB to write is a Format 1. The same will hold true for a Format 8 DSCB. A 140-byte buffer can also be provided for these DSCBs as long as the BFLEARG points to the actual DSCB that needs to be written. Buffer list entries that do not describe Format 1 or 8 DSCBs must provide a 140-byte buffer and its buffer address (BFLEARG) must point to the actual DSCB that needs to be written. The caller must also specify the buffer list entry argument (BFLEARG) as a CCHHR for these buffer list entries. A new flag in the buffer list entry could be set for entries where a format 0 DSCB verify before a write is not needed (BFLENVER). This will override the VERIFY=YES setting.
18.19 LSPACE macro

The LSPACE macro returns information about a DASD volume. The returned info can be in character or binary format. You can use the LSPACE macro to get free space, volume fragmentation, and volume table of contents (VTOC) status information for a direct access storage device (DASD) volume. The LSPACE macro returns status information (such as LSPACE subfunction, return code, and reason code) in the parameter list. The LSPACE macro also returns the return code in register 15. For volumes that are configured with more than 9999 cylinders, you can use the EXPMSG option to create an expanded message return area that the LSPACE macro needs.

- New keywords XEXPMSG=addr and EXPDATA=addr
  
  New free space information returned includes statistics from track-managed space. For volumes that are configured with more than 65,520 cylinders, you can use the XEXPMSG option to create an extended expanded message return area that the LSPACE macro needs. The expanded data return area (EXPDATA) will return binary data of free space information for volumes with more than 65,520 cylinders. You can have LSPACE return additional information such as the Format 4 DSCB, the total number of free extents on the volume or the fragmentation index.

- A new DATATYPE keyword is available to limit the amount of data and thus processing time to return LSPACE information. Any combination of the new keywords can be specified. Details of the macro changes can be found in the manual DFSMS Advanced Services, SC26-7400.

The new LSPACE keywords are described next:

- XEXPMSG=addr or (reg) or 0
  
  Specifies the address of a caller-provided 95-byte extended expanded message return area into which LSPACE returns either a free space message or, for unsuccessful requests, status information. Specify this keyword if you wish to obtain free space information in the message return area for volumes that are configured with more than 65,520 cylinders. The returned free space will include space for the total volume and space from the track-managed space on a volume. For volumes with 65,520 cylinders or less both sets of free space information will be returned but they will be the same.

- EXPDATA=addr or (reg) or 0
  
  - addr specifies the address of a caller-provided expanded data return area into which LSPACE returns expanded free space and volume information.
  
  - reg specifies a register containing the address of the expanded data return area.

  Specify this keyword if you wish to obtain free space information in the LSPACE data return area for volumes that are configured with more than 65,520 cylinders. The returned free space will include space for the total volume and space from the track-managed space on a volume. For volumes with 65,520 cylinders or less both sets of free space information will be returned but they will be the same.

- DATATYPE= (VOLUME,VTOC,INDEX,FRAGINDEX,ALL)
  
  This keyword is only allowed when the DATA or EXPDATA keyword is specified. Only the information specified will be returned to the caller. DATATYPE is valid for both non-EAV and EAV. This keyword will eliminate unnecessary I/O required to retrieve free space information that is not be required by the caller.

  - VOLUME – Provide free space information for the volume
  
  - VTOC – Provide free space information related to the VTOC
  
  - INDEX – Provide free space information related to the VTOC index
  
  - FRAGINDEX – Provide the fragmentation index
– ALL – Provide all available LSPACE statistics. This is the default.

- PLISTVER= {plistver | IMPLIED_VERSION | MAX}

This keyword defines the version of the LSPACE parameter list that should be generated for the MF=L form of the LSPACE macro.

- When plistver is specified, the value is either 1 or 2 to specify the version of the LSPACE parameter list that should be generated. The PLISTVER= keyword is required for any macro keywords associated with version 2 or later to be specified. The macro keywords associated with each supported version of the macro are these:

  PLISTVER=1
  • DATA
  • EXPMSG
  • F4DSCB
  • MSG
  • SMF

  PLISTVER=2
  • XEXPMSG
  • EXPDATA
  • DATATYPE

When PLISTVER is omitted, the default is the lowest version of the parameter list, which is version 1.

- When IMPLIED_VERSION is specified, the generated parameter list is the lowest version which allows all of the parameters on the invocation to be processed.

- When MAX is specified, the generated parameter list is the largest size currently supported. This size can grow from release to release thus affecting the amount of storage needed by your program. If your program can tolerate this, IBM recommends that you always specify MAX when creating the list form parameter list as that will ensure that the list form parameter list is always long enough to hold whatever parameters might be specified on the execute form.

18.20 DEVTYPE macro

Use the DEVTYPE macro to obtain device characteristics information about I/O devices, including the number of cylinders.

- The DEVTYPE macro issued without the INFOLIST parameter returns a 2-byte value for the number of cylinders. The 2-byte field is mapped by DVACYL in mapping macro IHADVA.

- The DEVTYPE macro issued with the INFOLIST parameter (INFO=DASD) returns a different format of the device characteristics information. This includes a 4-byte value for the number of cylinders and is mapped to field DVAICYL in mapping macro IHADVA.

Note: For a volume larger than 65520 cylinders, DVACYL contains a dummy value of x'FFE' (65534). Use INFO=DASD to obtain an accurate number of cylinders in DVAICYL.
18.21 SMF records

- SMF Type 14 and 15: These records are written in the EOV and CLOSE functions for non-VSAM data sets.

  A new flag, SMF14EADSCB, indicates whether a program specified EADSCB=OK on the DCBE macro. You can map this record with either the IFASMFR macro or the IFGSMF14 macro:

  - Programs that open an EAV VTOC or a VSAM data set that has a format 8 DSCB must specify EADSCB=OK on the DCBE macro. This flag is a migration aid in z/OS V1.10 to find programs that open the VTOC or VSAM data sets with EXCP (MACRF=E on the DCB macro) and have not specified EADSCB=OK.

  - IBM recommends that you upgrade programs to handle 28-bit cylinder numbers and code EADSCB=OK. You might want to test the following fields to identify programs to upgrade.
    - If bit 0 of the SMFDCBMF field is 1 then the OPEN is for an EXCP DCB.
    - If all of the first 44 bytes of the SMFJFCB1 field contain x'04', then the OPEN was for a VTOC.

  A new flag, SMF14EXCPBAM, indicates the program used a non-EXCP OPEN DCB (BSAM, QSAM, BPAM) and issued EXCP or XDAP.

  - When SMF14EADSCB is off and SMF14EXCPBAM is on, it identifies programs that might need to be upgraded to handle 28-bit cylinder numbers and have EADSCB=OK coded.

  - This is to help you prepare for a future release when IBM might deliver support for non-VSAM data sets in cylinder managed space.

  **Note:** Use the TRKADDR macro when upgrading programs that manipulate cylinder numbers.

- SMF Type 19 - This record contains DASD volume information.

  This record includes LSPACE statistics which have been expanded with track-managed free space statistics in addition to the total volume statistics.

- SMF Type 60, 61, 64, 65, 66 - These records contain information about VVDS and VSAM components.

  These might contain extent descriptors with 28-bit cylinder addresses.

- SMF Type 74, subtype 1 - This record contains device activity statistics from RMF. It contains a new device capacity field.

18.22 VVDS

The VVDS contains extent descriptors for VSAM data sets. For data sets that are eligible to be allocated or extended on an EAV, its extent descriptors can contain 28-bit cylinder addresses. Today these extents are stored in an internal VVDS structure called a VVR. Some programs might access the VVDS even though no interfaces are provided for this purpose. It is very important that the programs that refer to the extents within the VVR be reviewed, and possibly modified, to ensure they can handle the 28-bit cylinder addresses.

The TRKADDR macro should be used whenever manipulating 28-bit cylinder addresses.
Chapter 19. Planning to use EAV

The purpose of EAV support is to allow the amount of addressable DASD storage per volume to grow beyond the current limit of 65520 cylinders by changing how tracks on ECKD™ volumes are addressed. This allows z/OS operating systems to have more capacity per UCB and consequently be able to reduce the number of UCBs needed, as some installations today are already constrained by the 64K device number limit.

Larger (and fewer) volumes can ease management for the Storage Administrator, including daily space management, backup, and setting up copy services relationships.

EAVs will be addressed with 28-bit cylinder, 4-bit track values compared to today’s volumes that are addressed with 16-bit cylinder, 16-bit track values. The first release of EAV will provide support up to 262668 cylinders.

Moving to EAV will require considerations as to software and hardware positioning.

In this chapter we describe suggested planning to be done in relation to software, hardware, and storage management disciplines before exploiting EAV.
19.1 Software requirements and compatibility

Software support for the EAVs is introduced with z/OS V1.10. Pre-z/OS V1.10 systems will not support bringing an EAV online. This will require all systems that share disks that will be EAV to be at the z/OS V1.10 level. Lower level systems will only recognize volume sizes of up to 65520 cylinders, while z/OS V1.10 systems recognize a volume size of up to 262668 cylinders. If you start working with EAV on one z/OS V1.10 system, be sure to put on the recommended maintenance on the lower level systems before starting. The lower level systems will accept an IODF definition of an EAV, but will not allow an EAV to be brought online. IBM recommends defining EAV addresses in the HCD as staying by default offline at IPL time on systems not supporting EAV.

As a default, usage of EAVs will not be fully enabled when IPLing a z/OS V1.10 system the first time. This is due to the SYS1.PARMLIB IDGSMSxx parameter USEEAV(YES | NO) (NO being the default). This leaves installations time to plan the implementation and usage of EAVs even having migrated to DFSMS V1.10. The USEEAV(NO) parameter allows the EAVs to come online but does not allow data set creations.

ICKDSF R17 has been updated (see APAR PK56092) to support EAV and support the extended addressing and the larger record length and block size of the VTOC index.

19.2 IBM System Storage DS8000 support

EAV support is provided for the IBM System Storage DS8000. The release 4.0 Licensed Internal Code update allows users to define a new 3390 Model A volume and enable EAV support. The IBM DS8000 Storage Manager and DSCLI have been updated accordingly. To support volume sizes greater than 65520 cylinders, volumes have to be defined as 3390 Model As in the DS8000. This new Model A can be configured to have from 1 to 262668 cylinders. The 3390 Model A is a 3390 type DASD with the same geometry as older devices. There are no HCD considerations. The Model A definition is only seen in the DS8000 Storage Manager or DSCLI.

Note: There is currently no planned support for EAV in the IBM System Storage DS6000™.

19.2.1 Dynamic Volume Expansion

The Dynamic Volume Expansion (DVE) feature on the DS8000 has been available since December 2007. DVE supports expanding any type of 3390 volume from 1 cylinder to 65520 cylinders. With the DS8000 R4 LIC, DVE can expand any 3390 volume to a maximum of 262668 cylinders.

The DS8000 does not allow expanding non-EAVs or 3390 Model A’s to more than 65520 cylinders while the volume is online to a pre-z/OS V1.10 system. After all systems sharing the volume are at the z/OS V1.10 level, DVE can upgrade any type of volume below the 65520 boundary (Model A or otherwise) to a Model A beyond the 65520 cylinder limit.

Note: Pre-z/OS V1.10 systems will not allow a Model A with more than 65520 cylinders to be varied online.
DVE functions can be invoked through the DS8000 Storage Manager as well as through DSCLI commands.

19.2.2 Copy services considerations

For volumes that are in a copy services relationship (FlashCopy, PPRC or XRC), use of DVE is not possible without withdrawing the relationship prior to expansion. This requires a full resynchronization of the copy services pair after expanding the primary and secondary volumes in the relationship. Keep in mind that the size of the secondary volume must be at least as large as the primary.

19.3 Supported data set types

The space on an EAV is split into a track-managed space (the first 65520 cylinders) and a cylinder-managed space (beyond the first 65520 cylinders). In z/OS V1.10, VSAM data sets of the following types are supported to be allocated in the cylinder-managed space:

- KSDS
- RRDS
- ESDS
- Linear

The following data set types are not supported in the cylinder-managed space of an EAV:

- Catalogs (BCS and VVDS)
- VTOC (continues to be restricted to stay within first 64K-1 tracks)
- VTOC index
- Page data sets
- VSAM data sets with Imbed or Keyrange attributes
- VSAM data sets with incompatible CA sizes (those with CA sizes of 2, 4, 6 or 8 tracks)

Only VSAM data sets with a compatible Control Area size (1, 3, 5, 7, 9 or 15 tracks) will be able to reside in the cylinder-managed space. This is because the CA size (the minimum allocation unit for a data set) must divide evenly into 315 tracks (or 21 cylinders), which is the volume MCU size.

- Non-VSAM data sets

If there is no space available in track-managed space, the allocation of these data sets will fail.

In the future, IBM intends to expand support for EAV with larger volume sizes and support for additional data set types and access methods.

19.4 Application migration

To prepare for EAV, it is important that programs that can access EAVs will recognize the changed addressing beyond the pre-EAV limit of 65520 cylinders, as well as new and modified output of certain system services.
19.4.1 z/OS tracking facility

The z/OS tracking facility (which is referred to as the EAV migration assistance tracker in some z/OS product manuals) is an important tool that can help you find programs that might need changes to support EAV. It can be run against non-EAVs to detect instances that might need to be changed if actually run on an EAV. The tracking facility helps you:

- Identify instances of programs that would definitely fail if they were to be run on an EAV. These instances are flagged as errors. The tracking facility flags programs that use the functions shown below, when the target volume of the operation is non-EAV and the function invoked did not specify the new EADSCB=OK keyword:
  - OBTAIN
  - CVAFDIR
  - CVAFSEQ
  - CVAFDSM
  - CVAFFILT
  - Open of VTOC
  - DCB OPEN of an EAS eligible data set

  This enables the system programmer to identify programs by job and program name, without failing the programs. Programs identified here will continue to fail if the system service is issued for an EAV, if they are not upgraded to specify the EADSCB=OK on the identified system service.

- Identify instances of programs that might have improper use of returned information. For example, parsing 28-bit cylinder numbers in output as 16-bit cylinder numbers. These instances are flagged as warnings. Programs that might generate warnings include:
  - IEHLIST LISTVTOC
  - IDCAMS LISTCAT
  - IDCAMS LISTDATA PINNED

- Identify instances of programs that might require analysis for changes to use new or modified services. These instances are flagged as informational. They are not considered errors because the services return valid information. Examples of programs that might be flagged as informational include:
  - LSPACE
  - DEVTYPE
  - DCOLLECT

Tracking facility commands

The tracking facility can be manipulated with the following commands:

- The SETCON operator command, which is used to activate and deactivate the tracking facility (SETCON TRACKING=ON|OFF)

- The DISPLAY OPDATA,TRACKING operator command, which is used to display the current status of the tracking facility, along with any recorded instances of violations.

- The CNIDTRxx dynamic PARMLIB member lists violations that have already been identified in order to prevent them from being recorded again. Activate the recent changes using the SET CNIDTR=xx command. See tracking facility in z/OS MVS Planning: Operations, SA22-7601 for more information.

The tracking facility must run on a z/OS V1.10 system to collect data for EAV.
Figure 19-1 shows an example of a display of trace data collected by the tracking facility.

<table>
<thead>
<tr>
<th>D OPDATA,TRACKING</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNZI001I 11.08.01 TRACKING DISPLAY 160</td>
</tr>
<tr>
<td>STATUS=ON NUM=92 MAX=1000 MEM=00 EXCL=0 REJECT=0</td>
</tr>
<tr>
<td>----TRACKING INFORMATION---- -VALUE-- JOBNAME  PROGNAME+OFF-- ASID NUM</td>
</tr>
<tr>
<td>SMS-I:3 DEVTYPE 03 <em>MASTER</em> ICHSEC06 1DEF8 01 2</td>
</tr>
<tr>
<td>SMS-I:3 DEVTYPE 03 BOBH ISPICP 80 6A 1</td>
</tr>
<tr>
<td>SMS-I:3 DEVTYPE 03 BOBH ISRUDA 3834C 6A 3</td>
</tr>
<tr>
<td>SMS-I:3 DEVTYPE 03 DFHSM70 ARCCtl 227FC4 69 2</td>
</tr>
<tr>
<td>SMS-I:3 DEVTYPE 03 DFRMM EDGPARM 258 5F 1</td>
</tr>
<tr>
<td>SMS-I:3 DEVTYPE 03 DFRMM EDGPDO 108 5F 1</td>
</tr>
<tr>
<td>SMS-I:3 LSPACE EXPMSG= 803 MHLRES1 ISRUDA 3765E 6F 5</td>
</tr>
<tr>
<td>SMS-I:3 LSPACE EXPMSG= 803 MHLRES3 ISRUDA 3765E 6D 7</td>
</tr>
<tr>
<td>SMS-I:3 LSPACE EXPMSG= 803 MHLRES5 ISRUDA 3765E 6A 1</td>
</tr>
<tr>
<td>SMS-I:3 LSPACE EXPMSG= 803 MHLRES5 ISRUDA 3765E 72 6</td>
</tr>
<tr>
<td>SMS-W:2 IDCAMS LISTCAT 02 MHLRES5K IDCAMS E866 1D 1</td>
</tr>
</tbody>
</table>

TO REPORT THESE INSTANCES, SEND THIS MESSAGE VIA E-MAIL TO CONSOLESUS.IBM.COM. FOR ADDITIONAL INFORMATION OR TO OBTAIN A CURRENT EXCLUSION LIST, SEE APAR II13752.

Figure 19-1  D OPDATA,TRACKING display

Information conventions
The information returned by the tracking facility describes the occurrence of an instance. The tracking facility returns tracking information and a tracking value. The tracking information can be from 1 to 28 characters in length and the system can set any EBCDIC value. The tracking value is four bytes of binary data associated with this instance. These values include data to associate an instance to a specific DFSMS function and to define the reason for the instance being recorded. This convention allows for maximum flexibility in defining exclusion records that apply to DFSMS records. The conventions for the tracking information and tracking value for a DFSMS instance follow.

Tracking information
Following is an example of the tracking information for a DFSMS instance:

`‘SMS-I:3 LSPACE MSG= ’`

It can be broken down into several parts:

1. The first portion of the tracking information will be set to “SMS-” to identify this as a DFSMS instance.
2. Appended to this is an error category, of “E” for error, “W” for warning, or “I” for informational, followed by a colon.
3. Appended to the colon is a numeric value that identifies the reason for the recorded instance. These values are:
   - 1 - EAV migration: EADSCB=OK keyword was not specified on an invoking program where the target volume was non-EAV. The invoking program would fail if the target volume was an EAV. The following section of the tracking information indicates the error that would have occurred. This instance is recorded in the tracker as an error message.
2 - EAV migration. Formatted output display can contain 28-bit cylinder numbers. Program usage of these track addresses might need to be changed. Use the TRKADDR macro or the IECTRKAD routine for comparison and manipulation of 28-bit cylinder numbers. This instance is recorded in the tracker as a warning message.

3 - EAV migration. The new function is available to the invoking program. The identified program might want to exploit the available new function. This instance is recorded in the tracker as an informational message.

4. The remaining tracking information describes the function executing when the tracker recorded the instance.

In the example above, the ‘SMS-I:3 LSPACE MSG= ’ tracking information describes a DFSMS instance as an informational instance where new function is available that the invoking program might want to exploit. The function running when the tracker recorded this instance was LSPACE MSG=.

### Tracking value

For DFSMS instances, the tracking facility sets the tracking value as follows:

- The system sets the low order byte of the tracking value to the same numeric value that identifies the reason for the instance.
- The remaining high order 3-bytes are left for the function recording the tracked instance. These 3-bytes are optional. For example, these values could be set to return and reason codes or parameter list flags.

### Exclusion list

For DFSMS, these values include data to associate an instance to a specific DFSMS function and to define the reason for the instance being recorded. This standard allows for maximum flexibility in defining exclusion records that apply to DFSMS records. This allows you to exclude programs that are not yet ready for evaluation or that have already been evaluated as requiring no changes.

IBM provides an exclusion list to prevent common instances from being recorded. It is filtered based on the following information:

- Tracking Information Mask
- Job Name Mask
- Program Name Mask

Download the most recent exclusion list to your SYS1.PARMLIB(CNIDTRxx) member from the web-site:


Look for z/OS consoles 1-byte tracker exclusion list.

### Planning activity recommendations

In preparation for EAV, you might want to look at the following programs:

- Programs that calculate volume or data set size by any means, including reading a VTOC or VTOC index directly with a BSAM or EXCP DCB. The system cannot distinguish between programs that read DSCBs for space information and those that read them for metadata. Both types of program require attention.
- EXCP, XDAP and STARTIO macros for DASD channel programs and other programs that examine DASD channel programs or track addresses.
Chapter 19. Planning to use EAV

19.4.2 TRKADDR macro

Use the new TRKADDR macro for all track address comparisons and calculations. Programs should not need to do 28-bit manipulation themselves. The TRKADDR macro should be used for all track address computations, even those not directly affected by this support for VSAM data sets using EAS. IBM suggests that you use TRKADDR macro even for programs that do not access EAV to be prepared for possible future enhancements. It works on all non-EAVs.

For programs that are written in a high level language such as C, C++, COBOL, or PL/I, a new IECTRKAD routine can be called to execute the same functions as in the TRKADDR macro.

19.5 VTOC changes

The extended addressing means changes to the VTOC. In an EAV the VTOC is still a single extent. The restriction of the VTOC residing in the first 64K-1 tracks of the volume still applies. The volume label record on track 0, record 3 and UCBVTOC (TTR0) continues to identify the location of the start of the VTOC. The VTOC record size and format with keys do not change with z/OS V1.10.

New Format 8 and Format 9 DSCBs are introduced to address the cylinder managed area. A Format 8 DSCB is equivalent to a Format 1 DSCB. The Format 8 DSCB is created when a data set is created in the cylinder-managed space, or when an existing data set below the 65520 cylinder limit extends into the cylinder-managed space (Format 1 DSCB is converted to a Format 8 DSCB). The Format 8 DSCB will point to a Format 9 DSCB (additional attribute data and possible pointer to Format 3 DSCB).

On each EAV the VTOC index record size is 8192 instead of 2048 bytes. This is either as a result of a volume initialization or by a reformat due to Dynamic Volume Expansion (DVE). You will see the following message in the joblog due to DVE:

IEA019I DC65,xxxxxx,VOLUME CAPACITY CHANGE,OLD=0000F230,NEW=00010083

In this case a 3390 type of volume expands from below 65520 cylinders to become an EAV (3390 Model A). To address the added capacity, the VTOC index has to be reformatted. The REFORMAT command results in an index record size of 8192 instead of 2048 bytes for a non-EAV.

19.5.1 VTOC size considerations

The expansion of a volume might need focus on the current VTOC size. If you increase the size of a volume, consider the current VTOC and index size. Each track in the VTOC contains 50 DSCBs. You need at least one DSCB per data set for the data sets that can be track-managed data sets. Each cylinder managed data set needs two DSCBS. You can let the system define the size of the VTOC index automatically by not specifying the index when initializing your volume. You can choose to override this size to provide for a larger or smaller index.
**REFORMAT command**

You can expand both the VTOC and the VTOC index if needed with the REFORMAT command as shown in Figure 19-2.

```plaintext
//*
//VTOCEXT EXEC PGM=ICKDSF
//*
//SYSPRINT DD SYSOUT=*  
//vvvvv DD DISP=SHR, VOL=SER=vvvvv, UNIT=3390  
//SYSLIN DD  
//REFORMAT DDNAME(vvvvvv) VERIFY(vvvvvv) EXTVTOC(1415) EXTINDEX(240)  
//  
```

*Figure 19-2  ICKDSF REFORMAT of EAV VTOC*

Specify in the EXTVTOC parameter the total number of tracks you want your VTOC to grow into. For the EXTINDEX parameter, specify the target size of the extended index data set.

**Note:** Expanding the VTOC is limited by the number of free tracks on the volume following the current location of the VTOC. If there is no free space following the VTOC, move the data sets that will prevent you from expanding the VTOC and then do the expand. You can allocate dummy data sets that are adjacent to the VTOC to allow for future expansion or you can over allocate the VTOC size to avoid running out of VTOC space, avoiding the need to expand the VTOC.

Alternatively you can move the VTOC to a location on the volume with sufficient free space (in the first 65535 tracks) with the NEWVTOC parameter on the REFORMAT command. This requires that the volume be varied offline to all systems. If you are not moving the VTOC the volume can remain online.

Ensure that EAVs have an active index and that system management procedures are in place to detect and rebuild an index if it gets disabled. Non-indexed EAVs with many data sets will perform poorly. For further information about VTOC and VTOC index sizes, refer to *ICKDSF User’s Guide And Reference*, GC35-0033.

### 19.5.2 LSPACE macro changes

The following LSPACE macro changes are introduced with OA22449:

- (1) Pre-z/OS V1.10 systems are to accept the extended parameter list and will not fail it with RC12.
- (2) Pre-z/OS V1.10 systems are to treat EXPDATA= or XEXPMSG= requests as DATA= or EXPMSG= respectively.

### 19.5.3 VVDS size

The VVDS is still limited to be allocated in the track-managed space regardless of whether it is defined implicitly or explicitly. An implicitly defined VVDS is allocated with a default size of (10, 10) tracks. You can change this default dynamically with a new z/OS V1.10 modify command to the Catalog Address Space:

```plaintext
F CATALOG, VVDSSPACE (primary, secondary)
```
The modify changes the default primary and secondary allocation amount in tracks for an implicitly defined VVDS. The specified values are preserved across a Catalog Address Space restart, but are not preserved across an IPL. There is no PARMLIB option for these values.

The VVDS size must be taken into consideration when migrating to EAVs. An extensive number of data sets on an EAV volume can exhaust the VVDS, even if its current allocation seems adequate. Changing a VVDS is a burdensome task, as the data sets in the VVDS have to be off-loaded from the volume to redefine the VVDS at a larger size. With DVE you can go from a 3390 Model 3 to a very large volume, so you should analyze your current VVDS sizes and their average usage. In addition, you can consider creating your new volumes with a larger VVDS size.

IBM provides a product that can expand an exhausted VVDS; IBM Tivoli Advanced Catalog Management for z/OS. Contact your IBM representative for more information about this product.

19.5.4 Extent descriptors

The VVDS contains extent descriptors for VSAM data sets. For data sets that are eligible to be allocated or extended on an EAV, its extent descriptors can contain 28-bit cylinder addresses. Today these extents are stored in an internal VVDS structure. Some application and vendor programs can access these fields, even though no interfaces are provided for this purpose. It is very important that any application and vendor programs which refer to the extents within the catalog record be reviewed, and possibly modified, to ensure that they can handle the 28-bit cylinder addresses.

19.5.5 VTOC/VVDS EAV creation scenario

In our ITSO environment, we created an EAV of 75674 cylinders. We created a VTOC of 650 tracks and let the system calculate a default VTOC index, which turned out to be 38 tracks. The VVDS was the default size (10,10). The volume was filled entirely by 10077 21-cylinder data sets (351 sequential data sets, 3242 VSAM cluster entries, 3242 VSAM data entries and 3242 VSAM index entries.

The VVDS went into 8 extents (1 primary and 7 secondary = 80 tracks). We validated this number using the calculation found in the section titled Estimating Space Requirements for the VVDS in DFSMS Managing Catalogs, SC26-7409.

19.6 Allocation considerations

The volume size on a DS8000 is a flexible unit and can now expand even further. You will be able to adjust the volume size according to your need for data below and above the 65520 cylinder limit on the EAV. Most probably you will not start out using the maximum volume size, as this can be expanded dynamically later on.

The selected volume size and new limits like the high and low thresholds on both the total volume as on the track-managed space will affect SMS volume selection and HSM Daily space management. The impact of these important thresholds are discussed later on in this chapter, but can be illustrated graphically as shown in Figure 19-3.
Allocations in the track-managed space on an EAV will happen as in pre-z/OS V1.10 systems. The cylinder-managed space is different. Mapping of the cylinder-managed space happens in MCUs. An MCU is a 21 cylinder unit. This means that all requested allocations in the cylinder-managed space will be rounded up to the nearest increment of 21 cylinders. So if the track-managed request was 30 cylinders and the data set is a candidate for the cylinder-managed space, the allocation will be rounded up to the nearest increment of 21 cylinders (42 cylinders in this case). This might end up in using slightly more capacity for some allocations. Consider changing current allocation to be increments of 21 cylinders on a balanced primary and secondary allocation request. If you have allocation products supporting your data set extent process (for example like the IBM Tivoli Allocation Optimizer), you can adjust these procedures to work on a 21 cylinder boundary regardless of the allocation happening in the track-managed or in the cylinder-managed space (will also give you a more reusable fragmentation of your DASD).

DFSMShsm recalls will be based on the MCU when the recall goes to EAS. This will also be the case when DSS does a restore. In total, this rounding up of space is expected to have a modest cost due to increased allocation at the MCU boundary, as the biggest data sets are also expected to be in the cylinder-managed area.

The supported VSAM data sets will be allocated in the track-managed or the cylinder-managed space based on BPV and the amount of free capacity in each. The BPV is defined in the IGDSMSxx member in SYS1.PARMLIB and can be defined on the individual storage group as well. If no value is set, the default is 10 cylinders, meaning that for all supported data set space requests (primary or secondary) equal to or larger than 10 cylinders, the allocation preference will be the cylinder-managed space. A value of 0 means that all allocations will have a preference for the track-managed space. A value of 65520...
means that supported data sets will have a preference for the cylinder-managed space. When the BVP value is set on a storage group, this value will be used in preference to the IGDSMSxx setting or a value set dynamically by a SETSMS command.

**Note:** BPV is a preference. If there is insufficient free space in the preferred area, the allocation will be where the requested free space is available. The system can split a single allocation into both areas of the volume.

### 19.6.1 Volume size considerations

The operating system does not generally require DASD volumes to be the same size but storage administrators typically try to standardize on one or two sizes of volume. The reasons include these:

- It allows advanced copy services to work.
- It simplifies moving volumes around.
- Thinking of your disk capacity in terms of how many volumes you have simplifies storage estimation.
- When you decide to define volumes that are larger, it is most convenient if the new volume size is a multiple of the old volume size or a little more than a multiple. This simplifies the process of merging existing volumes to larger volumes. If you do not merge volumes, then this is unimportant.

Before the IBM 3390 direct access storage device (DASD) was emulated on IBM ESS (Enterprise Storage Subsystem), the 3390 had these models:

- Model 1 with 1113 cylinders, which is about 1 GB, where 1 GB is 1,000,000,000 bytes.
- Model 2 with 2226 cylinders, which is about 2 GB.
- Model 3 with 3339 cylinders, which is about 3 GB.
- Model 9 with 10,017 cylinders, which is about 9 GB (actually about 8.5 GB)

Since the availability of storage subsystems such as ESS and the IBM DS8000, storage administrators often define volumes of the following sizes even though many other sizes are valid:

- 30,051 cylinders, which is about 27 GB (actually about 25.5 GB)
- 60,102 cylinders, which is about 51 GB.
- 65,520 cylinders, which is about 55.7 GB. This is the maximum size allowed before z/OS V1.10. This maximum often is approximated as 54 GB.

When planning your EAV capacity, you can choose to continue the policy of multiplying the old volume capacity. This simplifies planning for growth and replication management. For example, nine 3390 devices with 3339 cylinders (3390 Model 3) could be consolidated on one 3390 device with 30,051 cylinders, or three 3390 devices with 10,017 cylinders could fit on one 3390 device with 30,051 cylinders (3390 Model 9). If your previous planning reflected this approach, you might want to consider a maximum EAV size of 240,408 cylinders, which would be the equivalent of eight 3390 devices with 30,051 cylinder volumes or twenty-four 3390 devices of 10,017 cylinders.

**Note:** When defining an EAV, either through the DSCLI or the DS8000 Storage Manager, the volume size is automatically rounded to the next multiple of 1113 cylinders.
Migration considerations
After you have decided on your volume size, you should consider how to get to this new environment. For example, here are two possible philosophies you can use:

- Allow existing data sets to expire and create new data sets on EAVs.
- Actively move data sets to the new environment. You can use tools such as DSS and TDMF® and you can use the HMIGRATE and HRECALL commands.

Using DSS and TDMF to move will require target volumes to be of the same size (or larger) as the source volume. As an example, you can choose to have 50 base volumes of different sizes (3390-3, 3390-9, 27 GB, and 54 GB). You will also have to consider a new EAV size that will be appropriate for your installation. If you are using PAVs (or HyperPAVs), you must consider how many addresses are available.

19.6.2 SMS storage group free space considerations

Having a storage group with potentially a mix of EAV and non-EAVs and the free space spread over the track-managed and the cylinder-managed space, you should plan how to manage this change compared to the procedures you are using today. You can have procedures monitoring your current free space to determine how it meets the requirements of your installation. Some installation have even automated this process adding free space capacity, when the free extents get too small or the total free space capacity in the storage group falls below a given threshold. With the introduction of EAV, you will have to ensure that there is sufficient free space in both the track-managed and the cylinder-managed space to meet the needs in your environment. Thus, you might want to start reporting on both areas.

By adjusting your procedures to handle free space split on the track-managed and the cylinder-managed spaces, there should be no problem having a storage group with a mix of EAV and non-EAVs.

**Note:** Space release is done on the MCU boundary. Partial release can release a portion or none of unused space. One single extent can not span both track-managed and cylinder-managed storage.

19.6.3 SMS volume selection

SMS volume selection has changed to meet the BPV distinction between the track-managed and the cylinder-managed space and to honor both the entire volume (high) threshold and the track-managed (high) threshold.

In the case of a cylinder-managed preferred allocation for an EAS-eligible data set, a selected volume must contain the allocation below the high threshold for the entire volume to still be regarded as a Primary Volume.

In the case of a track-managed preferred/enforced allocation, a selected volume must contain the allocation below the track-managed high threshold to be regarded as a Primary Volume.

In terms of SMS volume selection, the Primary Threshold can be either the entire volume (high) threshold or the track-managed (high) threshold based on the allocation request, amount of free space and the BPV.

The Secondary Threshold is defined as the less important threshold. This threshold is always the entire volume threshold.
Primary volumes must satisfy **all** volume preference attributes (meets Primary Threshold, Secondary Threshold, Enabled, EAV, Accessibility, PAV and Availability). Volumes that do not meet all volume preference attributes are secondary volumes. Primary volumes must have sufficient space below the threshold in the target area to hold the allocation (meeting the Primary Threshold) and sufficient space below the threshold for the entire volume to hold the allocation (meeting the Secondary Threshold).

**If requested space is >= BPV**, the preferred area is the cylinder-managed space. EAVs meet the EAV preference attribute, while non-EAVs of course will fail this. Therefore, EAVs can be Primary Volumes, if they meet all volume preference attributes, while non-EAVs can not be Primary Volumes.

When some Primary Volumes are selected, SMS will next call SRM (System Resource Manager) to select a volume. When there are no Primary Volumes (this can happen if none of the EAVs meet all volume preference attributes), SMS will randomly select a volume from the highest ranked volumes. Because EAV meets the EAV preference attribute, EAVs will be ranked higher than the non-EAVs, assuming all other volume preference attributes are the same.

**If requested space is < BPV**, the preferred area is the track-managed space. Both EAV and non-EAV meet the EAV preference attribute and will be equally preferred for selection. Again EAV and non-EAV must meet all volume preference attributes to become Primary Volumes. In terms of threshold attributes on track-managed space allocations, Primary Volumes must have sufficient track-managed space below the (track-managed) threshold to hold the allocation (i.e. meeting the Primary Threshold) and have sufficient space below the threshold for the entire volume to hold the allocation (i.e. meeting the Secondary Threshold). SMS will call SRM to select a volume from the Primary Volume list. If there are no Primary Volumes, SMS will select a volume from the highest rank.

SMS Volume Selection has also been adjusted in regard to VSAM striped data sets. All stripes in a striped data set have to be of equal size. Catalog will be called in the SMS Volume Selection process to ensure that the extents are all the same size.

Figure 19-4 is a sample ISMF POOL STORAGE GROUP ALTER panel. It shows how the thresholds are managed at the storage group level. *Allocation/migration Threshold* refers to thresholds for the entire volume. *Alloc/Migr Threshold track-managed* refers to thresholds for the track-managed space of an EAV. The BreakPointValue can be set individually by storage group in this panel. The value is in cylinders.
19.7 Analyzing your environment before implementing EAV

DCOLLECT is the tool often used as a base for reporting and automation disciplines as it reports widely on the catalog and the DASD environment.

The DCOLLECT Volume Record (V) has been changed to report on the track-managed space of an EAV. This information about the track-managed space is in addition to the information for the entire volume. The DCOLLECT volume record also stores information about the type of volume (EAV or non-EAV).

The cylinder-managed space size has to be calculated by subtracting the track-managed figures from the total volume figures. Also, the storage group record (SG) in DCOLLECT has been changed to report the track-managed capacity, the thresholds for the track-managed space and the BPV.

If you are currently using DCOLLECT, you should refer to DFSMS Access Method Services for Catalogs, SC26-7394 for changes in the record layout from the DCOLLECT program.

DCOLLECT is a valuable tool for evaluating your current data set allocation sizes (primary and secondary) and to evaluate the sizes of your volumes. This information will help you to decide what size volumes to define to start out with. It will also help you determine a reasonable BPV for your environment. The system uses the BPV to help optimize data set extent placement when deciding whether to use the track-managed or cylinder-managed space.
19.8 Defragmentation of EAVs

If you currently use the DEFRAG function, you should look into the changes introduced in DFSMS V1.10 to position for EAV.

The DEFRAG function consolidates the free space on a volume to decrease the chance of out-of-space conditions on new allocations. DEFRAG accomplishes this by relocating data set extents on a DASD volume to reduce or eliminate free space fragmentation, and prints a report on free space and other volume statistics.

19.8.1 DEFRAG Version 1

In the current version of DEFRAG, Version 1, data set extents can be combined as a result of specifying the optional CONSOLIDATE keyword. In Version 1, if a free space extent large enough to contain the entire data set is not available, no consolidation is performed. This design does not allow for partial extent reduction or for the user to specify which data sets require consolidation, as DEFRAG is performed at the volume level.

The DEFRAG function has been changed to support EAVs and to allow more flexibility.

The CONSOLIDATE keyword has been replaced by a new CONSOLIDATE command, and Version 2 DEFRAG will use 28-bit cylinder addressing to support volumes larger than 65520 cylinders.

19.8.2 DEFRAG Version 2

With the new DEFRAG Version 2, DFSMSdss will obsolete the CONSOLIDATE keyword. If CONSOLIDATE is specified, the user will receive an informational message indicating that the keyword is no longer supported. DEFRAG will then run the new CONSOLIDATE command against all eligible data sets after EXCLUDE and BY filtering. After CONSOLIDATE processing completes, DEFRAG Version 2, will run normally. The new CONSOLIDATE command, will try to reduce the number of extents of a data set as much as possible, even when the entire data set cannot be reduced to one extent. Using CONSOLIDATE also enables you to selectively specify the data sets you want CONSOLIDATE to focus on. Figure 19-5 is an example of the CONSOLIDATE command.

```
CONSOLIDATE -
DATASET(INCLUDE(**) -
EXCLUDE(USER2.**.LIST,.LOAD)) -
PHYSINDYNAM(VV VVV)
```

*Figure 19-5  CONSOLIDATE command example*

**VERSION 1 keyword**

The new keyword VERSION1 is used to invoke DEFRAG Version 1 instead of Version 2.

If VERSION1 is specified for a volume that is larger than 65520 cylinders, the command will fail with message ADR152E indicating that this keyword does not support large volumes.

If VERSION1 is specified along with CONSOLIDATE, DEFRAG Version 1 will execute with the CONSOLIDATE keyword, not the new CONSOLID command. DEFRAG Version 1 will not execute the new CONSOLID command.
**MMOVPC** keyword

MMOVPC (n,p,d) is a new keyword on the DEFRAG function.

DEFRAG processing attempts to move a fraction of the free tracks during each pass. It calculates an integer limit that is the larger of \( n\% \) of tracks divided by \( p \) (the number of passes) 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 based on the value specified for \( d \). Between each pass, the DEFRAG function releases and re-obtains volume serialization. This action can reduce the overall time, in which a DEFRAG operation makes a volume unavailable to other applications.

**MAXTIME** keyword

The MAXTIME keyword has also been added to DEFRAG:

MAXTIME specifies the maximum time, in minutes, for the DEFRAG operation to complete. MAXTIME is checked after each data set is processed. When the MAXTIME value is reached, the DEFRAG operation quiesces all current activity and ends.

---

**Note:** Data set extents will not be moved between the track-managed space and cylinder-managed space of an EAV during DEFRAG processing.

**Recommendations for DEFRAG**

Following are considerations when using DEFRAG on EAVs:

- Use MMOVPC, rather than MAXMOVE, when running DEFRAG on an EAV. The MMOVPC value applies separately to the track-managed space and the cylinder-managed space.
- MMOVPC and MAXMOVE are mutually exclusive.
- MMOVPC and VERSION1 are mutually exclusive.

If you specify MMOVPC, only the first method is used during each pass.

**19.8.3 RACF considerations**

A new STGADMIN profile called STGADMIN.ADR.CONSOLID has been added to protect the consolidate function. This profile allows you to perform a CONSOLIDATE operation without having READ access to the data sets that are moved.
19.9 DFSMShsm space management

DFSMShsm V1.10 has been changed to support EAVs in relation to thresholds on both the track-managed space view as well as the total space view of an EAV. DFSMShsm uses these thresholds as follows:

- DFSMShsm performs space management using both the volume thresholds (an existing process) and the track-managed thresholds (new with z/OS V1.10).
- DFSMShsm obtains the track-managed threshold in one of two ways.
  - For SMS EAVs, DFSMShsm will use the newly defined track-managed low and high threshold values associated with each storage group.
  - For non-SMS EAVs you can specify the TRACKMANAGEDTHRESHOLD keyword on the ADDVOL command as shown in Figure 19-6.

```
ADDVOL MLDE65 PRIMARY UNIT(3390) TRACKMANAGEDTHRESHOLD(80 40) THRESHOLD(85 60)
```

Figure 19-6  ADDVOL command with TRACKMANAGEDTHRESHOLD parameter

Use this keyword to specify high and low thresholds for the track-managed space of a non-SMS EAV Level 0 volume. If TMT is not specified, the corresponding values for THRESHOLD will be used. Refer to the z/OS DFSMS Storage Administration Reference, SC26-7402 for detailed information.

The purpose of the track-managed threshold is to ensure that the track-managed space gets managed separately, even if the overall volume has not exceeded the volume threshold value.

**Track-managed threshold**

Managing an entire EAV with one threshold is not sufficient because average data set size in the cylinder-managed space will likely exceed the average data set size in track-managed space. This will result in data sets allocated in the cylinder-managed space being migrated at a higher rate than data sets allocated in the track-managed space, potentially causing the track-managed space to remain full, while the overall volume utilization is low.

To prevent this, a unique threshold for the track-managed space (track-managed threshold) is used to manage the free space separate from the free space on the entire volume (volume-level threshold). DFSMShsm examines both volume level thresholds and track-managed thresholds for EAV L0 volumes. If either threshold has been exceeded then migration eligibility will be performed for data on the volume.

If track-managed threshold values are not specified, the default is to use the volume-level threshold values. Data set eligibility will be further qualified by the location of the first three extents of the data set. If any of the first three extents are allocated in track-managed space, the data set will be eligible to be processed, when the track-managed space or entire volume space is managed. If none of the first three extents are allocated in track-managed space, the data set will be eligible to be processed, when the entire volume space is being managed.

Figure 19-7 shows the criteria used by the daily space management task.
Space management (Interval Migration and Primary Space Management) will be considered complete after DFSMShsm determines both thresholds have been met. This applies only to hourly (Interval Migration) space management of non-SMS L0 EAVs and when the AUTO MIGRATE attribute is not equal to I (Immediate) for SMS-managed EAVs (see Figure 19-8). The AUTO MIGRATE attribute is in the ISMF Pool Storage Group Panel.

<table>
<thead>
<tr>
<th>Track-Managed Threshold Exceeded</th>
<th>Volume-Threshold Exceeded</th>
<th>Data Set Selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>All data sets</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>All data sets</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Only data sets with 1 or more of the first 3 extents in track-managed space</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>None</td>
</tr>
</tbody>
</table>

**Figure 19-7** Daily space management thresholds

Hourly space management (Interval Migration) of SMS-managed EAVs having the AUTO MIGRATE=I attribute specified in the Storage Group will be performed if either the midpoint between the high and low volume threshold values is met or if the midpoint between the high and low track-managed threshold values is reached or exceeded.
Daily space management (Primary Space Management) will be performed regardless of the high volume threshold and high track-managed threshold values.

19.10 Copy Services considerations

Copy Services support for EAVs is the same as for non-EAVs. Copy Services does not allow Dynamic Volume Expansion to occur without withdrawing the Copy Services relationship, which will mean a full resynchronization of the relationship afterwards.

A new Global Mirror keyword, XTNTLSTX, has been added to the FCESTABL command and the FCWITHDR to support the new extended addressing volume (EAV). XTNTLSTX specifies a list of extent sets that are in individual FlashCopy relationships. You can specify as many extent sets as allowed by the TSO command line limitation.

- You can use XTNTLSTX only if FlashCopy V2 is enabled on the storage control.
- You must use XTNTLSTX if either or both the source and target volumes are larger than 65520 cylinders (EAV).
- XTNTLSTX is not supported when OPENDVCS(YES), REMOTE(YES), or INCREMENTAL(YES) is specified.

19.11 SMF changes

SMF has been enhanced to support EAV. In this section we provide a summary of the enhancements.

19.11.1 SMF Type 14 and 15

These records are written in the EOV and CLOSE functions for non-VSAM data sets.

A new flag, SMF14EADSCB, indicates whether a program specified EADSCB=OK on the DCBE macro. You can map this record with either the IFASMFR macro or the IFGSMF14 macro.

- Programs that open an EAV VTOC or a VSAM data set that has a format 8 DSCB must specify EADSCB=OK on the DCBE macro. This flag is a migration aid in z/OS V1.10 to find programs that open the VTOC or VSAM data sets with EXCP (MACRF=E on the DCB macro) and have not specified EADSCB=OK.
- IBM recommends that you upgrade programs to handle 28-bit cylinder numbers and code EADSCB=OK. You might want to test the following fields to identify programs to upgrade.
  - If bit 0 of the SMFDCBMF field is 1, then the OPEN is for an EXCP DCB.
  - If all of the first 44 bytes of the SMFJFCB1 field contain x'04', then the OPEN was for a VTOC.

A new flag, SMF14EXCPBAM, indicates the program used a non-EXCP OPEN DCB (BSAM, QSAM, BPAM) and issued EXCP or XDAP.

- When SMF14EADSCB is off and SMF14EXCPBAM is on, it identifies programs that might need to be upgraded to handle 28-bit cylinder numbers and have EADSCB=OK coded.
- This is to help you prepare for a future release when IBM might deliver support for non-VSAM data sets in cylinder managed space.
19.11.2 SMF Type 19

This record contains DASD volume information. It includes LSPACE statistics which have been expanded with track-managed free space statistics in addition to the total volume statistics.

19.11.3 SMF Type 60, 61, 64, 65, 66

These records contain information about VVDS and VSAM components. They can contain extent descriptors with 28-bit cylinder addresses.

19.11.4 SMF Type 74, subtype 1

This record contains device activity statistics from RMF. It contains a new device capacity field.

19.12 DFSMSdss considerations

A full volume restore from a full volume dump of an EAV will not be successful on a pre-z/OS V1.10 system. Existing ADR309E message will be issued if that is attempted. A track restore request that includes track 0 of a dump from an EAV to a non-EAV will fail with message ADR389E.

Logical and physical data set restore (as well as a recall) to a pre-z/OS V1.10 system will restore all extents of a data set dumped from an EAV and will create a Format 1 DSCB instead of restoring the F8 and F9 DSCBs that are in the dump or migration copy.

Note: Ensure that the target of the restore or recall can accommodate all the data. The recall or restore will continue until all the data sets are moved or the target volumes have run out of space.

19.13 DB2 considerations

APAR PK58292 provides support for DB2 Admin Enablement and DB2 Log Manager with respect to EAV.

As long as you keep BSDS and active logs on non-EAV volumes and are not using Admin Enablement functions, you should be able to run DB2 V8 and V9 without any new DB2 EAV support code. Apply the appropriate coexistence PTFs before placing any BSDS or active logs on EAVs, or before using Admin Enablement functions on EAVs.
19.14 Out of space abends

Non-EAS eligible data sets will only be allocated in the track-managed space. When you try to allocate a non-EAS eligible data set and there is an insufficient amount of free space in the track-managed space on any volume, you will receive an error message indicating that there is insufficient freespace on the volumes in a storage group (see Figure 19-9). For non-EAS eligible data sets, this message will be issued regardless of the amount of freespace in the cylinder-managed space.

We recommend that you monitor the free space in the track-managed space as well as the cylinder-managed space. There are data sets that must reside in track-managed space and must be contained to one volume. Examples are ICF catalogs, PDSs and PDSEs.

You should monitor the free space falling below a predefined threshold in the track-managed space. This will allow you to take appropriate action to allow further allocations and extents for those data sets.

19.15 Migration positioning

Besides planning the hardware and product support for EAV, you should also consider your migration strategy early on. How you will migrate to EAVs will require different actions, whether you plan to migrate to EAV in place by expanding existing volumes, or by defining new EAVs. A migration can also be a combination of various migration techniques.

19.15.1 Expanding existing volumes with DVE

To expand existing volumes, you can use DVE in place after z/OS V1.10 is on all systems sharing the DS8000 volume. If you use this strategy, consider the impact on copy services relationships that are using the volumes you are expanding. You also need to evaluate the number of PAVs or HyperPavs, choose the appropriate volume size for your installation and expand to this size. You can expand a volume in increments taking into consideration VTOC and VVDS size in relation to expansion. Remember that expanding an existing volume will require that the capacity needed for the expansion is within the same extent pool already used for the current volume extents. Consolidate your data on a suitable number of volumes by copying data with DFSMSdss, TDMF, or zDMF, or by using a combination of these tools.
19.15.2 Creating new EAVs

You can migrate to newly created EAVs on a new address range that includes the new base base addresses and PAV addresses, Consolidate your data on a suitable number of volumes by copying data with DFSMSdss, TDMF or zDMF or by using a combination of these tools.

19.15.3 Planning for testing and implementing EAV

First, build a test environment of a single or a few EAVs in a separate DFSMS storage group to allow you to practice your EAV migration plan. In this environment you can test utilities and changes you plan to make to your procedures before going into production with EAVs.

Limit access to the environment through your ACS routines and by turning on and off the USEEAV parameter in SYS1.PARMLIB(IGDSMSxx).

Start out having the necessary support on your operation systems and your DS8000. As a minimum, set up test cases validating the following conditions:

- Create one or more EAVs on your DS8000.
- Create an EAV storage group and add your EAVs to this storage group.
- Update the track-managed and Total Volume thresholds for this storage group in the ISMF panels.
- Set your BPV value for the storage group in the ISMF panels.
- Activate EAV support by issuing the SETSMS USEEAV=YES command when ready.
- Allocate a number of VSAM and non-VSAM data sets.
- Evaluate where the VSAM data sets are allocated (below or above the 65520 cylinder limit). DISKMAP is a tool for this.
  - VSAM data sets with an allocation size > BPV should reside in cylinder-managed space. All others should reside in track-managed space.
- Test the BVP value and the impact of it on VSAM allocations.
- Where possible, test applications and vendor programs.
- Test your preferred utilities such as IEHLIST, DCOLLECT, DFSMSdss, DFSMShsm, ISMF and ISPF.
- Test copy services relationships if needed in your shop.
- Try to overfill the track-managed space in order to try to allocate unsupported data sets above the line, and see the messages as a result of this.
- Try coexistence between non-EAVs and EAVs by adding non-EAVs to your EAV test storage, with the IGD17279I message indicating insufficient space.
- Test migration scenarios using your EAV storage group.
- Test potential changes you plan to make to your current procedures (automation, reporting, and so on).
- Educate your end users on how EAV works.
- Gradually move live production data into your test storage group, when you feel comfortable about EAV.
19.15.4 Fallback from EAV processing

When planning to implement EAVs in your environment, you also need to consider a fallback plan. A stepwise implementation with pre-planned tests is always advisable, making it easier to back off of EAV.

The USEEAV parameter makes it easy to stop new data set allocations on your EAVs. Set the parameter to NO with the SETSMS command, while you solve any EAV related problems. USEEAV=NO will prevent new data set allocations. It will allow current data set extends.

After you have solved your EAV problem, enable EAV processing again by setting the USEEAV parameter to YES.

19.15.5 User education

EAV and its new extended addressing is a major change. Even though EAV is, in most cases, transparent to the end user, some basic introduction is recommended before implementation.

A basic understanding of the EAV basic structure is necessary for the user. It is important to understand changes in allocation as well as new and modified messages and output.

19.15.6 Miscellaneous changes

The ISPF data set list utility has been updated to display information about the track-managed space and on the entire volume space. See the new display in Figure 19-10.

```
Volume . : VVVVVV
Command ===>
Unit . . : 3390              Free Space

Table: VTOC Summary Information

<table>
<thead>
<tr>
<th></th>
<th>Total</th>
<th>Tracks</th>
<th>Cyls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tracks</td>
<td>480</td>
<td>1,026,961</td>
<td>68,464</td>
</tr>
<tr>
<td>%Used</td>
<td>1</td>
<td>980,655</td>
<td>65,377</td>
</tr>
<tr>
<td>Free DSCBS</td>
<td>23,972</td>
<td>Free</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extents . :</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Track Managed</th>
<th>Tracks</th>
<th>Cyls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tracks</td>
<td>1,051,785</td>
<td>980,656</td>
<td>65,377</td>
</tr>
<tr>
<td>%Used</td>
<td>2</td>
<td>980,655</td>
<td>65,377</td>
</tr>
<tr>
<td>Trks/Cyls</td>
<td>15</td>
<td>Free</td>
<td></td>
</tr>
</tbody>
</table>

F1=Help    F2=Split   F3=Exit    F9=Swap   F12=Cancel
```

*Figure 19-10  ISPF VTOC Summary Information of EAV*

To calculate the cylinder-managed space, subtract the Track-managed size figures from the TotalSize value.
19.15.7 DEVSERV display

DEVSERV PATHS has been changed to recognize EAVs with a different volume type (A) and greater volume size (see Figure 19-11).

![Figure 19-11 DEVSERV PATHS command output](image)

**Note:** The volume type 3390A and the volume size of 70119 (above 65520 cylinders).

The DEVSERV QDASD has also been updated to recognize the larger amount of cylinders (see Figure 19-12).

![Figure 19-12 DEVSERV QDASD command output](image)
Chapter 20. DS8000 EAV implementation

In this chapter we go through the necessary actions to actually implement EAV on your DS8000 from the hardware perspective. There are basic tasks that must be done, such as upgrading the DS8000 microcode, the DSCLI, and the DS8000 Storage Manager. There are also preparations that need to be made in your installation environment setup depending on the starting point for EAV implementation and the migration path.

In our scenario we describe a migration path based on an existing environment with a DS8000 installed.

If you do not have a DS8000 and will be building your environment, you should do a total review of your environment. We briefly address this subject at the end of this chapter in “Alternate scenarios” on page 367.
20.1 Implementing EAV for the DS8000

Migrating to EAVs on the DS8000 in an existing environment requires you to look at your existing environment in regard to current volumes sizes, the number of PAVs (or HyperPAVs), your distribution of data (VSAM and non-VSAM) and average sizes on your data sets.

When you know your existing environment in detail, you will be able to make the best possible design for the EAV environment. EAV implementation in your environment is not necessarily accomplished in one step; it might require several steps be taken.

20.1.1 Base installation

There are considerations for your existing environment before implementing EAVs.

DFSMS V1.10 prerequisite
A prerequisite starting out with the implementation of EAV volumes in production is the DFSMS V1.10 level - plus compatibility support on lower level systems sharing the DS8000 volumes.

**Note:** You will not be able to bring EAV volumes online to pre-DFSMS V1.10 systems. See Chapter 19, “Planning to use EAV” on page 327 in this book for more information.

Microcode level
The EAV feature is only available on the DS8000. To enable it you will have to migrate to the Release 4.0 Licensed Internal Microcode level.

The HMC installation of this code will do the necessary upgrade to the DS8000 Storage Manager to work with the full features of Dynamic Volume Expansion (DVE). The Storage Manager enables you to migrate to EAVs by using DVE to expand an existing volume in place.

After upgrading the DS8000 to the required LIC, upgrade the DSCLI to level 5.4.0.262. This will allow you to use the DVE functions through the DSCLI as an alternative to the DS8000 Storage Manager.

Capacity planning
EAV implementation might require you to expand your DS8000 capacity. You can make the additional capacity available by either using DVE on existing volumes, or by making the capacity available through a HCD definition of a new address range.

Greater capacity might call for expanded features on PPRC, PAV, FlashCopy, Global Mirror and even new features—for example, HyperPAV.

Copy Services considerations
If your environment is using Copy Services, you will need to plan potential volume expansions in relation to this function. The expansion process needs exclusive control of both the primary and secondary volume during expansion, meaning a withdraw of the PPRC relationship for a short while during volume expansion - and a subsequent full resynchronization of the Copy Services pair.
20.1.2 Alias considerations

Your current HCD defined environment will hold a balanced number of base and alias addresses at the LCU level to meet the performance criteria for the current volume sizes. The volumes defined at a LCU level can be of different volume sizes and types. They might have been migrated to the DS8000 with TDMF or other data migration tools, requiring at a minimum volumes of equal or larger volume size compared to the source volume.

A given number of aliases are bound to the individual LCU, designed to meet the performance criteria of the existing number of volumes (and their current accumulated size).

Assuming that you have a mix of different volume sizes (54 GB, 27 GB, 9 GB, and 3 GB) and you want to expand some or all of these over time to EAV sizes beyond 54 GB, the accumulated I/O-rate will potentially grow at least with the same percentage as the logical expansion. This calls for attention, when designing the volumes sizes in the target environment. The current number of aliases might have to be increased proportionally to the capacity growth.

If you start out having 64 base and 64 standard alias addresses, and that is a good balance for your environment, expanding the base volumes significantly in capacity might create a performance bottleneck for you.

20.1.3 HyperPAV and PAV features

You can, at this point, consider implementing the HyperPAV feature instead of the standard PAV feature, as the HyperPAV is far more efficient. The number of aliases required can be approximated by the peak I/O rates multiplied by the average response time (Little's Law).

For example, if the average response time is 4ms and the peak I/O rate is 2000 per second, then the average number of I/O operations executing at one time for that LSS during the peak is eight. Therefore eight PAV-aliases should be all that is needed to handle the peak I/O rate for the LSS, along with all the other base addresses in the LSS. Experience often shows a 10x reduction in PAV-alias UCBs using HyperPAV. This gives back a significant number of base device numbers from the 64K device address limit. The combination of HyperPAV and EAV allows you to significantly reduce the constraint on 64K device address limit and in turn increase the amount of addressable storage available on z/OS.

Implementing HyperPAV can be done without requiring changes to your configuration. The HyperPAV feature code 7899 has to be ordered and installed on the DS8000. It is activated in on EOF in two ways:

- Through a SETIOS HYPERPAV=YES|NO|BASEONLY command
- PARMLIB activation of SYS1.PARMLIB(IECIOSxx) allowing enablement of HyperPAV at the LPAR level using the HYPERPAV=OPTION.

The available options are:
- YES - Attempt to initialize LSSs in HyperPAV mode.
- NO - Do not initialize LSSs in HyperPAV mode.
- BASEONLY - Attempt to initialize LSSs in HyperPAV mode, but only start I/Os on base volumes.

The BASEONLY option returns the LSSs with enabled HyperPAV capability to a pre-PAV behavior for this LPAR.
Before activation, check if you have the necessary maintenance on your systems to run HyperPAV. For more information about the activation of HyperPAV, refer to the manual, *MVS Setting Up A Sysplex*, SA22-7625.

There is, of course, an upper limit of total throughput on the DS8000. The potential physical capacity is not always the limit, as online systems with a high volume of transactions might exhaust the DS8000 before the physical capacity limit is reached. We recommend that you do a Disk Magic™ simulation before deciding how to expand the capacity (and the workload) in your current DS8000.

### 20.2 Creating EAVs

At this stage you should have done the necessary preparation to build your EAV environment based upon an existing environment.

The hardware and software prerequisites are in place, the required features are installed, and there is sufficient capacity in the DS8000 to hold the EAV volumes you want to expand to based on DVE, or for new volumes defined in a new address range to hold newly defined EAVs.

The intermediate size (if you want to take a stepwise approach towards larger volumes) and or target size of your EAV is decided along with the estimated number of PAVs or HyperPAVs per LCU.

#### 20.2.1 Creating a new EAV

Use the DS8000 Storage Manager or the DSCLI to define the new 3390 Model that you plan to start out with. In the following examples, we use the DS8000 Storage Manager to illustrate the process. At the end of the chapter, we cover the process of defining and extending a volume by using DSCLI.

The new 3390 Model A is now a selectable volume type such as a 3390-3, 3390-9, and 3390 customer volume in the DS8000 Storage Manager. The 3390 Model A is not a new device type and is not defined in a special way in the HCD. The Model A type only applies to the DS8000 as a device that can expand beyond 65520 cylinders in capacity.

The 3390 Model A can be defined as new, when defining a new EAV volume beyond 65520 cylinders, but can also be defined as low as one cylinder (EAV capable volume). Expanding a 3390 type of volume that is below 65520 cylinders to be larger than 65520 cylinders will at the same time convert the volume from any existing 3390 standard volume type to a 3390 Model A.

Using the IBM System Storage DS8000 Storage Manager, you define an EAV (3390 Model A) as you would any volume as shown in Figure 20-1. Select the extent pool you will create the EAV in and click **Next** to continue.
Figure 20-1  Create zSeries® volume → Select extent pool panel

Figure 20-2 shows the Define base volume characteristics panel. On this panel you specify the Volume type of 3390 Mod A. Note also the Extent allocation method selection where you specify Rotate volumes or Rotate extent. Rotate volumes means that the volume is striped on the first array, as long as there is capacity. Rotate extent stripes the volumes over the array in the entire extent pool, rotating one at a time. We specify Rotate volumes and click Next to continue.
The next panel is the *Define base volume properties* panel as shown in Figure 20-3. In this panel you specify the size of the volume. Keep in mind that you can specify a value from one cylinder to 262668 cylinders depending on the amount of available storage and the size of the EAV you need. Note that we had 121317 cylinders available for the EAV. Click **Next** to continue.

![Figure 20-3  Define base volume properties panel](image-url)
The next panel is the *Create volume nicknames* panel as shown in Figure 20-4. Enter the volume Prefix and Suffix. Click **Next** to continue.

![Figure 20-4 Create volume nicknames panel](image)

The next panel is the *Define alias assignments* panel. After you enter the required information, click **Next** to continue.

![Figure 20-5 Define alias assignments panel](image)
The next panel is the *Verification* panel as shown in Figure 20-6. Review the information you have entered in the previous panels and click **Finish** to continue.

![Verification panel](image-url)

*Figure 20-6  Verification panel*
You will see the *Long Running Task Indicator* panel as shown in Figure 20-7. In this panel you will see the progress of the Create volume task. Choose the next action you want to take. We chose **Close**.

![Figure 20-7 Long Running Task Indicator panel](image)

Figure 20-7 shows the list of zSeries volumes. Note the new 3390 Mod A with a nickname of MLDC65 that was just created.

![Figure 20-8 Listing of the zSeries volumes](image)
20.2.2 ICKDSF initialization

To have the volume recognized on z/OS, do a normal ICKDSF initialization. Sensing a volume larger than 65520 cylinders, ICKDSF will know this is an EAV volume and create the VTOC index with a record length and a block size of 8192 bytes to meet the need for extended addressability.

Vary the volume online to z/OS and verify the size through a `DS QD,uuuu` DEVSERV command, which will show the number of available cylinders as shown in Figure 20-9.

The DEVSERV PATHS operator command is updated to report a new field for the number of cylinders. The DEVSERV QDASD operator command is changed to recognize and display the larger device size for an EAV. This larger size is reflected in the CYL column of the DEVSERV QDASD display.

```
RESPONSE=SC70
IEE459I 21.19.50 DEVSERV QDASD 002
UNIT VOLSER SCUTYPE DEVTYPE CYL SSID SCU-SERIAL DEV-SERIAL EFC
DE65 vvvvv 2107922 2107900 74571 89EE 0175-BALB1 0175-BALB1 *OK
**** 1 DEVICE(S) MET THE SELECTION CRITERIA
**** 0 DEVICE(S) FAILED EXTENDED FUNCTION CHECKING
```

Figure 20-9 DEVSERV QDASD command results for EAV

20.2.3 Expanding existing volumes

Existing 3390 volumes of any type can be expanded by DVE to either a larger 3390 customized size or be expanded to an EAV volume (larger than 65520 cylinders) and, at the same time, be converted to a 3390 Model A through the DS8000 Storage Manager.

Expanding an existing volume will require that the capacity needed for the expansion is within the same extent pool as already used for the current volume extents.
Volume expansion
In the DS8000 Storage Manager, select the volume you want to increase the capacity of. In this scenario we selected the volume MLDE65. From the Select Action pull-down menu, choose Increase Capacity as shown in Figure 20-10.

![Select Action - Increase Capacity](image)

In the Increase Capacity panel, you specify the number of cylinders that you want to increase the volume to. In our scenario, the existing 3390 Model A is 75684 cylinders. We entered a value of 75700 in the Specify the new capacity field. Remember that the number of cylinders in cylinder-managed space must be a multiple of 1113. Because 75700 is not a multiple of 1113, we receive an error message as shown in Figure 20-11 and are given two options:

- Click Cancel to end the Increase Capacity task and specify a valid cylinder value.
- Click Continue to accept the DS8000 Storage Manager cylinder calculation to the next valid multiple of 1113 and submit the Increase Capacity task.
Figure 20-11  Invalid cylinder specification for Increase Capacity

After specifying a valid cylinder amount in the Specify the new capacity field, you receive a warning message as to the effect of the task (see Figure 20-12). Click OK to continue or Cancel to end the task.

Figure 20-12  Warning message as part of volume expansion
After clicking **OK**, the Increase Capacity task is submitted. You will see the task progress in the *Long Running Task Properties* panel as shown in Figure 20-13.

![Successful expansion of volume](image)

**Figure 20-13  Successful expansion of volume**

**Copy services considerations**

Trying to do a Dynamic Volume expansion while a Copy Services relationship is active will be rejected by the DS8000 Storage Manager. You will receive an error message as shown in Figure 20-14. The expansion of an existing volume also requires potential Copy Services relationships to be withdrawn during the expansion. Plan the break of the Copy Services relationship to be done at a time where it has the least impact on your disaster recovery plan for the volumes involved. The expansion process is fast (a matter of seconds), but the resynchronization might take some time depending on the amount of data on the volume.

In case of a mirrored environment, repeat the increase process for all primary and secondary volumes involved.
Pre-DFSMS V1.10 system consideration
A volume larger than 65,520 cylinders cannot be brought online to a pre-z/OS V1.10 system or expanded over 65,520 cylinders on a pre-z/OS V1.10 system. You will receive an error message as shown in Figure 20-15.
20.2.4 Reformatting VTOC

After creating an EAV, the next step is to reformat the VTOCs of the volumes with added capacity to reflect the new capacity. This is done with the REFORMAT command in ICKDSF as illustrated in Figure 20-16. Use the REFORMAT parameter to rebuild the VTOC in its current location, using the same track size. If the VTOC is an indexed VTOC, the index data set will be deleted. If the device is online to MVS, the index data set will be rebuilt to reflect the VTOC changes.

Figure 20-16  ICKDSF REFORMAT VTOC

The JOBLOG from the ICKDSF job will display the changed size of the volume involved, as shown in Figure 20-17.

Figure 20-17  ICKDSF JOBLOG

In the SYSLOG the volume expansion is recognized with the message shown in Figure 20-18. You might want to send an alert to the people responsible for this function based on this message or even automate the REFORMAT job.

Figure 20-18  Message IEA019I

Your devices are now ready for use on z/OS. Continue the process of adding the new volumes to your SMS Storage Groups as you normally would. The volumes that have been expanded in place will probably be there already.
20.2.5 Creating and expanding volumes using DSCLI

To use the newest functions in DVE from DSCLI, be sure to have the correct version of DSCLI installed. At the time of writing, that version was 5.4.0.262. The following examples show volume creation and expansion using the DSCLI.

Volume creation

To create a new volume, use the MKCKDVOL command. Add the information for the DS8000 serial number, the extent pool name, the volume id, the capacity, and the UCBs involved. Figure 20-19 is an example of creating four sequential addresses.

```
DSCLIi> mkckdvol -dev IBM.2107-75FA120 -extpool P1 -name my_volume_id -cap 3339
0100 0101 0102 0103
```

Figure 20-19 DSCLI mkckdvol command

The resulting output can be seen in Figure 20-20.

```
Sun Aug 11 02:23:49 PST 2008 IBM DS IBM.2107-75FA120
CKD volume 0100 successfully created.
CKD volume 0101 successfully created.
CKD volume 0102 successfully created.
CKD volume 0103 successfully created.
```

Figure 20-20 Result of mkckdvol command

Volume expansion

Use the DSCLI CHCKDVOL command to expand the size of volumes. An example of expanding a volume at address 0F65 to a 3390 Model A with a new capacity of 75684 cylinders is shown in Figure 20-21.

```
dscli> chckdvol -cap 75684 0f65
Date/Time: March 19, 2008 8:24:22 PM EDT IBM DSCLI Version: 5.4.0.262 DS: IBM.21
07-75BALB1
CMUC00332W chckdvol: Some host operating systems do not support changing the volume size. Are you sure that you want to resize the volume? [y/n]: y
CMUC00022I chckdvol: CKD Volume 0F65 successfully modified.
```

Figure 20-21 DSCLI chckdvol command

Notice the warning from the DS8000 that some operating systems do not support changing the volume size beyond 65520 cylinders.
20.3 Monitoring EAV

Having implemented EAV in your production environment, we recommend that you monitor your storage subsystem for potential performance issues.

Use RMF to study throughput and response times before and after migrating to EAV. If the response time has grown, you should study the component contributing the most to these prolonged response times (IO-rate, queue time, disconnect time, or connect time). Adjust the workload and the setup accordingly.

The bottleneck could also be the VTOC, VVDS, or space in the track-managed space or the cylinder-managed space.

To avoid serious impact on your systems, it is advisable to monitor all the critical components and thresholds by setting up alerts, in order to give you time to react.

20.4 Alternate scenarios

If you do not base your migration to EAV on an existing DS8000 environment, for example, if you are migrating from hardware that does not support the EAV function, you have a different scenario to implement EAVs.

You will have to go through planning a new environment with an install of DS8000 and necessary features and HCD definitions of base and PAV addresses based on the target environment you plan to end up with.

This configuration will be a mixture of base and PAV addresses (PAV or HyperPAV) based on an estimated maximum volume size. It would again be advisable to do a simulation on how this environment will work based on your existing workload (plus potential expected growth) through a DiskMagic simulation.

Review Chapter 19, “Planning to use EAV” on page 327 in this book covering general considerations for going to a new volume type and size.
DFSMSShsm and EAV

In this chapter we describe the following topics:

- DFSMSShsm owned volumes
  - Recall / Recover scenario
  - EAV and SMALLDATASET PACKING
  - DFSMSShsm DUMP at a volume level
- DFSMSShsm space management
21.1 DFSMShsm owned volumes

In general, IBM recommends that EAV volumes should not be used for DFSMShsm owned volumes (ML1, ML2 and Backup). This is because DFSMShsm cannot use the cylinder-managed space. The one exception is small data set packing (SDSP) data sets.

Migrated and backed up data sets cannot reside in cylinder-managed space because they are stored as non-VSAM data sets. The on exception is SDSP data sets on ML1. Therefore, there is no reason to define ML2 backup volumes with more that 65520 cylinders.

Note: EAV volumes must not be used for ML1 volumes until all DFSMShsm hosts in the HSMplex are at z/OS V1.10 or higher.

21.1.1 Recall / Recover scenario

In this section we go through a scenario showing the DFSMShsm Space Management on an EAV Volume.

Figure 21-1 shows a DISKMAP output where data set MHLEAV.EDSBIG02 is allocated in a cylinder-managed space.

The data set MHLEAV.EDSBIG02 is backed up and migrated to ML2 (see Figure 21-2).

--- End VTOC 2 DATA SETS >>>

Figure 21-2 Data set MHLEAV.EDSBIG02 migrated to ML2
After the data set was recalled (see Figure 21-3) it was allocated on an EAV and went back to cylinder-managed space (see Figure 21-4).

Figure 21-3  MHLEAV.EDSBIG02 RECALL processing

The status of volume MLDE65 is changed to DISNEW in Storage Pool MHLEAV (Figure 21-5) and the data set MHLEAV.EDSBIG02 was deleted. Only the volume MLDC65 has a status of ENABLE in Storage Group MHLEAV.

Figure 21-5  Volume MLDC65 status in Storage Group MHLEAV
The data set MHLEAV.EDSBIG02 will now become a candidate for being allocated on volume MLDC65 after being recovered. The volume MLDC65, however, does not have available space above 65520 cylinders, because it was defined with 70118 cylinders, that is, only with 4,599 cylinders in the cylinder-managed space (70118-65519 = 4599). Figure 21-6 shows the size of the MLDC65 volume.

Figure 21-6  MLDC65 volume size

The volume MLDC65 has free space (see Figure 21-7), but only in the track-managed space because it has only 4599 cylinders in cylinder-managed storage, as previously discussed. In the cylinder-managed space there is a single data set allocating that entire area.

Volume . : MLDC65
Command ===>

Unit . . : 3390               Free Space

VTOC Data
Tracks . : 900        Size . . : 794,306       52,832
%Used . . : 6         Largest . : 176,280       11,752
Free DSCBS: 42,632    Free Extents . : 860

Volume Data
Tracks . : 1,051,785   Size . . : 794,306       52,832
%Used . : 24          Largest . : 176,280       11,752
Trks/Cyls: 15         Free Extents . : 860

Figure 21-7  Volume MLDC65 free space
When data set MHLEAV.EDSBIG02 is recovered, the data set is allocated in the track-managed space. Figure 21-8 shows the output of DISKMAP where you can see the data set MHLEAV.EDSBIG02 allocated in cylinder/track 53908 which is below 65520 cylinders.

21.1.2 EAV and SMALLDATASET PACKING

SDSP data sets can be allocated in the cylinder-managed space of EAVs (see Figure 21-9).

![Figure 21-8 DISKMAP of MLDC65 showing data set MHLEAV.EDSBIG02 below 65520 cylinder](image)

**Note:** The scenario above shows that a VSAM data set that was allocated in the cylinder-managed space can be recovered by DFSMSShsm in any part of the volume, either the track-managed or cylinder-managed space.

![Figure 21-9 SDSP on an EAV](image)
21.1.3 DFSMShsm DUMP at a volume level

An example of DFSMShsm doing a DUMP COPY of the EAV volume is shown in Figure 21-10. There are no new keywords to execute this DFSMShsm function.

![Figure 21-10](image)

21.2 DFSMShsm space management

Data set size is a determining factor for the order in which data sets are migrated. Managing an entire EAV with one threshold is not sufficient because average data set size in the cylinder-managed space will likely exceed the average data set size in track-managed space.

DFSMShsm has been changed to support EAV in relation to thresholds on both the track-managed space as well as the total view of an EAV volume. A track-managed threshold was added to ensure that space management is done especially for the track-managed space, even if the overall volume has not exceeded the threshold value.

Refer to “DFSMShsm space management” on page 343 for information of the changes in DFSMShsm in support of EAV, both SMS-managed and non-SMS managed, and how to implement this support.
APARs to be reviewed

In this appendix we provide the text of APARs that have been mentioned in the book and APARs that should be reviewed.
MISSING IOS070E DEV, MOUNT PENDING MESSAGE AT Z/OS 1.9 WHEN ENABLED VIA PARMLIB
IECIOSSXX EX:MIH MOUNTMSG=YES,MNTS=03:00

Summary Page
APAR Identifier: OA24276  Last Changed: 03/07/2008
Symptom: IN INCORROUT  Status: INTRAN
Severity: 3  Date Closed:
Component: 5752SC1C3  Duplicate of:
Reported Release: 740  Fixed Release:
Component Name: Special Notice:
Customer Name: PHOENIX SOFTWARE INTE  Customer #: 7121622 00
Current Target Date: / / Flags:
SCP:
Platform:
Status Detail:

'Fix Page
PE PTF List:
PTF List:
Parent APAR:
Child APAR List:

'Submitter Page
Error.
Description: Even after the mount message function is enabled correctly, IOS070E dev, MOUNT PENDING messages do not occur at Z/OS 1.9.

Local Fix: ++USERMOD(ZAPMIHP).
++VER(Z038) FMID(HBB7740) PRE(UA39317).
++ZAP(IOSRIHP).
   NAME IEANUC01 IOSRIHP
   VER 2B5A A784 0124
   REP 2B5A A784 0006

Responder Page
Problem
Summary:
Problem
Conclusion:
Temporary Fix:
Comments:
Modules/Macros:
SRLS:
Return Codes:
Circumvention:
Message To
Submitter:

Server=W36 \
APAR OA20749

APAR Identifier: OA20749 Last Changed: 01/28/2008
Symptom: IN INCORROUT Status: OPEN
Severity: 3 Date Closed:
Component: 5752SC1CW Duplicate of:
Reported Release: 730 Fixed Release:
Component Name: Special Notice:
Customer Name: AETNA LIFE INSURANCE CO Customer #: 0070001 00
Current Target Date: 04/11/2008 Flags:
SCP:
Platform:
Status Detail: REVIEW

'Fix Page
PE PTF List:
PTF List:
Parent APAR:
Child APAR List:

'Submitter Page
Error.
Description: The MVS Initialization and Tuning Guide (SA22-71591-04) in section 2.2 "Page Data Set Sizes" indicates page datasets can be no more than 1M (1,048,576) slots (4GB) in size. As of z/OS R1.8, this is no longer true, page datasets can now be allocated as large as 16M (16,777,216) slots (64GB) in size. HOWEVER, ASM still supports only up to 4G page datasets.

Local Fix: IDCAMS DEFINE PAGESPACE supports >4G page datasets at z/OS R1.8. HOWEVER, ASM still only supports page datasets up to 4G in size.

APAR OA23206

DS DELETED FROM LO AFTER CLOSE ERROR ON ML2

Last changed

08/03/04
Symptom

IN INCORROUT
Status

CLOSED PER
Severity

3
Date closed
A data set being migrated could be deleted from the primary volume incorrectly if a close error occurs on the ML2 tape during a recall takeaway. This tape volume is needed for
recall before the migration tape closes. The data set is deleted on the primary volume, and there is a valid migrated copy of the data set despite on the error on the tape. But the data set is recataloged back on the primary volume.

Messages received could be:
ARCO9231 ERROR CLOSING TAPE DATA SET HSM.HMIGTAPE.DATASET, RC=0
ARCO208I TRAP FOR ERROR CODE 0020, MODULE ARCTCLOS

Local fix

Data will need to be recoverd from a backup if this happens before fix is applied.

Problem summary

**********************************************************************************************
* USERS AFFECTED: All users of z/OS DFSMShsm V1R8 and V1R9.  
**********************************************************************************************
* PROBLEM DESCRIPTION: A data set being migrated could be deleted from the primary volume incorrectly if a close error occurs on the ML2 tape during a recall takeaway. The close error in this case was an abend 714-OC.  
**********************************************************************************************
* RECOMMENDATION:  
**********************************************************************************************
A data set being migrated could be deleted from the primary volume incorrectly if a close error occurs on the ML2 tape during a recall takeaway. The close error in this case was an abend 714-OC.

Problem conclusion

DFSMShsm has been changed to correct the problem.

Temporary fix

********
* HIPER
********
OA23206

Comments

None

Modules/macros

None
PTF OA21487

Summary Page
APAR Identifier: OA21487 Last Changed: 08/05/02
Symptom: NEW FUNCTION Status: CLOSED UR1
Severity: 4 Date Closed: 08/03/26
Component: 5695DF111 Duplicate of:
Reported Release: 180 Fixed Release:
Component Name: DEV SUPPORT SER Special Notice:
Customer Name: Customer #: 9999998 99
Current Target Date: 08/04/18 Flags: SPEC/ATTN/Y XSYSTEM
SCP:
Platform:
Status Detail: SHIPMENT
' Fix Page
PE PTF List:

PTF List: Release 180 : UA40228 available 08/04/09 (F804)
Release 190 : UA40229 available 08/04/09 (F804)

Parent APAR:

Child APAR List: OA22449 OA22900 OA22976 PK56092

' Submitter Page
Error.
Description: NEW FUNCTION COEXISTENCE SUPPORT

Local Fix:

Responder Page

Problem

Summary: *************************************************************************
* USERS AFFECTED: IDCAMS LISTDATA PINNED and DEVSERV QDASD *
* needs this APAR to recognize EAV volume *
* under z/OS R1V8.0 and R1V9.0. *
*************************************************************************

* PROBLEM DESCRIPTION: APAR provides message IDC31562I for *
* IDCAMS LISTDATA PINNED when PINNED *
* device is EAV under z/OS V1R8.0 *
* or V1R9.0. *
*
* Also DEVSERV QDASD provides EAV volume *
* size under z/OS V1R8.0 and V1R9.0. *
*************************************************************************

* RECOMMENDATION: *
*************************************************************************

For IDCAMS LISTDATA PINNED, this change returns message IDC31562I THE PINNED PARAMETER IS NOT AVAILABLE FOR THE SPECIFIED SUBSYSTEM OR DEVICE for EAV volume. For DEVSERV QDASD, this APAR re-arranges column to show the device cylinder numbers.
See ++HOD(DOC) to have the detail message layout change for IEE459I message.

Problem

Conclusion:  KEYWORDS: R10COEXS/K

Temporary Fix:

Comments: N/A

Modules/Macros: IDCSS05  IGUDSX02

SRLS: SA22762717 SA22763715 SC26740209

Return Codes:

Circumvention:

Message To
Submitter:
Migration from z/OS 1.7 requires the PTF for the following APAR to be applied on z/OS 1.7 to allow access to objects created by the z/OS V1R8 or z/OS V1R9 Program Binder.

OA13294
OA13525
(is still described in 7617V9PDSE.FM).
* PROBLEM DESCRIPTION: Compatibility with PM5 program objects. *
****************************************************************
* RECOMMENDATION:                                              *
****************************************************************
This fix provides compatibility with PM5 level program objects introduced in HBB7730.

Problem
Conclusion:

Temporary Fix:

Comments: Note that the HBB7730 PTF for this APAR does not contain new function - it is provided to prevent SMPE error messages during installation of HBB7730 over an earlier release with this fix installed. Additional symptoms that may indicate that this fix is required include:
- ISPF browse fails with IEC036I 002-C8
- Programs which invoke IEWLCNV1 may fail

Modules/Macros: IEWLCNVX IEWPLMH IEWPMAR

SRLS: NONE

Return Codes:

Circumvention:

Message To Submitter:

OA13525

Summary Page
APAR Identifier: OA13525 Last Changed: 06/05/02
Symptom: FUNCTION Status: CLOSED UR1
Severity: 3 Date Closed: 05/11/06
Component: 5695DF115 Duplicate of:
Reported Release: 1G0 Fixed Release:
Component Name: EXTENDED DATA S Special Notice:
Customer Name: PK0142 Customer #: 9999999 00
Current Target Date: 05/12/19 Flags:
SCP:
Platform:
Status Detail: TESTPACKAGE

Fix Page
PE PTF List:

PTF List: Release 1G0 : UA22348 available 06/04/04 (F604)
Release 1H0 : UA22349 available 06/04/04 (F604)
Release 1J0 : UA22350 available 06/04/04 (F604)
Release 1K0 : UA22351 available 06/04/04 (F604)

Parent APAR: OA13294

Child APAR List:

Submitter Page
Error.
Description: new function

Local Fix:

Responder Page
Problem
Summary:

* USERS AFFECTED: Users of Program Objects built in z/OS
  * 1.8 (PO5) in the earlier releases of z/OS
  * (1.3, 1.4, 1.5, 1.6 and 1.7).

* PROBLEM DESCRIPTION: This is the compatibility fix to use Program Objects built in z/OS 1.8 (PO5) in the earlier releases of z/OS (1.3, 1.4, 1.5, 1.6 and 1.7).

* RECOMMENDATION:

This is the compatibility fix to use Program Objects built in z/OS 1.8 (PO5) in the earlier releases of z/OS (1.3, 1.4, 1.5, 1.6 and 1.7).

Problem
Conclusion:

Temporary Fix:

Comments: PO5 compatibility is now provided by FAMS in z/OS 1.3, 1.4, 1.5, 1.6 and 1.7.

Modules/Macros: IGWAMAM7 IGWAMCC3 IGWAMCC6 IGWAMCV1 IGWAMFG0 IGWAMLMO

SRLS: NONE
Return Codes:

Circumvention:

Message To
Submitter:

APAR OA22738

Summary Page
APAR Identifier: OA22738 Last Changed: 08/04/01
Symptom: IGD17295I Status: CLOSED PER
Severity: 3 Date Closed: 08/02/14
Component: 5695DF101 Duplicate of:
Reported Release: 180 Fixed Release: 999
Component Name: STORAGE MGMT SU Special Notice:
Customer Name: VERIZON DATA SERVICES INC Customer #: 3981401 00
Current Target Date: 08/04/04 Flags: SPEC/ATTN/Y PERVASIVE
SCP:
Platform:
Status Detail: SHIPMENT
' Fix Page
PE PTF List:

PTF List: Release 180 : UA39453 available 08/03/15 (F803)
Release 190 : UA39454 available 08/03/15 (F803)

Parent APAR:

Child APAR List:

' Submitter Page
Error.
Description: Partitioned data sets are not eligible to be multi-volume data sets and are normally failed, if they have a volume count greater than one, with message:

IGD17293I DATA SET XYZ HAS
PARTITIONED ORGANIZATION AND IS NOT ELIGIBLE
TO BE A MULTI-VOLUME DATA SET, ALLOCATION FAILED

Because of APAR OA13500, a new message was created:

IGD17295I dsname HAS PARTITIONED ORGANIZATION AND CANNOT HAVE A DATA CLASS WITH A DYNAMIC VOLUME COUNT GREATER THAN ONE, ALLOCATION FAILED
This change was implemented at base release z/OS 1.8 and will stop partitioned data sets from receiving a dynamic volume count greater than one. The change was not documented in the z/OS 1.8 Migration Guide so info apar III4304 was created to inform customers to change their ACS routines to prevent partitioned data sets from getting assigned a Data Class with a dynamic volume count greater than one.

New data sets created using the LIKE keyword where the LIKE data set is a PDS/PDSE will be identified as a physical sequential data set during ACS processing. This situation creates a new problem for customers because they are not able to prevent the IGD17295I via ACS routines. Because of this new problem this APAR will reverse the changes made in component 5695DF101.

Additional Keywords:
msgIGD17295I DVC DATACLAS

Local Fix: Usermod is currently available per customer request to remove the message.

Responder Page
Problem

Summary:****************************************************************
* USERS AFFECTED: All users of DFSMS V1R8 and higher.  *
****************************************************************
* PROBLEM DESCRIPTION: If a SMS data set with partitioned*
  * organization is being created with a  *
  * DATA CLASS that has a DYNAMIC VOLUME *
  * COUNT > 1, this creation will be failed *
  * with the IGD17295I message.  *
****************************************************************
* RECOMMENDATION:                                            *
****************************************************************

This causes problems for customers, particularly those customers that use the LIKE= JCL keyword on the DD card. The ACS routines cannot determine if the new data set being created is PO and are unable to direct these data sets to a DATA CLASS that has a DYNAMIC VOLUME COUNT of 0 or 1.

Problem

Conclusion: The DYNAMIC VOLUME COUNT check for PO data sets will be removed. Consequently the IGD17295I message will be eliminated.

Temporary Fix:

Comments:
Appendix A. APARs to be reviewed

Modules/Macros: IGDVTSCR

SRLS: NONE

Return Codes:

Circumvention:

Message To Submitter:

APAR OA17965

Summary Page
APAR Identifier: OA17965 Last Changed: 08/04/29
Symptom: NEWFUNC Status: CLOSED UR1
Severity: 4 Date Closed: 07/01/04
Component: 5695DF186 Duplicate of:
Reported Release: 1K0 Fixed Release:
Component Name: DFSMSRMM Special Notice:
Customer Name: Customer #: 9999990 72
Current Target Date: 07/01/10 Flags: SPEC/ATTN/Y NEW FUNCTION XSYSTEM
SCP:
Platform:
Status Detail: SHIPMENT

PE PTF List:

PTF List: Release 1J0 : UA31619 available 07/01/16 (F701)
Release 1K0 : UA31620 available 07/01/16 (F701)
Release 180 : UA31621 available 07/01/16 (F701)

Parent APAR:

Child APAR List:

Local Fix:
Responder Page
Problem
Summary:  ****************************************************************
* USERS AFFECTED: All users of DFSMSrmm.                       *
****************************************************************
* PROBLEM DESCRIPTION: This APAR provides the toleration       *
*                      for future changes in a DFSMSrmm        *
*                      release later than V1R8.                *
****************************************************************
* RECOMMENDATION: Install the appropriate PTF(s) for           *
*                 APAR OA17965 on all systems sharing          *
*                 the same DFSMSrmm control data set.          *
****************************************************************

Problem
Conclusion:  KEYWORDS: R10COEXS/K

Temporary Fix:

Comments:  This APAR delivers toleration for future new function and may
require action. You do not need any action unless you see
EDG0229E during initialization of DFSMSrmm and you plan to
start DFSMSrmm on a release later than z/OS V1R8 in a RMMPLEX
sharing the same CDS.

Recommendation: If you have no CDSID implemented for your
DFSMStmm control data set, add the CDSID operand to the OPTION
statement in your DFSMSrmm parmlib member and refresh DFSMSrmm
to set a CDSID.

Changes to z/OS MVS System Messages Volume 5 (EDG - GFS)
--------------------------------------------------------
MESSAGE EDG0229E

EDG0229E NO CDSID WAS SPECIFIED FOR OPTION IN PARMLIB

EXPLANATION: During initialization DFSMSrmm checks to ensure
that you have a CDSID specified in the EDGRMMxx parmlib
member. A CDSID is required in a future release of z/OS.

SYSTEM ACTION: Processing continues.

OPERATOR RESPONSE: Notify the system programmer.

SYSTEM PROGRAMMER RESPONSE: You must add a CDSID to your
OPTION statement in EDGRMMxx before attempting to start
DFSMStmm on a release later than V1R8. IBM recommends
that you implement a CDSID now and not wait for the
future release.
You can do this by adding a CDSID operand to the
OPTION command in parmlib.
SOURCE: DFSMSrmm

DETECTING MODULE: EDGPARM

Changes to DFSMSrmm Application Programming Interface
-----------------------------------------------

"Command SFIs" table in chapter "SFIs for Subcommand Output Data" of "Appendix A. Structured Field Introducers" has been changed.

+-------------------------------------+-------------------------------------+-------------------------------------+-------------------------------------+
| Figure 1. Command SFIs             |                                    |                                    |
|                                    | SFI NUMBER                         | SFI NAME                            | SFI LENGTH                         |
|                                    |                                    |                                    | SFI DATA                           |
|                                    |                                    |                                    | TYPE                               |
|                                    | . . .                              | . . .                               | . . .                              |
|                                    | X'812900'                          | CDSQ                                | 3                                 |
|                                    |                                    |                                    | Binary(8)                          |
|                                    | . . .                              | . . .                               | . . .                              |

+-------------------------------------+-------------------------------------+-------------------------------------+-------------------------------------+
| Figure 1. Command SFIs (cont.)     |                                    |                                    |
|                                    | SFI NUMBER                         | DATA DESCRIPTION                    | SUBCMD MAND  |
|                                    |                                    |                                    |             |
|                                    | . . .                              | . . .                               | . . .       |
|                                    | X'812900'                          | CDS ENQ                             | LC CNTL     |
|                                    |                                    |                                    |             |
|                                    | . . .                              | . . .                               | . . .       |

~**** PE08/02/25 FIX IN ERROR. SEE APAR OA24092 FOR DESCRIPTION

Modules/Macros: EDGACTIO EDGAMGR EDGBKUP EDGCOMMA EDGDOC EDGDOCS EDGIRIP
EDGMAIN EDGMCATX EDGMCMD EDGMCREC EDGMDIO EDGMEQU EDGMFIC
EDGMFIO EDGMFIS EDGMHKP EDMLCS EDGMTAB EDGMTSO EDGMUPD
EDGMWTO EDGPARM EDGQMGR EDGRPTX EDGSOCE EDGTSO EDGUTIL
EDGVRCE1 EDGWAMFI EDGXHCLL EDGXHINT EDGXHXML EDGXMV EDGXSFB
EDGXSFB EDGXSFT EDGSFTX

SRLS: SC26740304 SC26740305 SC26740306 SA22763509             SA22763510 SA22763511

Return Codes:

Circumvention:
Message To
Submitter:

APAR OA16523

Summary Page
APAR Identifier: OA16523 Last Changed: 08/04/29
Symptom: NEWFUNCTION Status: CLOSED UR1
Severity: 4 Date Closed: 06/09/08
Component: 5695DF186 Duplicate of:
Reported Release: 1J0 Fixed Release:
Component Name: DFSMSRMM Special Notice:
Customer Name: Customer #: 9999990 72
Current Target Date: 06/08/31 Flags: SPEC/ATTN/Y NEW FUNCTION XSYSTEM
SCP:
Platform:
Status Detail: SHIPMENT

Fix Page
PE PTF List:

PTF List: Release 1J0 : UA29139 available 06/09/22 (F609)
Release 1K0 : UA29140 available 06/09/22 (F609)
Release 180 : UA29141 available 06/09/22 (F609)

Parent APAR:

Child APAR List:

Submitter Page
Error.
Description: Preconditioning APAR for Tape SPE toleration.
Enables a Client/Server RMMplex to handle different
versions of transferred records.
Additional Keywords:
   DFSMSrmm DFRMM RMM client server

Local Fix:

Responder Page

Problem
Summary: *****************************************************
* USERS AFFECTED: All DFSMSrmm Client/Server users. *
*****************************************************
* PROBLEM DESCRIPTION: Preconditioning APAR for Tape SPE *
* toleration. *

390  DFSMS V1.10 and EAV Technical Guide
Enabling a Client/Server RMMplex to handle different versions of transferred records.

RECOMMENDATION: Refer to the ++HOLD information.

Problem
Conclusion: KEYWORDS: R1O0CEXS/K

Temporary Fix:

Comments:

Modules/Macros: EDGMFIC EDGMFICB EDGMFIS

SRLS: NONE

Return Codes:

Circumvention:

Message To Submitter:

APAR OA16524

Summary Page
APAR Identifier: OA16524 Last Changed: 08/07/23
Symptom: NEWFUNC Status: CLOSED UR1
Severity: 4 Date Closed: 06/09/28
Component: 5695DF186 Duplicate of:
Reported Release: 1G0 Fixed Release:
Component Name: DFSMSRMM Special Notice:
Customer Name: Customer #: 9999990 72
Current Target Date: 06/08/31 Flags: SPEC/ATTN/Y NEW FUNCTION XSYSTEM
SCP:
Platform:
Status Detail: SHIPMENT
Fix Page

PTF List: Release 1G0 : UA29597 available 06/10/06 (F610)
Release 1H0 : UA29598 available 06/10/06 (F610)
Release 1J0 : UA29599 available 06/10/10 (F610)
Release 1K0  : UA29600 available 06/10/10 (F610)
Release 180  : UA29601 available 06/10/10 (F610)

Parent APAR:

Child APAR List:

  ` Submitter Page
  Error.
  Description: Toleration support for new function

Local Fix:

Responder Page

Problem

Summary: *****************************************************************************
  * USERS AFFECTED: All users of DFSMSrmm.  *
  *****************************************************************************
  * PROBLEM DESCRIPTION: Toleration APAR for Tape SPE  *
  *****************************************************************************
  * RECOMMENDATION: Refer to the ++HOLD information.  *
  *****************************************************************************

Tolerates a new function.
Within a Client/Server RMMPlex now different versions
of transferred records are sent and received.

Problem

Conclusion:  KEYWORDS: R10COEXS/K

Temporary Fix:

Comments:

Modules/Macros: EDGACTIO EDGAMGR EDGAUDA EDGAUTH EDGBKUP EDGCNVT EDGVCTL
  EDGCVOL EDGDATE EDGDFHSM EDGINERS EDGIPDA EDGIRIP EDGLCSUX
  EDGMAIN EDGMCATX EDGMCMD EDGMDIO EDGMFCNV EDGMFEXP EDGMFIC
  EDGMFICB EDGMFIO EDGMFIS EDGMHKP EDGMLCS EDGMTSO EDGMUPD
  EDGMOVREC EDGMWTO EDGQMGR EDGRACF EDGRCSR EDGRPTX EDGSOCE
  EDGTSO EDGTSOCL EDGUTIL EDGVRECM EDGVRECO EDGXSF REDGXSFB EDGXSF
  EDGXSTX

SRLS:  NONE

Return Codes:

Circumvention:
Message To Submitter:

APAR OA15698

Summary Page
APAR Identifier: OA15698  Last Changed: 06/11/03
Symptom: Status: CLOSED UR1
Severity: 4 Date Closed: 06/10/12
Component: 5695DF186 Duplicate of:
Reported Release: 1J0 Fixed Release:
Component Name: DFSMSRMM Special Notice:
Customer Name: SJ2005 Customer #: 9999999 00
Current Target Date: 06/10/27 Flags: SPEC/ATTN/Y NEW FUNCTION XSYSTEM
SCP:
Platform:
Status Detail: SHIPMENT
Fix Page
PTF List:

  PTF List:  Release 1JB : UA90315 available 06/10/26 (F610)
  Release 1JC : UA90316 available 06/10/26 (F610)
  Release 1J0 : UA90314 available 06/10/26 (F610)
  Release 1KB : UA90318 available 06/10/26 (F610)
  Release 1KC : UA90319 available 06/10/26 (F610)
  Release 1KO : UA90317 available 06/10/26 (F610)

Parent APAR: OA15685

Child APAR List:

Submitter Page
Error.
Description: NEW FUNCTION

Local Fix: NEW FUNCTION

Responder Page
Problem
Summary: ****************************************************************************
  * USERS AFFECTED: This APAR is part of the full function support for the enhanced D/T3592 Model E05 encryption support. *
  * For full D/T3592 Model E05 support, install *
enabling APAR OA15685 (z/OS V1R6 and V1R7).  

In addition the APAR provides the toleration for future changes in volume and data set records.

PROBLEM DESCRIPTION: New Function Support for D/T3592 Model E05 encryption.

RECOMMENDATION: Planned availability of the D/T3592 Model E05 encryption support for

z/OS V1R8 - 11/2006 (Enabling APAR OA17562)

z/OS V1R4 and z/OS V1R5 - 02/2007 (Enabling APAR OA18111)

See z/OS DFSMS Software Support for IBM TotalStorage Enterprise Tape System 3592" (SC26-7514-03), "Chapter 9: DFSMSrmm modifications for the encryption-capable TS1120 tape drive".

Problem  
Conclusion:  
Temporary Fix:  
Comments: For more information see EDGDOC / EDGDOCS, and for complete documentation see "z/OS DFSMS Software Support for IBM TotalStorage Enterprise Tape System 3592" (SC26-7514-03), "Chapter 9: DFSMSrmm modifications for the encryption-capable TS1120 tape drive".

Modules/Macros: EDGACTIO EDGAMGR EDGAPISR EDGAUDA EDGAUTH EDGBKUP EDGCDYNM  
EDGCEBIN EDGCLREC EDGCMOS EDGCMO1 EDGCNVT EDGCPIC2 EDGCTDSI  
EDGCVCTL EDGCVOVL EDGC4LDR EDG5LDR EDGDATE EDGFHSM EDGDOC  
EDGDOCS EDGEXTSY EDGHSKP EDGHTO10 EDGHTO34 EDGHTO44 EDGINERS  
EDGPUA EDGIRIP EDGI100 EDGLAB EDGLCSUX EDGLMINF EDGMAIN  
EDGMCATX EDGMCMD EDGMDIO EDGMDREC EDGMFCNV EDGMFCXP EDGMFIC  
EDGMFIO EDGMFIS EDGMHKP EDGMLAB EDGMLCS EDGMSGXC EDGMTAB  
EDGTM50 EDGTMUD EDGVMRCE EDGWMTO EDGVOEOV EDGVCEOM EDGCRT  
EDGQMGR EDGRCACF EDGRCCVR EDGRMMAV EDGRMMCV EDGRMMSV EDGRPTD  
EDGPRPTX EDGVRVEXT EDGRXEXT EDGDSREC EDGSMFSY EDGSOCE EDG5SE02  
EDG5SE10 EDG5SE12 EDG5SE13 EDG5SE14 EDG5SSSM EDG5VMREC EDG5XMC  
EDG5X3FE EDGTSO EDGTSOCL EDGTO3 EDGUTIL EDGVREC EDGVRECH  
EDGVREC01 EDGVRECE EDGVRECM EDGVREC0 EDGWTO EDGWTOA EDGWTOX  
EDGXXCALL EDGXXINT EDGXXMEL EDGXXMPL EDGXMVP EDGXRP15
Appendix A. APARs to be reviewed

APAR OA17574

Summary Page
APAR Identifier: OA17574 Last Changed: 06/12/02
Symptom: Status: CLOSED UR1
Severity: 4 Date Closed: 06/10/25
Component: 5695DF186 Duplicate of:
Reported Release: 180 Fixed Release:
Component Name: DFSMSRMM Special Notice:
Customer Name: SJ2005 Customer #: 9999999 00
Current Target Date: 06/12/14 Flags: SPEC/ATTN/Y NEW FUNCTION XSYSTEM
SCP:
Platform:
Status Detail: SHIPMENT
' Fix Page
PE PTF List:

PTF List: Release 18E : UA90321 available 06/11/09 (F611)
Release 18K : UA90322 available 06/11/09 (F611)
Release 180 : UA90320 available 06/11/09 (F611)

Parent APAR: OA15685

Child APAR List: OA18782

' Submitter Page
Error.
Description: NEW FUNCTION

Local Fix: NEW FUNCTION
**Responder Page**

**Problem**

**Summary:**

* USERS AFFECTED: This APAR is part of the full function support for the enhanced D/T3592 Model E05 encryption support.

* For full D/T3592 Model E05 support, install enabling APAR OA17562 (z/OS V1R8).

* In addition the APAR provides the toleration for future changes in volume and data set records.

* PROBLEM DESCRIPTION: New Function Support for D/T3592 Model E05 encryption.

* RECOMMENDATION: Planned availability of the D/T3592 Model E05 encryption support for:

  * z/OS V1R4 and z/OS V1R5 - 02/2007 (Enabling APAR OA18111)

  * z/OS V1R6 and z/OS V1R7 - 10/2006 (Enabling APAR OA15685)

* See z/OS DFSMS Software Support for IBM System Storage TS1120 Tape Drive (3592) (SC26-7514-03) for details on the software support for D/T3592 Model E05 encryption.

This APAR is part of the full function support for the new Hardware Encryption support of the D/T3592 Model E05. DFSMSrmm supports the key encryption key label and key encryption encoding mechanism, and the new recording format EEFMT2.

**Problem**

**Conclusion:**

**Temporary Fix:**

**Comments:** ~**** AE06/10/31 FIX IN ERROR. SEE APAR OA18782 FOR DESCRIPTION

**Modules/Macros:**

EDGACTIO EDGAMGR EDGAPISR EDGAUDA EDGAUTH EDGBKUP EDGCDYNM EDGCEBIN EDGCLREC EDGCMM01 EDGMM01 EDGCNVT EDGCFTDSI EDGCVOL EDGC5LDR EDGDATE EDGFHSM EDGDOC EDGDOCS EDGEXTSY EDGHSKP EDGHT034 EDGHT044 EDGINERS EDGIPDA EDGIRIP EDGIK100 EDGLAB EDGLCSUX EDGLMINF EDGMAIN EDGMCATX EDGMCMD EDGMDOIO EDGMDDREC EDGMCNV EDGMPFEXP EDGMPFIC EDGMPFI EDGMPFS EDGMHKP EDGMLAB EDGMLCS EDGMSGXC EDGMTAB EDGMTSO EDGMUPD EDGVREC EDGMWTO EDGOCFEO EDGOECM EDGPCRT EDGQMGRT EDGRACF EDGRCVR EDGRMMAV
EDGRMMCV EDGRMMSV EDGRPTD EDGRPTX EDGRVEXT EDGRXEXT EDGSDREC
EDGSMPSY EDGSOCE EDGSSE02 EDGSSE10 EDGSSE12 EDGSSE13 EDGSSE14
EDGSSSM EDGSVREC EDGSB3FE EDGTSO EDGTSOCL EDGTO3 EDGUTIL
EDGVRECB EDGVRECH EDGVRECI EDGVRECK EDGVRECM EDGVRECO EDGWTO
EDGUTOWA EDGWTOX EDGXMLAP EDGXMLAP EDGXMOV
EDGXRTPX EDGXRP15 EDGXRP35 EDGXSF EDGXSB EDGXST EDGXSTB EDGXSTUB
EDGXSTUB EDG3X71 IGDRMM00

SRLS:  NONE

Return Codes:

Circumvention:

Message To
Submitter:

Label anomaly processing
APAR OA18455 was created to prevent DFSMSrmm from setting the PENDING RELEASE
ACTION INIT flag, when a "servo track format error" is detected. The tape is not
blocked anymore by DFSMSrmm, with the message as shown in Figure 8-72.

EDG4033I VOLUME xxxxxx REJECTED. THE VOLUME IS WAITING TO BE REINITIALIZED.

EDG4033I Message
(is still described in 7617V9RMM.FM).

APAR OA18455

Summary Page
APAR Identifier: OA18455 Last Changed: 06/12/02
Symptom: INTEGRITY Status: CLOSED UR1
Severity: 2 Date Closed: 06/11/09
Component: 5695DF186 Duplicate of:
Reported Release: 1J0 Fixed Release:
Component Name: DFSMSRMM Special Notice:
Customer Name: Customer #: 9999990 72
Current Target Date: 06/12/19 Flags:
SCP: Platform:
Status Detail: SHIPMENT Fix Page
PE PTF List:

PTF List: Release 1J0 : UA30637 available 06/11/18 (F611)
Release 1K0 : UA30638 available 06/11/18 (F611)
Release 180 : UA30639 available 06/11/18 (F611)
HSM OA22804:
Supports VCC keywords and support of recall/recover on pre z/OS V1.10 system of an EAS data set with FMT 8 DSCB that was migrated or backed up on z/OS V1.10. Without this support, if you are using HSM to manage EAV volumes on z/OS V1.10, then on the downlevel system, if you attempt to recall or recover an EAS data set with a format 8 that was migrated or backed up on z/OS V1.10, the function will fail. This will directly impact recall, which is done implicitly. It will also impact recycle and audit. For VCC, keywords specified in the management class will not be processed correctly. (If VPREF or VREQ is specified in the management class, data set will be backed up using standard I/O method without this coexistence APAR.
Information added in '76171HSM.FM'.

APAR OA22804

Summary Page
APAR Identifier: OA22804 Last Changed: 08/05/02
Symptom: TOLERATION Status: CLOSED UR1
Severity: 4 Date Closed: 08/04/01
Component: 5695DF170 Duplicate of:
Reported Release: 1K0 Fixed Release:
Component Name: DFSMSHSM, ISMF Special Notice:
Customer Name: Customer #: 9999909 00
Current Target Date: 08/04/17 Flags: SPEC/ATTN/Y XSYSTEM
SCP:
Platform:
Status Detail: SHIPMENT
' Fix Page
PE PTF List:

PTF List: Release 1J0 : UA40303 available 08/04/08 (F804)
Release 1K0 : UA40304 available 08/04/08 (F804)
Release 180 : UA40305 available 08/04/08 (F804)
Release 190 : UA40306 available 08/04/08 (F804)

Parent APAR:

Child APAR List:
Appendix A. APARs to be reviewed

Submitter Page
Error.
Description: NEW FUNCTION: COEXISTENCE FOR DFSMSHSM EXTENDED ADDRESS VOLUME (EAV) SUPPORT

Local Fix:

Responder Page
Problem
Summary: ****************************************************************
* USERS AFFECTED: All users of z/OS DFSMSHsm in HSMplex         *
* environment with V1R10 DFSMSHsm and                          *
* pre-V1R10 DFSMSHsm levels.                                    *
****************************************************************
* PROBLEM DESCRIPTION: This APAR provides Extended Address     *
*                      Volume (EAV) coexistence support for       *
*                      pre-V1R10 DFSMSHsm versions.              *
****************************************************************
* RECOMMENDATION:                                              *
****************************************************************
This APAR provides coexistence support of the pre-V1R10
DFSMShsm versions to:
- restore EAS eligible data sets from a backup or migrated
copy on an online volume.

- enable pre-V1R10 systems to process the new VCC keywords
  that may be specified in the Management Class Backup Copy
  Technique field.

- allow pre-V1R10 systems to recognize MCV records for EAVs

- restrict the selection of Extent Reduction request
  from the CRQ, if the request is directed to an EAV.

- to issue the new ARC0784I message and update FSR record
  if Format 9 DSCB vendor attributes are lost during
  the RECALL/RECOVER/ARECOVER function.

Note:
EAV volumes will not be accessible or able to be
ADDVOLed on lower level (pre-V1R10) systems.
Errors which can still occur in a mixed
environment with the coexistence installed
are described below:
- Recall of a data set, who's
  migration copy is allocated on
  ML1/ML2 EAV will fail.
- Recall of a data set, who's migration
  copy is in a SDSF that resides
  on a ML1 EAV will fail.
- Recover of a data set, who's backup
copy is allocated on a ML1 EAV will fail.
- Restore from dump of a data set will fail on the pre-V1R10 systems if the VTOC copy is allocated on ML1 EAV.

Problem
Conclusion: KEYWORDS: R10COEXS/K

Temporary Fix:

Comments: SUPPORT
The pre-V1R10 versions of z/OS DFSMShsm can coexist in HSMplex with z/OS DFSMShsm V1R10. In this situation the EAV will be inaccessible for the pre-V1R10 systems since they will be offline. Data sets which have been migrated or backed up from an EAV to ML1, ML2 and Backup volumes (non-EAV) under z/OS DFSMShsm V1R10, will be accessible for the pre-V1R10 DFSMShsm systems. This toleration APAR is required to support Recall, Recover and Recycle for these data sets in the pre-V1R10 systems. If toleration APAR is not installed, the Recall, Recover and Recycle of these data sets will fail during pre-V1R10 DFSMShsm processing.
AUDIT and RECYCLE are changed to process data set CDDs with Format 8 DSCB.

Coexistence is added for pre-V1R10 systems in case vendor attributes of the DSCB are lost during the RECALL/RECOVER/ARECOVER function. If they are lost then:
- the new ARC0784I message will be issued.
- FSR record will be updated.

Support is provided for enabling pre-V1R10 systems to process the new VCC keywords that may be specified in the Management Class Backup Copy Technique field.
Coexistence is added for pre-V1R10 systems to understand the new MCV records for EAV volumes. These fields will be ported from V1R10 systems to the pre-V1R10 systems.

RESTRICTIONS
If a recall for Extent Reduction is directed to an EAV, the selection of this request from the CRQ will be restricted in the pre-V1R10 systems.

Use of EAV volumes in mixed environments where the volume cannot be brought online to all systems can also result in problems recalling/recovering data sets that belong to a Storage Class having the Guaranteed Space attribute.

RECOMMENDATIONS
If you need to recover a data set backup made from a SMS EAV volume to a lower level system then prior to the recover ensure the following:
- if the EAV is still available on the higher level system delete or rename the data set there
OR:
- if the EAV is no longer available and a catalog entry exists then use IDCAMS DELETE NOSCRATCH to remove the data set from the catalog.

NOTE: The backup copy must reside on a non-EAV and the volume must be online to the lower level system.

Modules/Macros:  ARCASDAT ARCAUDDP ARCBDSN ARCCATBU ARCCPLM2 ARCFDVTE ARCFSR
ARCGRDSP ARGRVDS ARCGVDS ARCGRENT ARCRJSML ARCKMANG ARCMCK
ARCMDS ARCMSE ARCRDS ARCRDSSP ARCGGUI ARCRGUI ARCRVDS
ARCBBELG ARGCWVT AARCWCON ARCDWNUM ARCTRFS ARCRMK
ARCNSMG ARCNSE ARCRCUIM ARCRGUIS ARCRVDS
ARCSBELG ARCWCVT ARCCWCON ARCDWUI ARCTRFS ARCRMK
ARCWVSMM ARCTRDS

SRLS:  SC35042104 SA22763210

Return Codes:

Circumvention:

Message To
Submitter:

DSS OA22900:

DFSMSdss full volume and tracks dumps of EAVs are not compatible with dumps of volumes that are 64 K cylinders or fewer due to changes required to format the extended-address space in the dump. Changes have been made in z/OS V1R10 DFSMSdss to identify dumps of EAVs. Since many customers run more than one level of z/OS in their environments or at times must provide data in the form of DFSMSdss dumps to out-centers that are not at the same z/OS level. DFSMSdss provides limited restore capability on supported lower levels of z/OS for data dumped from an EAV on z/OS V1R10 or higher levels.

The following describes the restore functions and limitations that will be supported via a coexistence APAR on pre-z/OS V1R10 system. Versions of DFSMSdss prior to V1R10 will not perform a full volume restore of a full volume dump from an EAV. In addition, tracks restore where track 0 is included will fail and tracks restore, not including track 0, will restore only the track-managed space from an EAV. Logical data set restore and physical data set restore will be changed to convert F8/F9 DSCBs to F1 DSCBs. When a data set is restored that had an F8/F9 pair when it was dumped, if attributes are being lost due to the inability to restore the F9, a new message, ADT556W, will be issued. If no attribute values exist, no message will be issued.
APAR OA22900

Summary Page
APAR Identifier: OA22900 Last Changed: 08/08/13
Symptom: NEW FUNCTION Status: CLOSED UR1
Severity: 4 Date Closed: 08/03/28
Component: 5695DF175 Duplicate of:
Reported Release: 1K Fixed Release:
Component Name: DFSMSdss, ISMF Special Notice:
Customer Name: SJ2005 Customer #: 99999999
Current Target Date: 08/06/28 Flags: SPEC/ATTN/Y XSYSTEM
SCP:
Platform:
Status Detail: SHIPMENT

Fix Page
PE PTF List:

PTF List: Release 1J0 : UA40263 available 08/04/03 (F804)
Release 1K0 : UA40264 available 08/04/03 (F804)
Release 180 : UA40265 available 08/04/03 (F804)
Release 190 : UA40266 available 08/04/03 (F804)

Parent APAR: OA21487

Child APAR List: OA24505

Submitter Page
Error.
Description: NEW FUNCTION COEXISTENCE SUPPORT

Local Fix:

Responder Page
Problem
Summary: *****************************************
* USERS AFFECTED: All users of DFSMSdss 1K, 180, and 190 *
* attempting to restore data that was dumped *
* from an extended address volume. *
*****************************************

* PROBLEM DESCRIPTION: This APAR provides toleration support *
* when attempting to restore data from a *
* DFSMSdss dump data set containing data *
* from an extended address volume. *

*****************************************
* RECOMMENDATION: *
*****************************************
Problem
Conclusion: KEYWORDS: R10COEXS/K

Temporary Fix:

Comments: DFSMSdss 1K0, 180, and 190 are modified to restore data from a
DFSMSdss dump data set containing data from an extended address
volume (EAV).

Restore of data sets dumped from an EAV with physical or logical
data set is supported. Track restore of data dumped from the
track-managed space of an EAV using physical dump processing
is supported.

When a data set that was dumped from
an EAV which has F8 and F9 DSCBs is restored with physical
or logical data set processing and vendor attributes existed in
the F9 DSCB, a MSGADR566W will be issued with reason code 1.

Full volume restore to a nonEAV will fail since the EAV is
larger than a nonEAV; MSGADR309E will be issued. Physical
tracks restore attempting to process tracks dumped from the
cylinder-managed space of an EAV will fail. One of the
following messages will be issued:
MSGADR024E is issued when the track address specified is
within the range of 65520 and 65535
MSGADR136E is issued when the track address specified is
great than 65535.

Documentation Updates to "z/OS V1R9.0 MVS System Messages,
Vol 1 (ABA-AOM)" (SA22-7631)

1. Message ADR309E needs to be updated to contain the
larger capacity of an EAV.

ADR309E (xxx)-mmmmm(yy), SOURCE AND TARGET DEVICE CAPCITIES
DO NOT MATCH. CYLINDER CAPACITY OF SOURCE
VOLUME nnnnnnnn, TARGET VOLUME nnnnnnnn

EXPLANATION: The capacity of the source volume is greater
than that of the target volume. The nnnnnnnn is the highest
cylinder number in hexadecimal.

2. New message ADR566W needs to be added. The message text
of ADR566W is as follows:

ADR566W (xxx)-mmmmm(yy), EXTENDED ATTRIBUTES FOR DATA SET
dname WERE NOT RETAINED DURING RESTORE, 1

EXPLANATION: The data set was restored but some extended
attributes were lost for the following reason (RSN):

1 Vendor attributes from the F9 DSCB of the dumped
data set were not retained for the target data set because the volume on which it was placed did not support F8/F9 DSCBs.

~~~ PE08/07/11 FIX IN ERROR. SEE APAR OA25824 FOR DESCRIPTION
~~~ PE08/07/21 FIX IN ERROR. SEE APAR OA25872 FOR DESCRIPTION

Modules/Macros: ADRACS ADRADDXT ADRASIZE ADRAUTH ADRBMB ADRCLNB ADRCNVSM ADRDITFP ADRDEBUP ADRDEVSU ADRDIVSM ADRDSVSM ADRDTDS ADRDTDSC ADRFRAG ADRIO1D ADRINPU ADREXTND ADRFDSOC ADRFDSOC ADRFDSRL ADRFDSRS ADRFILT A DRFILTV ADRFT4B ADRFTLB0 ADRGBUFF ADRGTDIR ADRINL ADRIO8B ADRLD ADRIORT ADRIOWT ADRLSPAC ADRMI0 ADRMRDFM ADRMRMIO ADRMRRFM ADRMRSGS ADRPNCL ADRPRNT ADRRALLC ADRRLE0 ADRSBRTN ADRSB004 ADRSB005 ADRSB100 ADRSB105 ADRSB111 ADRSDM0I ADRSDMX0 ADRSETUP ADRSRPB ADRTAB ADRTDOS ADRTFIP ADRTDLOG ADRTDNVS ADRTDPV ADRTDUNL ADRTDVSM ADRUIMB ADRUIMXT ADRUPKEY ADRVOLCK ADRVSBK2 ADRVSBKL ADRVTRLB0 ADRZBLK2 ADRZINIT ADRZOUT ADRZSCH ADRZFYO ADR4T3B

SRLS: GA22749900

Return Codes:

Circumvention:

Message To Submitter:

APAR PK58292

Summary Page
APAR Identifier: PK58292 Last Changed: 08/06/02
Symptom: NF Status: CLOSED UR1
Severity: 3 Date Closed: 08/04/25
Component: 5740XYR00 Duplicate of:
Reported Release: 810 Fixed Release:
Component Name: DB2 OS/390 & Z/ Special Notice:
Customer Name: IBM INTERNAL - Customer #: 9999999 00
Current Target Date: 08/09/30 Flags:
SCP:
Platform:
Status Detail: SHIPMENT
' Fix Page
PE PTF List:

PTF List: Release 810 : UK35901 available 08/05/13 (F805)
Release 910 : UK35902 available 08/05/13 (F805)
Parent APAR:

Child APAR List:

  ' Submitter Page
Error.
Description: New function

Local Fix:

Responder Page
Problem
Summary: **************************************************************
* USERS AFFECTED: All DB2 users.                              *
**************************************************************
* PROBLEM DESCRIPTION: New function.                          *
**************************************************************
* RECOMMENDATION:                                             *
**************************************************************
New function.

Problem
Conclusion:

Temporary Fix:

Comments: New function.

Modules/Macros: DSNDQJST DSNJDOS04 DSNJDOS06 DSNJLOGF DSNJRS03 DSNJR008 DSNJW001

SRLS:  NONE

Return Codes:

Circumvention:

Message To
Submitter:
APAR OA22449

Summary Page
APAR Identifier: OA22449 Last Changed: 08/05/02
Symptom: NEW FUNCTION Status: CLOSED UR1
Severity: 4 Date Closed: 08/03/26
Component: 5695DF133 Duplicate of:
Reported Release: 1K0 Fixed Release:
Component Name: DEVICE MGMT SER Special Notice:
Customer Name: SJ2005 Customer #: 9999998 99
Current Target Date: 08/03/20 Flags: SPEC/ATTN/Y XSYSTEM
SCP:
Platform:
Status Detail: SHIPMENT
' Fix Page
PE PTF List:
PTF List: Release 1K0 : UA40219 available 08/04/09 (F804)
Release 180 : UA40220 available 08/04/09 (F804)
Release 190 : UA40221 available 08/04/09 (F804)

Parent APAR: OA21487

Child APAR List:

' Submitter Page
Error.
Description: EXTENDED ADDRESS VOLUME (EAV) COEXISTENCE SUPPORT FOR LSPACE

Local Fix:

Responder Page
Problem
Summary: *******************************************************
* USERS AFFECTED: DFSMS z/OS V1R10 ( for EAV support) when  *
* the V1R10 LSPACE macro is assembled and  *
* the resulting code is then executed on  *
* earlier releases ( HDZ11K0 HDZ1180 HDZ1190 ) *
***************************************************************
* PROBLEM DESCRIPTION: This APAR adds support to DFSMS z/OS  *
* V1R7, V1R8, and V1R9 to allow these  *
* releases to map V1R10 LSPACE parameters  *
* XEXPMSEG and EXPDATA to  *
* EXPMSG and DATA respectively.  *
* *
* Note that the V1R10 LSPACE parameter  *
* DATATYPE is ignored on V1R7, V1R8,  *
* and V1R9.  *
* *
* If this APAR is not applied, then  *

406 DFSMS V1.10 and EAV Technical Guide
Programs that are assembled using the DFSMS z/OS V1R10 LSPACE macro using the new keywords (XEXPMSG and EXPDATA) and then executed on prior releases of DFSMS will get an LSPACE error unless this coexistence APAR has been applied to these prior releases.

Problem

Conclusion: KEYWORDS: R10COEXS/K

Temporary Fix:

Comments:

Modules/Macros: IECLSPWA IGC0007H IGC0107H IGC0207H

SRLS: NONE

Return Codes:

Circumvention:

Message To

Submitter:

The 3390 model A support was part of the DS8000 R3 SPE apar. This support enabled DEVSERV to display '3390A' instead of just '3390'. The support was additionally retrofitted back to z/OS V1R7 through these PTF's (available 11/19/2007):

UA38011 5695DF111 - 1K0(z/OS 1.7)
UA38012 5695DF111 - 180 (z/OS 1.8)
APAR OA22903

Summary Page
APAR Identifier: OA22903 Last Changed: 07/12/16
Symptom: IN INCORROUT Status: CLOSED PER
Severity: 3 Date Closed: 07/11/29
Component: 5695DF103 Duplicate of:
Reported Release: 180 Fixed Release: 999
Component Name: DFSMS IDCAMS Special Notice:
Customer Name: COCA-COLA ENTERPRISES INC Customer #: 3903936 00
Current Target Date: 08/01/02 Flags:
SCP:
Platform:
Status Detail: SHIPMENT
PE PTF List:

PTF List: Release 180 : PTF not available yet
Release 190 : PTF not available yet

Parent APAR:

Child APAR List:

Submitter Page
Error.
Description: Issue a LISTC ALL command using IDCNOGFL for z/OS 1.8 results in no PageSpace values. In this case, a pagespace value was equal to zero.

Local Fix: none

Responder Page
Problem
Summary: *******************************************************
* USERS AFFECTED: All users, Level 180 or higher. *
*****************************************************************
* PROBLEM DESCRIPTION: With this support, the non-CSI calls *
* for LISTCAT made mostly through either *
* ENT or IDCNOGFL entry point calls *
* will seek a PAGESPACE as a separate *
* object than a CLUSTER. This will *
* ensure retrieval of PAGESPACE objects *
* for non-CSI-type calls. *

408  DFSMS V1.10 and EAV Technical Guide
PAGESPACE objects with this support will be retrieved as separate objects for non-CSI, non-filter-key type calls invoked either as LISTCAT ENT with present interface, or with IDCNOGFL entry point. This will ensure that a PAGESPACE is retrieved.

Problem

Conclusion: PAGESPACE objects, with this support, will be retrieved and reported on when the end-user does not specify the new CSI filter-key method of LISTCAT retrieval. This will ensure that PAGESPACEs will appear on LISTCAT reports whether the new CSI filter-key retrieval method is used or the ENT retrieval or IDCNOGFL entry point is selected.

Temporary Fix: The non-GFL call needs to specify the PAGESPACE as a P type object and not a C or CLUSTER object, as GFL retrieves it. This fix allows the non-GFL to retrieve the PAGESPACE as a P type object.

Comments:

Modules/Macros: IDCLC01

SRLS: NONE

Return Codes:

Circumvention: The circumvention is to ensure that the non-CSI filter-key retrievals in LISTCAT retrieve PAGESPACE objects as separate objects. With this support, those objects are retrieved.

Message To
Submitter:

APAR OA22625

Summary Page
APAR Identifier: OA22625 Last Changed: 08/02/01
Error.
Description: In some cases the message IDC2950I may be issued when running an IDCAMS LISTCAT LEVEL VOL command. The condition code is set to 12. This occurs when using the entry point IDCNOGFL. When using the IDCAMS entry point the messages IDC3012I, IDC3007I and IDC1566I are issued.

Local Fix: None
Problem
Conclusion: LISTCAT LVL with a Level Name and a VOL, rather than an ALL option, will return a CC=08 and note the ALIAS object is in error.

Temporary Fix: The solution is that the final return code for the non-gfl/csiinterface, when returning to the main program, must be zero.

Comments:

Modules/Macros: IDCLC05

SRLS: NONE

Return Codes:

Circumvention: Circumvention is the APAR support described, which ensures CC=08 upon completion of the step.

Message To Submitter:
Parent APAR:

Child APAR List:

'  Submitter Page
Error.
Description: Issuing LISTCAT LEVEL GDG ALL for a GDG that has no Generation
Data Sets (GDS's) returns blanks for the Association rather than
-------(NULL) .

Local Fix: Use the Entry Point IDCNOGFL provided in apar OA20472

Responder Page
Problem
Summary: ************************************************************
* USERS AFFECTED: All users, Level 180 of code or higher.  *
***************************************************************
* PROBLEM DESCRIPTION: When doing a LISTCAT with GFL/CSI      *
*                 facility, an empty GDG BASE does not   *
*                 have a (NULL) item printed when it has    *
*                 no associations. With this support,     *
*                 however, the (NULL) will print.         *
***************************************************************
* RECOMMENDATION:                                           *
***************************************************************
LISTCAT of a GDG BASE with no associations, with this support,
will print a (NULL) indicating the fact that the
associations list is null.

Problem
Conclusion: With this APAR support, a GDG BASE that contains no GDS
associations will note this fact by printing (NULL)
when displaying the associations list.

Temporary Fix: The fix is to ensure that the NULL message will hit when the GDG
base object is checked for associations on LISTCAT VOL or ALL
listings.

Comments:

Modules/Macros: IDCLC08

SRLS: NONE

Return Codes:
Circumvention: The circumvention is the solution here. When a GDG BASE object has a null associations list, with this APAR support, a (NULL) will be printed.

Message To Submitter:

APAR OA22632

Summary Page
APAR Identifier: OA22632 Last Changed: 07/12/03
Symptom: IN INCORROUT Status: CLOSED PER
Severity: 3 Date Closed: 07/11/08
Component: 5695DF103 Duplicate of:
Reported Release: 180 Fixed Release: 999
Component Name: DFSMS IDCAMS Special Notice:
Customer Name: HVB INFORMATION Customer #: 0315790 72
Current Target Date: 07/12/05 Flags:
SCP:
Platform:
Status Detail: SHIPMENT
Fix Page
PE PTF List:

PTF List: Release 180 : UA37935 available 07/11/27 (F711)
Release 190 : UA37936 available 07/11/27 (F711)

Parent APAR:

Child APAR List:

† Submitter Page
Error.
Description: LISTCAT CATALOG NONVSAM does not include GDG's. It includes NONVSAM data sets and generation data sets (GDS's), but the GDG base is not listed.

Local Fix: Use new entry point IDCNOGFL provided in apar OA20472

Responder Page
Problem Summary: ****************************************************************
* USERS AFFECTED: All users, Level 180 or higher. *
****************************************************************
* PROBLEM DESCRIPTION: A LISTCAT of a UCAT with a NVSAM type-restriction should show both a GDG Base object as well as GDSes and all non-VSAM Alien objects. With this support, that is what this listing shows.  

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

* RECOMMENDATION: NVSAM type-restricted reports should show GDG base, NVSAM Alien, and GDSes for a listing of a catalog. With this support, these objects show.

Problem

Conclusion: NVSAM type-restricted LISTCAT of a catalog should show GDG Bases, GDSes, and all NVSAM Alien objects. With this support, these reports do show these objects.

Temporary Fix: LISTCAT NONVSAM includes GBS (GDG BASE DATA SETS) with this support.

Comments:

Modules/Macros: IDCLC01 IDCLC05 IDCLC06

SRLS: NONE

Return Codes:

Circumvention: The fix is to ensure that NVSAM type-restricted LISTCAT of a catalog shows GDG Base, GDSes, and any NVSAM alien objects.

Message To Submitter:

APAR OA22542

Summary Page

APAR Identifier: OA22542 Last Changed: 08/07/24
Symptom: IN INCORROUT Status: OPEN
Severity: 3 Date Closed:
Component: 5695DF103 Duplicate of:
Reported Release: 180 Fixed Release:
Component Name: DFSMS IDCAMS Special Notice:
Customer Name:  KMD A/S, BALLERUPCENTR  Customer #:  0475000 67
Current Target Date:  08/09/30  Flags:  HIPER/Y DATALOSS FUNCTIONLOSS PERVERSIVE
SCP:
Platform:
Status Detail:  DESIGN/CODE

' Fix Page
PE PTF List:

PTF List:

Parent APAR:

Child APAR List:

' Submitter Page
Error.
Description:  LISTCAT CATALOG ALL may end with a return code 4 and not list all the entries in the catalog. Catalogs with a large number of GDG's may show this symptom. Also, LISTCAT LEVEL NVSAM ALL may not return any entries depending on the number of GDS's for a GDG.

Local Fix:  Use the alternate entry point (IDCNOGFL) supplied in apar OA20472. Review additional information documented in apar III4250 for updates.

A new target date July 29 has been entered to accommodate additional commitments from internal test teams. ++APARs may also be updated with new versions during this time.

Responder Page
Problem
Summary:

Problem
Conclusion:

Temporary Fix:

Comments:

Modules/Macros:

SRLS:

Return Codes:

Circumvention:

Message To
Submitter:

APAR OA22078

Summary Page
APAR Identifier: OA22078 Last Changed: 07/11/19
Symptom: NEWFUNCTION Status: CLOSED PER
Severity: 3 Date Closed: 07/09/10
Component: 5695DF103 Duplicate of:
Reported Release: 1J0 Fixed Release: 999
Component Name: DFSMS IDCAMS Special Notice:
Customer Name: IBM INTERNAL Customer #: 9999999 84
Current Target Date: 07/12/05 Flags:
SCP:
Platform:
Status Detail: SHIPMENT
Fix Page
PE PTF List:

PTF List: Release 1J0 : UA36781 available 07/10/27 (F710)
Release 1K0 : UA36782 available 07/10/27 (F710)

Parent APAR:

Child APAR List:

Submitter Page
Error.
Description: In z/os 1.8 LISTC LVL was changed to be a higher performance
GFL type call. This type of call has limitations and some of
these limitations prompted users to request access to the older
form of LISTC LVL. This is being added in apar OA20472 and
will allow clients to use a new alias for IDCAMS called IDCNOGFL
which can be linked/called in z/os 1.8.

For toleration of all supported releases, we need to make this
alias available on z/os 1.6 and z/os 1.7

Local Fix:

Responder Page
Problem
Summary: ************************************************************
* USERS AFFECTED: All DFSMS release R1.6 and R1.7 affect.  *
***************************************************************
* PROBLEM DESCRIPTION: Apar OA20472 provided the support that  *

enables IDCNOGFL as an option on AMS
LISTCAT command at release 1.8 and higher, so that users can select how the LISTCAT command performs. It can be picked to have an improved performance via the call of CSI interface (PGM=IDCAM), or it can be picked to list the dataset names in sequential order (PGM=IDCNOGFL).
To be compatible with apar OA20472, R1.6 and R1.7 need to support the option of IDCNOGFL as well.

RECOMMENDATION:
This is toleration apar for OA20472. For R1.6 and R1.7, the specification of PGM=IDCNOGFL performs the same way as that of PGM=IDCAM, because there are no CSI interface support in these 2 releases.

Problem
Conclusion: All releases from R1.6 and higher are compatible for IDCNOGFL option.

Temporary Fix:

Comments: ~**** PE07/11/16 FIX IN ERROR. SEE APAR OA22832 FOR DESCRIPTION

Modules/Macros: HDZ11J0J HDZ11K0J IDCSA01 SGIDC401

SRLS: NONE

Return Codes:

Circumvention:

Message To Submitter:

APAR OA16912

Summary Page
APAR Identifier: OA16912 Last Changed: 07/07/02
Symptom: IN INCORROUT Status: CLOSED PER
Severity: 3 Date Closed: 06/10/13
Component: 5695DF103 Duplicate of:  
Reported Release: 180 Fixed Release: 999  
Component Name: DFSMS IDCAMS Special Notice:  
Customer Name: Customer #: 99999999  
Current Target Date: 06/11/16 Flags:  
SCP:  
Platform:  
Status Detail: TESTPACKAGE

' Fix Page  
PE PTF List:  

PTF List: Release 180 : UA29965 available 06/11/08 (F611)  

Parent APAR:  

Child APAR List: OA18720  

' Submitter Page  
Error.  
Description: ROLL UP APAR FOR IDCAMS FOR 1.8 ROLL-UP  

Local Fix:  
Responder Page  
Problem  
Summary:  
* USERS AFFECTED: All users, Level 1.8 and higher.  
* PROBLEM DESCRIPTION: This APAR represents the roll-up APAR for  
* AMS LISTCAT with GFL.  
* RECOMMENDATION:  
This APAR is the roll-up APAR for AMS LISTCAT with GFL.  

Problem  
Conclusion: This APAR is the roll-up APAR for AMS LISTCAT with GFL.  

Temporary Fix:  

Comments:  

Modules/Macros:  
IDCLC01 IDCLC05 IDCLC06 IDCLC07 IDCLC08 IDCLC09 IDCMP03 IDCTSUYO  

SRLS: NONE  

418 DFSMS V1.10 and EAV Technical Guide
Return Codes:
Circumvention:
Message To Submitter:

**APAR OA18184**

Summary Page
APAR Identifier: OA18184 Last Changed: 07/08/07
Symptom: IN INCORROUT Status: CLOSED PER
Severity: 2 Date Closed: 06/10/19
Component: 5695DF105 Duplicate of:
Reported Release: 180 Fixed Release: 999
Component Name: ICF CATALOG & C Special Notice:
Customer Name: GAD Customer #: PIP0001 72
Current Target Date: 06/11/12 Flags:
SCP:
Platform:
Status Detail: SHIPMENT
' Fix Page
PE PTF List:

PTF List: Release 180 : UA30086 available 06/11/08 (F611)

Parent APAR:

Child APAR List:

' Submitter Page
Error.
Description: IDCAMS LISTCAT does not list error messages for volumes that are offline in z/OS 1.8

Local Fix:

Resonder Page
Problem
Summary: ****************************
* USERS AFFECTED: HDZ1180 users that are using LISTCAT. *
**************************************************************************
* PROBLEM DESCRIPTION: LISTCAT was changed to use GFL for z/OS *
* 1.8. Changes had to be made to the GFL *
* code to allow it to return data for *

Appendix A. APARs to be reviewed 419
volumes which were offline and the data could not be retrieved.

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

RECOMMENDATION:

****************************************************************
The GFL code has been enhanced to process offline volumes and return a return code of RC50 RSN6 for cases where the volumes are offline.

Problem
Conclusion: Processing will now allow offline volumes to return the other data requested for a data set, but a RC50 RSN6 will be issued if the volume is offline.

Temporary Fix: APAR AA18184 is available

Comments: APAR is available.

***** PE07/02/28 FIX IN ERROR. SEE APAR OA20169 FOR DESCRIPTION

Modules/Macros: IGGGFLNE IGGOCLFA IGGOCLFS

SRLS: NONE

Return Codes:

Circumvention:

Message To Submitter:

APAR OA13742

Summary Page
APAR Identifier: OA13742 Last Changed: 07/09/07
Symptom: IN INCORROUT Status: CLOSED PER
Severity: 3 Date Closed: 05/11/30
Component: 56950F175 Duplicate of:
Reported Release: 1J0 Fixed Release: 999
Component Name: DFSMSDSS, ISMF Special Notice:
Customer Name: INTERNATIONAL BUSINESS MACHIN Customer #: 6333328 00
Current Target Date: 05/12/06 Flags:
SCP:
Platform:
Status Detail: SHIPMENT
Appendix A. APARs to be reviewed

Fix Page

PE PTF List:

PTF List:  Release 1G0  : UA22851 available 05/12/01 (F511)
Release 1H0  : UA22852 available 05/12/01 (F511)
Release 1J0  : UA22853 available 05/12/01 (F511)
Release 1K0  : UA22854 available 05/12/01 (F511)

Parent APAR:

Child APAR List:

Submitter Page

Error.
Description:  Tape management systems are unable to determine tape usage when DFSMSdss DUMP or COPYDUMP is run as no blocksize is available in the TAPE LABEL.

Note.  If this issue involves Removable Media Manager (RMM), please review informational apar II08148.  This informational apar covers RMM and DSS as it pertains to the problem described in this apar.
Additional keywords:
  DFSMSdss  DFDS DSS DFSMSrmm DFRMM RMM LARGE BLOCK INTERFACE LBI BLKSIZE VOLUME

Also Note:
If O/C/EOV APAR OA09868 is not applied on system where RESTOREs will be performed, DSS APAR OA13742 must be applied on these systems for 1J0 and below.

Also note II:
When DFSMSdss in invoked via Cross Memory (ADRXMAIA), BLOCKSIZE in the TAPE LABEL will continue to be zero.

Note: DFSMSdss APAR OA21690 for z/OS V1R9 fixed this problem.

Additional Information:
Application of this fix may make users at 1G0 susceptible to receive ABEND806-04 if the PTF for OW53778 is not applied.

Local Fix:

Responder Page

Problem
Summary:  **************************************************
  * USERS AFFECTED: All users of DFSMSdss DUMP and COPYDUMP.  *
  ****************************************
  * PROBLEM DESCRIPTION: DFSMSdss does not provide the  *
  * BLOCKSIZE in the TAPE LABEL for the  *

Appendix A. APARs to be reviewed 421
Problem
Conclusion: DFSMSdss has been changed to correct this problem.

COPYDUMP will only reflect a valid tape label blocksize if the dump data set was created with this APAR applied.

Temporary Fix:

Comments:

Modules/Macros: ADRioBB ADROPNCL

SRLS: NONE

Return Codes:

Circumvention:

Message To Submitter:

APAR OA20169

Summary Page
APAR Identifier: OA20169 Last Changed: 08/01/28
Symptom: PE INCORROUT Status: CLOSED PER
Severity: 2 Date Closed: 07/08/07
Component: 5695DF105 Duplicate of:
Reported Release: 180 Fixed Release: 999
Component Name: ICF CATALOG & C Special Notice: PE
Customer Name: STATE FARM MUTUAL AUTOMOBILEINSURA Customer #: 7826318 00
Current Target Date: 07/08/24 Flags:
SCP:
Platform:
Status Detail: SHIPMENT
' Fix Page
PE PTF List:

PTF List: Release 180 : UA36132 available 07/09/06 (F709)
Parent APAR:

Child APAR List:

' Submitter Page
Error.
Description: Clients running LISTC VOL CAT(mcatname) or LISTC CAT(mcatname) ALL have reported an abendOC4 in IGGOCLFS with UA30086 applied.
.
The code was checking the returned name from the GFL call and abending due to REG2 being zeroes.

Local Fix:

Responder Page
Problem
Summary: ****************************************************************
* USERS AFFECTED: HDZ1180 and above. *
****************************************************************
* PROBLEM DESCRIPTION: Under certain circumstance, a LISTC as *
* indicated in the error description *
* will cause an OC4 because an attempt is *
* made to set a flag based upon a zero *
* pointer. Subsequent code checks for *
* the pointer's existence, but is too *
* late to prevent the error situation. *
****************************************************************
* RECOMMENDATION: *
****************************************************************
The setting of the error flag has been moved down in the code to after the check for the pointer's existence so as to insure addressability.

Problem
Conclusion: Code was corrected to insure addressability to error flag.

Temporary Fix:

Comments: ~**** PE07/11/27 FIX IN ERROR. SEE APAR OA23341 FOR DESCRIPTION

Modules/Macros: IGGOCLFS

SRLS: NONE
Return Codes:

Circumvention:

Message To Submitter:

**APAR OA23595**

Summary Page
APAR Identifier: OA23595 Last Changed: 08/05/02
Symptom: IGD1015I Status: CLOSED PER
Severity: 3 Date Closed: 08/02/11
Component: 5695DF101 Duplicate of:
Reported Release: 1K0 Fixed Release: 999
Component Name: STORAGE MGMT SU Special Notice:
Customer Name: RESORTQWEST Customer #: 0596000 67
Current Target Date: 08/03/18 Flags:
SCP:
Platform:
Status Detail: APARCLOSURE
Fix Page
PE PTF List:

PTF List: Release 190 : UA39360 available 08/04/23 (F804)

Parent APAR:

Child APAR List:

Error.
Description: TSO XMIT/RECEIVE command made a change in z/OS R1.9 that sets DSNTYPE=BASIC for temp work area that was directed to VIO and causes Receive to fail with message:

IGD01015I DATA SET ALLOCATION REQUEST FAILED - THE ACS STORAGE GROUP ROUTINE ASSIGNED A DSNTYPE=BASIC DATA SET TO A STORAGE GROUP WHICH IS NOT OF TYPE POOL

Local Fix: 1) Remove DSNTYPE=BASIC if possible.
2) Direct the allocation to a DASD device.

Responder Page
Problem
Summary: ****************************************************************
* USERS AFFECTED: USERS AT Z/OS390 DFSMS R1.9 AND HIGHER *
* RELEASES .
****************************************************************
* PROBLEM DESCRIPTION: TSO XMIT/RECEIVE FAILS WITH MSG *
* IGD01015I AFTER GOING TO Z/OS390 *
* RELEASE 1.9. *
****************************************************************
* RECOMMENDATION: *
****************************************************************
DSNTYPE=BASIC MEANS TO ALLOCATE A NON-EXTENDED DATA SET. WHICH
IS THE SAME AS THE DEFAULT WHEN DSNTYPE= IS NOT SPECIFIED.
THE PROBLEM OCCURS WHEN TSO XMIT/RECEIVE INTRODUCED CHANGE THAT
SPECIFIED DSNTYPE=BASIC ON THE TEMP WORKAREA AND A VIO
STORAGE GROUP WAS ASSIGNED BY THE INSTALLATION ACS ROUTINE

Problem
Conclusion: ACS WILL BE CHANGED TO ALLOW DSNTYPE=BASIC TO BE VIO ELIGIBLE.

Temporary Fix:

Comments:

Modules/Macros: IGDACSM1 IGDACS01

SRLS: NONE

Return Codes:

Circumvention:

Message To Submitter:

APAR PK48337

Summary Page
APAR Identifier: PK48337 Last Changed: 07/10/02
Symptom: COR Status: CLOSED PER
Severity: 3 Date Closed: 07/08/07
Component: 5655HAL00 Duplicate of:
Reported Release: 180 Fixed Release: 999
Component Name: TCP/IP V3 MVS Special Notice:
Customer Name: PARKER HANNIFIN CORP Customer #: 6914700 00
Current Target Date: 07/09/03 Flags:
SCP:
Platform:
Status Detail: SHIPMENT
' Fix Page
PE PTF List:

PTF List: Release 170 : UK28003 available 07/09/20 (F709)
Release 180 : UK28004 available 07/09/20 (F709)
Release 190 : UK28005 available 07/09/20 (F709)

Parent APAR:

Child APAR List:

' Submitter Page
Error.
Description: The z/OS FTP client and server do not support a data set name type (dsntype) with a format of LARGE (exceeds 65535 tracks). FTP can still access MVS data sets of dsntype=large when the BLOCKTOKENIZE(NOREQUIRE) is coded in the IGDSMxx member. However, access fails when BLOCKTOKENIZE(REQUIRE) is coded.

Local Fix: Code BLOCKTOKENIZE(NOREQUIRE) in your IGDSMxx SYS1.PARMLIB member.

Responder Page
Problem
Summary: ******************************************************
* USERS AFFECTED: All users of the IBM Communications Server *
* for z/OS Version 1 Release(s) 7, 8 and 9               *
* IP: FTP                                              *
******************************************************
* PROBLEM DESCRIPTION: The z/OS FTP client and server do not *
* support a data set name type (dsntype) *
* with a format of LARGE (exceeds 65535 *
* tracks). FTP can still access MVS *
* data sets of dsntype=large when the *
* BLOCKTOKENIZE(NOREQUIRE) is coded in *
* the IGDSMxx member. However, access *
* fails when BLOCKTOKENIZE(REQUIRE) *
* is coded.                                            *
******************************************************
* RECOMMENDATION:                                      *
******************************************************
FTP BSAM open does not specify BLOCKTOKENIZE=LARGE parameter on the DCBE macro, when it opens the data set. BLOCKTOKENIZE=LARGE must be specified if the data set contains more than 65,535 tracks on a volume.
+-------------------------------------------------------------+
+ Please check our Communications Server for OS/390 homepages +
+ for common networking tips and fixes. The URL for these    +
+ homepages can be found in Informational APAR II11334.       +
+------------------------------------------------------------------+

Problem
Conclusion: FTP has been corrected.

* Cross Reference between External and Internal Names

Temporary Fix:

Comments:

Modules/Macros: EZAFTPIO

SRLS: NONE

Return Codes:

Circumvention:

Message To
Submitter:

APAR II14250

Summary Page
APAR Identifier: II14250 Last Changed: 08/04/23
Symptom: INFO Status: INTRAN
Severity: 4 Date Closed:
Component: INFOV2LIB Duplicate of:
Reported Release: 001 Fixed Release:
Component Name: Special Notice:
Customer Name: INFORMATION ITEM Customer #: 9999999 00
Current Target Date: Flags:
SCP:
Platform:
Status Detail:
' Fix Page
PE PTF List:

PTF List:

Parent APAR:
Child APAR List:

Submitter Page

Error.

Description: IDCAMS LISTCAT processing in z/OS 1.8 and higher releases has been enhanced to provide better runtime performance and more diagnostic information. For z/OS 1.8 and 1.9 the PTFs UA38133 for z/OS 1.8 and UA38134 for z/OS 1.9 should be considered mandatory maintenance as they roll up a number of corrections to IDCAMS LISTCAT processing.

-------------------------------------------------------------

z/OS AMS LISTCAT does not support toleration, change controls, or guarantees being provided by formally defined IBM interfaces (API). z/OS AMS LISTCAT output format and contents may change as a result of service and new function support. IBM is not committed to maintain similarities in LISTCAT format between releases, SPEs, and service for automation or user needs.

AMS LISTCAT displays objects which are returned from Catalog:
In z/OS V1R8, LISTCAT was modified to use the Catalog Search Interface (CSI).
The objects returned from CSI are defined by the CSI interface and may produce different results than the prior generic LOCATE interface rules. Results may differ in the return and condition codes or number of objects being returned by Catalog.

Review information for the Catalog Search Interface (CSI) in the z/OS DFSMS Managing Catalogs and HLASM Programmer's Guide to assist in determining expected results and differences.

Catalog provides filters preventing the listing of an object when using generic LOCATE. These filters may differ and not apply to CSI returned objects, values, and rules. IBM will review marketing change requests for the inclusion or exclusion of CSI provided objects by AMS LISTCAT. IBM is not committing to maintain toleration of LISTCAT formats between releases, SPEs, and service.

IBM recommends applications processing LISTCAT output be migrated to IBM supported APIs or tools.
To ease the migration of applications, a fallback APAR OA20472 is provided in z/OS V1R8 and V1R9 for obtaining LISTCAT results as in prior releases. Companion APAR OA22078 provides toleration support in z/OS V1R7 when utilizing fallback support in a PLEX running mixed releases including z/OS V1R7.

Fallback introduces a new entry point PGM=IDCNOGFL. For migration ease, this entry point will remain in all future releases allowing a one time conversion. PGM=IDCNOGFL will not be documented beyond associated APARs. In a future release to be determined, IDCNOGFL invocation will change to the newer...
CSI interface producing the same results as PGM=IDCAMS.
Migration plans should be established to migrate applications dependant on prior LISTCAT formats to IBM supported solutions.

If there is a continued need for prior generic locate results, please contact IBM as described in z/OS DFSMS Introduction, copied below.

---

z/OS V1R8 V1R9 DFSMS Introduction (copied March 14 2008).
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.
---

Another option is a STEPLIB to a z/OS V1R7 version of IDCAMS. z/OS V1R7 version of IDCAMS may not recognize features or satisfy requirements of later releases. Proper tests are required before accepting STEPLIB as a solution for each release STEPLIB will be used. STEPLIB usage should be added to future release migration and test plans until a proper replacement solution is provided.

For extreme outage related to automated program dependancy on LISTCAT output, please open a PMR providing IBM a description of the outage and impact, requesting temporary relief. The following questions will help assess the impact. Please include additional impact statements as needed to help clarify the relief needed. Suggested Impact Questions:
- Description of the impact, outage, scope
- How are the jobs critical to your business?
- Description of any time limitations and impact results
- How many jobs are affected?
- Description of SYSTEM and PLEX dependancies
- additional info to help understand your particular requirements.

LISTCAT provides catalog information in human readable format. Provided an availability of the PGM=IDCNOGFL fallback option, IBM requests all sev1 and sev2 APARs reporting errors in the new LISTCAT results should include the above impact statements for improved reviews and relief.

IBM may request the use of fallback support as relief for an error in the new CSI interface until a proper solution can be provided via a field fix or release offering. If fallback support does not provide needed relief, please provide the above impact statements to IBM service who will forward your
request for review. For automated applications which are exposed to errors discovered in the new interface, IBM recommends utilizing the fallback APAR and migrating the application to an IBM supported solution.

--------------------------------------------------------------

If you run an IDCAMS LISTCAT CAT(your.catalog.name) ALL or a LISTCAT LEVEL and there are catalog entries that indicate the data set is on a sms managed volume if the data set does not actually exist on the volume you will receive the following messages:
IDC3014I CATALOG ERROR
IDC3009I ** VSAM CATALOG RETURN CODE IS 50- REASON CODE IS IGGOCLEO-6
IDC1566I ** your.deleted.dataset NOT LISTED

and the LISTCAT will end with a condition code of 4. In prior releases, you would receive what information is available from the catalog and the LISTCAT would end with a condition code of 0.

If you have catalog entries that indicate the data set is not on a sms managed volume and the data set does not exist, you will continue to receive the information that is available from the catalog entry.

There are also some differences in how IDCAMS LISTCAT LEVEL processing works for GDG's and ALIASes. These changes make IDCAMS and ISPF option 3.4 more compatible.

To illustrate these changes we will use the following data set names:
SAMPLE.GDG
SAMPLE.GDG.G0001V00
SAMPLE.GDG.G0002V00
Which are cataloged in catalog sample.catalog
SAMPLE1.DATA.SET
SAMPLE1.DATA.SET.TWO
SAMPLE1 => ALIAS
which are cataloged in catalog sample.catalog.two
SAMPLE2.DATA
SAMPLE2.DATA.GOOD
SAMPLE2.DATA.REALLY.GOOD
which are cataloged in catalog sample.catalog.three

In z/OS 1.8 if you run IDCAMS LISTCAT LEVEL(SAMPLE.GDG) you will receive the output below:
GDG BASE ------ SAMPLE.GDG
IN-CAT --- SAMPLE.CATALOG

NONVSAM ------ SAMPLE.GDG.G0001V00
IN-CAT --- SAMPLE.CATALOG

NONVSAM ------ SAMPLE.GDG.G0002V00
IN-CAT --- SAMPLE.CATALOG

In prior releases the GDG BASE entry was not listed.

In z/OS 1.8 if you run IDCAMS LISTCAT LEVEL(SAMPLE1) you will receive the output below:
NONVSAM ------ SAMPLE1.DATA.SET
Appendix A. APARs to be reviewed

IN-CAT --- SAMPLE.CATALOG.TWO

NONVSAM ------- SAMPLE1.DATA.SET.TWO
IN-CAT --- SAMPLE.CATALOG.TWO

ALIAS ------- SAMPLE1
IN-CAT --- YOUR.MASTER.CATALOG
In prior releases the ALIAS entry was not listed.

In z/OS 1.8 if you run IDCAMS LISTCAT LEVEL(SAMPLE2.DATA) you will receive the output below:
NONVSAM ------- SAMPLE2.DATA
IN-CAT --- SAMPLE.CATALOG.THREE

NONVSAM ------- SAMPLE2.DATA.GOOD
IN-CAT --- SAMPLE.CATALOG.THREE

NONVSAM ------- SAMPLE2.DATA.REALLY.GOOD
IN-CAT --- SAMPLE.CATALOG.THREE
In prior releases only the second and third entries would be listed.

As a part of the above changes and the changes documented in the manual z/OS V1R8.0 DFSMS Using the New Functions the output listing from the IDCAMS LISTCAT command in z/OS 1.8 has been changed to include the additional header line per page as follows:
(Pre z/OS 1.8)
The output has following header line always after 17 entries:
   IDCAMS SYSTEM SERVICES   TIME: 10:08:42   09/28/06  PAGE 2310

(z/OS 1.8)
The 1.8 output has following header lines always after 16 entries:
   IDCAMS SYSTEM SERVICES   TIME: 10:13:49   09/28/06  PAGE 2461

   LISTING FROM CATALOG -- SYS1.USER.CATALOG

This change might increase the number of lines or pages produced by IDCAMS LISTCAT commands. This is to be expected in z/OS 1.8 and higher releases.

LISTCAT LEVEL processing in z/OS 1.8 and higher releases includes the following line:
   IN-CAT --- your.catalog.name
In previous releases this line did not occur if the CATALOG parameter was specified on the LISTCAT LEVEL command.

If you wish to see these enhancements under TSO, you will need to set your TSO PROFILE to NOPREFIX. If your TSO PROFILE is PREFIX, then you will receive output as in previous releases.

Generation data sets GDS are now returned by Catalog in time sequence order.

In z/OS 1.8 and higher releases LISTCAT LEVEL of a GDG may not return the GDS's associated with the GDG in generation number sequence. You may receive G0003V00 in the listing before G0002V00. All GDS's will be listed, but the sequence they are returned can vary if the GDS was updated. An example of an update that can cause this to occur is migration/recall of a GDS. If you have applications (assembler, cobol, pl/1, c/c++, rexx) that process LISTCAT LEVEL output they should be reviewed to ensure there is not a dependency on the order that GDS's are listed on the listcat output.

If your applications do have a dependency on the order that GDS's are returned in, then those applications will need to be updated to remove that dependency.

This may take time to accomplish. There are several means available to have LISTCAT LEVEL return GDS's in generation number order. These are as follows:

1. Add EXPIRATION(9999) to the LISTCAT LEVEL command
2. Add CREATION(0) to the LISTCAT LEVEL command
3. If running under TSO, Use TSO PROFILE PREFIX(value)
   Note that it does not matter what the value is, only that the TSO PREFIX is set.
4. After application of PTFs for OA20472, change your JCL from //STEP01 EXEC PGM=IDCAMS to //STEP01 EXEC PGM=IDCNOGFL
   Or change applications that invoke IDCAMS to invoke the entry point IDCNOGFL, rather than IDCAMS.

   Note that with any of the above options the performance enhancements and constraints removal in z/OS 1.8 will not be in effect.

The use of option 4 should be considered a temporary solution for the issue of GDS's being returned in an order other than generation number order. This option was provided to ease the migration to z/OS 1.8 and higher releases and is currently intended to be removed in a future release. To help ease migration the PTFs for apar OA22078 will provide the new entry point IDCNOGFL on z/OS 1.6 and z/OS 1.7. Preview announcements and product announcements for z/OS future releases will include this as a statement of direction. Removal of this option will be included in release and migration guides.

LISTCAT LIST CATalog formats returned Cataloged objects without interpretation. To determine a system determined data set attribute such as a time stamp may require additional resources such as VTOC DSCBs not queried or formatted by LISTCAT. Compare the results of interpretive products such as TSO to see system determined data set attributes such as security and timestamps. Also see DFSMS Using Data Sets and z/OS Network File System publications for additional information regarding system determined attributes.
In addition to differences in returned Catalog objects, the new Catalog interface may also return codes associated with the new interface. LISTCAT is also a conduit for Catalog errors, and may result in a change in the JCL COND CODE by design. Migration plans should include tests which exercise each JCL condition code path which are dependant on LISTCAT results.

Local Fix:
Responder Page
Problem
Summary:

Problem
Conclusion:

Temporary Fix:
Comments:

Modules/Macros:
SRLS:
Return Codes:
Circumvention:

Message To
Submitter:

APAR OA18720

Summary Page
APAR Identifier: OA18720 Last Changed: 07/02/26
Symptom: Status: CLOSED PER
Severity: 3 Date Closed: 06/10/26
Component: 5695DF103 Duplicate of:
Reported Release: 180 Fixed Release: 999
Component Name: DFSMS IDCAMS Special Notice:
Customer Name: SJ2025 Customer #: 999999999
Current Target Date: 06/11/16 Flags: HIPER/Y FUNCTIONLOSS PERVERSIVE
SCP:
Platform:
Status Detail: APARCLOSURE

' Fix Page
PE PTF List:

PTF List: Release 180 : UA29965 available 06/11/08 (F611)

Parent APAR: OA16912

Child APAR List:

' Submitter Page
Error.
Description: ROLL UP APAR FOR IDCAMS FOR 1.8

Local Fix:

Responder Page
Problem
Summary: ****************************************************************
* USERS AFFECTED: All users, Level 1.8 and higher.             *
************************************************************************
* PROBLEM DESCRIPTION: This APAR represents the roll-up APAR f *
* AMS LISTCAT with GFL.                                      *
************************************************************************
* RECOMMENDATION:                                              *
************************************************************************
This APAR is the 1.8 roll-up APAR for AMS LISTCAT with GFL.

Problem
Conclusion: This APAR is the roll-up APAR for AMS LISTCAT with GFL.

Temporary Fix:

Comments:

Modules/Macros: IDCLC02 IDCLC09

SRLS: NONE

Return Codes:

Circumvention:

Message To
Submitter:
Summary Page
APAR Identifier: IO03548 Last Changed: 08/08/25
Symptom: NEWFUNCTION Status: CLOSED UR1
Severity: 3 Date Closed: 08/08/08
Component: 565826001 Duplicate of:
Reported Release: 500 Fixed Release:
Component Name: OS/VS EREP VER Special Notice:
Customer Name: IBM INTERNAL Customer #: 9999998 00
Current Target Date: 07/04/24 Flags:
SCP:
Platform:
Status Detail: SHIPMENT

PTF List: Release 500 : U000829 available 08/08/25 (1000)

Parent APAR:

Child APAR List:

Local Fix:

Responder Page
Problem
Summary:  **********************************************************************
* USERS AFFECTED: D/T 2107 for EAV devices, model 3390-A
* PROBLEM DESCRIPTION: Provide formatting support in Detailed
*            Edit and System Exception reports for
*            EAV devices, Mod-A. For more details,
*            refer to publication changes.
* RECOMMENDATION:
* Provide formatting support for MDR/OBR records when using EAV
devices, 2107 Mod-A.

Problem
Conclusion:

Temporary Fix:
Comments: Refer to EREP Reference Manual for documentation updates.

Modules/Macros: EER3500J IFCBDAS2 IFCB3301 IFCB3305 IFCD2026 IFCI3390 IFCRDAS2 IFCXSELT IFCXTBLS

SRLS: GC35015204

Return Codes:

Circumvention:

Message To
Submitter:

APAR II13752

Summary Page
APAR Identifier: II13752 Last Changed: 08/05/01
Symptom: IN INCORROUT Status: INTRAN
Severity: 3 Date Closed:
Component: INFOV2LIB Duplicate of:
Reported Release: 001 Fixed Release:
Component Name: Special Notice:
Customer Name: IBM INTERNAL Customer #: 9999999
Current Target Date: Flags:
SCP:
Platform:
Status Detail:
  Fix Page
PE PTF List:

PTF List:

Parent APAR:

Child APAR List:

  Submitter Page
Error.
Description: This APAR describes the Tracking Facility, its output, how to send the output to IBM for analysis, and how to pull the current Parmlib member to aid in reducing the amount of data the Tracker collects.

The Tracking Facility has been included with the BCP element of z/OS in the Console Availability Feature (JBB7727) and has been included in all following BCP elements of z/OS releases.
It has several purposes:
- Identify products that continue to use functions that will not be supported in future z/OS releases.
- Identify products that would benefit from changing services or using different keywords on existing services.

Starting with z/OS 1.6 (HBB7709) IBM began removing support for 1-Byte Console IDs. Console Names or 4-Byte Console IDs (introduced in MVS/ESA 4.1.0) must be used instead. This became the first usage of the Tracking Facility.

With APARs OA19948 and OA20428, IBM began Tracking invocations of services where insufficient serialization was held or services were invoked when no processing was requested.

It is expected that over time, IBM will provide additional users of the Tracking Facility.

IBM testing organizations, ESP accounts, and customers should run with this facility active to help identify products that will need to be changed. It is recommended that the facilities ONWITHABEND option NOT be specified in z/OS 1.9 or above because some users of the facility are not reporting problems.

To learn more about the facility see Appendix A in z/OS MVS Planning: Operations (SA22-7601).

To activate/deactivate the facility, use the SETCON TRACKING command (see z/OS MVS System Commands - SA22-7627).

Once the facility is active, it will record instances of the event. The DISPLAY OPDATA,TR command is used to display the recorded events via message CNZ1001I (see z/OS MVS Commands - SA22-7627) is used to display the recorded events via message CNZ1001I (see z/OS MVS System Messages Volume 4 (CBD-DMO) - SA22-7634).

Before contacting IBM support concerning Tracker questions, send an e-mail to consoles@us.ibm.com. If you do need to contact IBM support, attempt to determine the appropriate IBM organization. For instance, if the Tracked entry indicates SMS, the DF/SMS support may be the best place to start.

To notify IBM of any recorded instances, send an e-mail to consoles@us.ibm.com providing the output produced by the DISPLAY OPDATA,TR command (message CNZ1001I). Also include any information that would help identify which product/vendor caused the event(s) to be tracked. IBM may notify the product owner or ask you to notify the vendor and request that updates be made. Please note that this is a staged evolution. The changes do not have to be made immediately as existing code will continue to work as it does today.

The Tracking facility supports an Exclusion list that informs the facility which instances have already been reported and
should no longer be tracked. This list is updated from the data sent to IBM. Current lists are available for download from...


This web page will include the date when the list was last updated. After IBM has updated the Exclusion list, it is suggested that the latest list is downloaded and installed to help to reduce duplicate notifications. When migrating to a new z/OS release, obtain the Exclusion List that is appropriate for that Release. Once the list has been downloaded, the "Special Processing" section should be reviewed and appropriate action(s) taken. Place the list in Parmlib and use the SET CNIDTR command (see z/OS MVS System Commands - SA22-7627) to activate it. Note that events that have been recorded before the activation of the new list will remain recorded and displayed in CNZ1001I, even if Excluded in the new list. To remove the old instances, the facility must be de-activated and re-activated using the new Exclusion list.

IBM encourages installations to update their copy of the Exclusion list for their specific applications. The list that can be downloaded contains details and examples of proper syntax for updating the list.

Following is an example of the Tracker report output (note, some columns were compressed to fit in this APAR description):

```
CNZ1001I 15.47.47 TRACKING DISPLAY
STATUS=ON       NUM=2    MAX=1000 MEM=n/a EXCL=0     REJECT=0
--TRACKING INFORMATION-- -VAL- JOBNAME  PROGNAME+OFF-- ASID NUM
WTO: IEC350I CATALOG ADD    00 CATALOG  IGG0CLX0 80BAC   1A   1
WTO: IEF677I WARNING MES    00 JES2     IEFNB903  C9AA   13   1
---------------------------------------------------------------
TO REPORT THESE INSTANCES, SEND THIS MESSAGE VIA E-MAIL TO CONSOLES@US.IBM.COM. FOR ADDITIONAL INFORMATION OR TO OBTAIN A CURRENT EXCLUSION LIST, SEE APAR II13752
```

Local Fix:

Responder Page
Problem
Summary:

Problem
Conclusion:

Temporary Fix:

Comments:
Modules/Macros:

SRLS:

Return Codes:

Circumvention:

Message To Submitter:
Code samples DFSMS 1.9

This appendix contains sample code that can be useful when implementing DFSMS V1.9.

Each sample program is documented separately so that they can be used individually as required.

In Table B-1 we list the programs provided.

*Table B-1  SMF extract sample programs*

<table>
<thead>
<tr>
<th>Program</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMF85TA (see “SMF Record type 85 subtype 1-7 data display program” on page 442) Note that this program was provided with the DFSMS 1.8 Technical Guide and that version can run without change because the additions are only to the FLAG contents.</td>
<td>OAM SMF TYPE85 records subtypes 1-7</td>
</tr>
<tr>
<td>SMF85TH (see “SMF Record type 85 subtype 32-35 data display program” on page 451)</td>
<td>OAM SMF TYPE85 records subtypes 32-35</td>
</tr>
<tr>
<td>SMF85TO (see “SMF Record type 85 subtype 38 data display program” on page 462) Note that this program was provided with the DFSMS 1.8 Technical Guide and that version can run if the change noted in the source code is implemented and the program re-assembled.</td>
<td>OAM SMF TYPE85 records subtype 38</td>
</tr>
<tr>
<td>SMF85TJ (see “SMF Record type 85 subtype 40 data display program” on page 468)</td>
<td>OAM SMF TYPE85 records subtype 40</td>
</tr>
</tbody>
</table>
B.1  SMF Record type 85 subtype 1-7 data display program

Program SMG85TA displays the contents of selected fields of SMF record Type 85 subtypes 1/2/3/4/5/6/7 data. It is not intended to provide a comprehensive report on OAM activity but rather to verify that immediate backup is occurring.

There are three steps to build the program, which need to be done once, after which it can be executed several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

Step 1: Create a PDS/PDSE to hold the members
In this example the PDS is called MHLRES1.SMF85TA.SOURCE.

The LRECL/RECFM must be 80/FB—all other attributes can be chosen by you.

Step 2: Store the program source in the PDS
Make sure that your ISPF EDIT menu settings has the screen format defined as DATA.

Cut and paste the contents of Figure B-1 on page 443 and all subsequent ones through to Figure B-7 on page 449, one after the other, into member SMFT85AA. The result should contain 333 lines.
MACRO
&NAME SEGSTART
&NAME STM 14,12,12(13)  SAVE HIS REGS IN HIS SAVE AREA
R0 EQU 0
R1 EQU 1
R2 EQU 2
R3 EQU 3
R4 EQU 4
R5 EQU 5
R6 EQU 6
R7 EQU 7
R8 EQU 8
R9 EQU 9
R10 EQU 10
R11 EQU 11
R12 EQU 12
R13 EQU 13
R14 EQU 14
R15 EQU 15
BALR 12,0  SET UP ADDRESSABILITY
USING *,12  USE REG 12 AS BASE REG
ST 13,SAVEREGS+4  SAVE @ OF HIS SAVEAREA IN MINE
LA 03,SAVEREGS  LOAD @ OF MY SAVE AREA IN REG 3
ST 03,8(13)  SAVE @ OF MY SAVE AREA IN HIS
LR 13,03  LOAD @ OF MY SAVE AREA IN REG 13
MEND
MACRO
&NAME SEGEND
&NAME L 13,SAVEREGS+4  LOAD REG13 WITH @ OF HIS SAVE
LM 14,12,12(13)  RESTORE REGS FROM HIS SAVEAREA
XR R15,R15
BR 14  RETURN TO CALLING RTN VIA REG 14
SAVEREGS DC 18F'0'  SET UP SAVE AREA
MEND
SMFR85TA SEGSTART
* THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS PARTS OF
* THE SMF TYPE 85 SUBTYPE 1-7 RECORDS.
* IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED
* TO SELECT ANY OR ALL OF TYPE 85 SUBTYPES 1-7
* RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR
* OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.
* THE STANDARD SMF RECORD MAPPING MACROS ARE USED.
* REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD
* R3 START OF WHOLE RECORD
* THERE IS 1 DSECTS TO BE MAPPED
* R4 START OF SUBTYPE RECORDS
* R5 SPARE
* R6 SPARE
* R7 SPARE
* OTHER REGISTER USES
* R12 OVERALL BASE REGISTER

Figure B-1 SMF85TA assembler source (part 1 of 7)
* R8 RECORD TYPE/SUBTYPE CHECKING/WORKING
* R9 LENGTH OF PARTICULAR DSECT
* R10 NUMBER OF ENTRIES IN THE TRIPLET
*
* QSAM GET LOCATE PROCESSING IS USED
*
OPEN SMFIN
OPEN (PRINTCB,(OUTPUT))
PUT PRINTCB,PRINTHDR
READ GET SMFIN
* COPY PARAMETER POINTER
   LR R3,R1
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
   USING CBRSMF85,R3
* CHECK IF TYPE 85
   CLI SMF85RTY,X'55'
   BNE IGNORE
*
DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS
CHKSTYP1 DS 0H
* CHECK IF ANY OF SUBTYPE 1-7
   CLI SMF85STY+1,X'01'
   BNE *+18
   MVI STYPE,C'1'
   MVC FUNC,=CL15'OSREQ ACCESS'
   B STOK
   CLI SMF85STY+1,X'02'
   BNE *+18
   MVI STYPE,C'2'
   MVC FUNC,=CL15'OSREQ STORE'
   B STOK
   CLI SMF85STY+1,X'03'
   BNE *+18
   MVI STYPE,C'3'
   MVC FUNC,=CL15'OSREQ RETRIEVE'
   B STOK
   CLI SMF85STY+1,X'04'
   BNE *+18
   MVI STYPE,C'4'
   MVC FUNC,=CL15'OSREQ QUERY'
   B STOK
   CLI SMF85STY+1,X'05'
   BNE *+18
   MVI STYPE,C'5'
   MVC FUNC,=CL15'OSREQ CHANGE'
   B STOK
   CLI SMF85STY+1,X'06'
   BNE *+18
   MVI STYPE,C'6'
   MVC FUNC,=CL15'OSREQ DELETE'
   B STOK

Figure B-2  SMF85TA assembler source (part 2 of 7)
CLI   SMF85STY+1,X'07'
BNE  *+18
MVI   STYPE,C'7'
MVC   FUNC,=CL15'OSREQ UNACCESS'
B    STOK
* OTHERWISE IGNORE
B     IGNORE

STOK  EQU  *
*
DC   F'0' CREATE AN ABEND TO LOOK AT THE RECORDS
* IS ONE OF TYPE 85 SUBTYPE 1-7 SO EXTRACT DATA
* R3 IS THE START OF THE WHOLE RECORD
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION
* ADD R8 TO R3
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS
LA   R4,SMF85END
USING ST1,R4
L   R8,SMF85S0
LH  R9,SMF85S0L
LH  R10,SMF85S0N
* PROCESS THE SUMMARY ENTRIES TRIPLET.
* FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START
* SECOND HW IS THE LENGTH OF EACH TRIPLET
* THIRD HW IS THE NUMBER OF TRIPLETS
* FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM
* THE SMF RECORD TYPE 85 SUBTYPE 1-7 RECORDS.
* COLN COMES FROM ST1COLN
* OBJN COMES FROM ST1OBJN
* ETC
* ST1FLGS IS NOT INTERPRETED - EACH BIT JUST SHOWN AS 1 OR 0
*
SCOTRIP  DS    0H
LA   R4,0(R3,R8)
LA   R4,0(R3,R8)
UNPK  YYDDD(7),SMF85DTE
CLI   YYDDD+1,C'0'
BE    SETD0
CLI   YYDDD+1,C'1'
BE    SETD1
* OTHERWISE ABEND AS SOMETHING HAS GONE WRONG
DC   F'0'
SETD0   MVC   YYDDD(2),=C'19'
B    SETDZ
SETD1   MVC   YYDDD(2),=C'20'
*
SETDZ EQU *
* CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT
LA R5,100 PREPARE TO DIVIDE BY 100
LA R6,0
L R7,SMF85TME GET THE TIME
DR R6,R5 -> SECS IN R7, HUNS IN R6
CVD R6,DWORD
OI DWORD+7,X'0F' FIX THE SIGN FOR PRINTING
UNPK HUS,DWORD+6(2)
*
DC F'0'
* NOW GET THE SECS
LA R5,60 PREPARE TO DIVIDE BY 60
LA R6,0
DR R6,R5 -> MINS IN R7, SECS REMAINDER IN R6
CVD R6,DWORD
OI DWORD+7,X'0F' FIX THE SIGN FOR PRINTING
UNPK SS,DWORD+6(2)
* NOW GET THE MINS
LA R6,0
DR R6,R5 -> HRS IN R7, MINS REMAINDER IN R6
CVD R6,DWORD
OI DWORD+7,X'0F' FIX THE SIGN FOR PRINTING
UNPK MM,DWORD+6(2)
CVD R7,DWORD DO HOURS
OI DWORD+7,X'0F' FIX THE SIGN FOR PRINTING
UNPK HH,DWORD+6(2)
*
PUT PRINTDCB,PRINTLO
*
PUT PRINTDCB,PRINTL1
* COPY COLN
MVC COLN,ST1COLN
* COPY OBJN
MVC OBJN,ST1OBJN
PUT PRINTDCB,PRINTL2
*
* COPY SGN
MVC SGN,ST1SGN
* COPY SCN
MVC SCN,ST1SCN
* COPY MCN
MVC MCN,ST1MCN
*
* CONVERT LEN
L R1,ST1LEN
CVD R1,DWORD
OI DWORD+7,X'0F'
UNPK LEN(11),DWORD+2(6)

Figure B-4 SMF85TA assembler source (part 4 of 7)
* CONVERT  TTOK & TOK
* DO TOK FIRST

* TOK IS 8 BYTES BINARY -> 16 BYTES PRINTABLE
* MVC  TOK,STITOK
UNPK  TRWORK(15),STITOK+1(8)  15 BYTES (ONE REDUNDANT BYTE)
NC  TRWORK(15),=15X'0F'
TR  TRWORK(15),TRTAB
MVC  TOK+2(14),TRWORK
UNPK  TRWORK(3),STITOK(2)  LAST BYTE + ONE REDUNDANT BYTE
NC  TRWORK(3),=3X'0F'
TR  TRWORK(3),TRTAB
MVC  TOK(2),TRWORK

* TTOK IS 16 BYTES BINARY -> 32 BYTES PRINTABLE
* HAVE TO UNPACK THIS AS TWO SETS OF 16 AS PER TOK
* FIRST DO 16 BYTES, THE REPEAT FOR THE NEXT TWO
UNPK  TRWORK(15),STITTKOK+1(8)  15 BYTES (ONE REDUNDANT BYTE)
NC  TRWORK(15),=15X'0F'
TR  TRWORK(15),TRTAB
MVC  TTOK+2(14),TRWORK
UNPK  TRWORK(3),STITTKOK(2)  LAST BYTES + ONE REDUNDANT BYTE
NC  TRWORK(3),=3X'0F'
TR  TRWORK(3),TRTAB
MVC  TTOK(2),TRWORK

* NOW DO IT ALL AGAIN WITH OFFSET OF 8 ON STITTKOK AND OFFSET OF 16 ON
* TTOK
UNPK  TRWORK(15),STITTKOK+8(8)  15 BYTES (ONE REDUNDANT BYTE)
NC  TRWORK(15),=15X'0F'
TR  TRWORK(15),TRTAB
MVC  TTOK+16(14),TRWORK
UNPK  TRWORK(3),STITTKOK+8(2)  LAST BYTES + ONE REDUNDANT BYTE
NC  TRWORK(3),=3X'0F'
TR  TRWORK(3),TRTAB
MVC  TTOK+16(2),TRWORK

* TRANSLATE TO PRINTABLE
   PUT  PRINTDCB,PRINTL3
*
* COPY VSN & VMT
   MVC  VSN,STIVSN
   MVC  MT,STIVMT
* CONVERT RC & RS
* CONVERT RC
   L   R1,STIRC
   CVD  R1,DWORD
   OI  DWORD+7,X'0F'
   UNPK  RC(08),DWORD+4(4)
* CONVERT RS
   L   R1,STIRS
   CVD  R1,DWORD
   OI  DWORD+7,X'0F'
   UNPK  RC(08),DWORD+4(4)

Figure B-5   SMF85TA assembler source (part 5 of 7)
* PRINT FLAGS
    UNPK FLGS(09),ST1FLGS(5) UNPK 1 MORE THAN NEEDED
    MVI FLGS+8,C' ' BLANK OUT THE EXTRA BYTE
    NC FLGS(08),=8X'0F'
    TR FLGS(8),TRTAB
    PUT PRINTDCB,PRINTL4

WRITEIT DS OH
    PUT PRINTDCB,PRINTBLK

* LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLET
*

* WHEN BCT REACHES ZERO GO GET ANOTHER RECORD
    LA R8,0(R8,R9)
    BCT R10,SCOTRIP
    B READ

IGNORE DS OH EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS
    B READ

FINISH DS OH
SEGEND

SMFIN DCB DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,
     EODAD=FINISH

PRINTDCB DCB DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133

DWORD DS D
    ORG DWORD
    DC C'12345678'

TRWORK DS CL33

TRTAB DC C'0123456789ABCDEF'

PRINTBLK DC CL133' '

PRINTHDR DC CL133'1SMF TYPE 85 SUBTYPE 1-7 RECORDS'

PRINTLO DC CL133' SMFDTE/TME:'
    ORG PRINTLO+27

YYDDD DC CL7' '
    DC CL1' '

HH DC CL2' '
    DC C':'

MM DC CL2' '
    DC C':'

SS DC CL2' '
    DC C':'

HUS DC CL3' '
    DC CL1' '
    ORG

PRINTL1 DC CL133' STYPE:'
    ORG PRINTL1+27

STYPE DC CL1' ' CONVERTED
    DC CL1' '

FUNC DC CL15' '
    ORG

Figure B-6  SMF85TA assembler source (part 6 of 7)
**Step 3: Store the JCL to assemble and link the source in the PDS**

Cut and paste the contents of Figure B-8 on page 450 into your PDS MHLRES1.SMF85TA.SOURCE as member SMFT85AJ. The result should contain 32 lines.

Run the job when the member has been created.
This will create data set MHLRES1.SMF85TA.LOAD.

Note: If this JCL needs to be rerun, change the lines:

```
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TA.LOAD,DISP=(,CATLG,DELETE),
//                  UNIT=SYSALLDA,SPACE=(CYL,(1,1,1))
//LKED.SYSIN DD *
SETSSI 00001800
NAME SMF85TA(R)
```

To read:

```
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TA.LOAD,DISP=SHR,(,CATLG,DELETE),
/* UNIT=SYSDA,SPACE=(CYL,(1,1,1))
To stop it from trying to make the data set again.
B.2 SMF Record type 85 subtype 32-35 data display program

Program SMG85TH displays the contents of selected fields of SMF record Type 85 subtypes 32/33/34/35 data. It is not intended to provide a comprehensive report on OAM activity but rather to verify that immediate backup is occurring.

There are three steps to build the program which needs to be done once, after which it can be executed several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

Step 1: Create a PDS/PDSE to hold the members
In this example the PDS is called MHLRES1.SMFG85TH.SOURCE.

The LRECL/RECFM must be 80/0F—all other attributes can be chosen by you.

Step 2: Store the program source in the PDS
Make sure that your ISPF EDIT menu settings has the screen format defined as DATA.

Cut and paste the contents of Figure B-9 here and all subsequent ones through to Figure B-18 on page 460, one after the other, into member SMG85TH. The result should contain 501 lines.

```
MACRO 00010099
&NAME SEGSTART 00020099
&NAME STM 14,12,12(13) 00030099
R0 EQU 0 00040099
R1 EQU 1 00050099
R2 EQU 2 00060099
R3 EQU 3 00070099
R4 EQU 4 00080099
R5 EQU 5 00090099
R6 EQU 6 00100099
R7 EQU 7 00110099
R8 EQU 8 00120099
R9 EQU 9 00130099
R10 EQU 10 00140099
R11 EQU 11 00150099
R12 EQU 12 00160099
R13 EQU 13 00170099
R14 EQU 14 00180099
R15 EQU 15 00190099
BALR 12,0 00200099
USING *,12 00210099
ST 13,SAVEREGS+4 00220099
LA 03,SAVEREGS 00230099
ST 03,8(13) 00240099
LR 13,03 00250099
MEND 00260099
MACRO 00270099
&NAME SEGEND 00280099
&NAME L 13,SAVEREGS+4 00290099
LM 14,12,12(13) 00300099
XR R15,R15 00310099
BR 14 00320099
```

Figure B-9 SMG85TH assembler source (part 1 of 10)
SAVEREGS DC 18F'0'
MEND
MACRO
BINDEC &KEY
L R7,ST32&KEY
CVD R7,DWORD
OI DWORD+7,X'0F'
UNPK &KEY.(7),DWORD+4(4)
MEND
SMFR85TH SEGSTART
* THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS PARTS OF
* THE SMF TYPE 85 SUBTYPE 32-35 RECORDS.
* IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED
* TO SELECT ANY OR ALL OF TYPE 85 SUBTYPES 32-35
* RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR
* OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.
* THE STANDARD SMF RECORD MAPPING MACROS ARE USED.
* REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD
* R3 START OF WHOLE RECORD
* THERE IS 1 DSECTS TO BE MAPPED
* R4 START OF SUBTYPE RECORDS
* R5 FOR DIVIDING (DIVISOR)
* R6 FOR DIVIDING - EVEN-ODD PAIR WITH R7 (DIVIDEND)
* R7 FOR DIVIDING
* OTHER REGISTER USES
* R12 OVERALL BASE REGISTER
* R8 RECORD TYPE/SUBTYPE CHECKING/WORKING
* R9 LENGTH OF PARTICULAR DSECT
* R10 NUMBER OF ENTRIES IN THE TRIPLLET
* QSAM GET LOCATE PROCESSING IS USED

OPEN SMFIN
OPEN (PRINTDCB,(OUTPUT))
PUT PRINTDCB,PRINTHDR
READ GET SMFIN
* COPY PARAMETER POINTER
   LR R3,R1
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
   USING CBRSMF85,R3
* CHECK IF TYPE 85
   CLI SMF85RTY,X'55'
   BNE IGNORE
   DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS
* CHKSTYP1 DS OH
* CHECK IF ANY OF SUBTYPE 32-35
   CLI SMF85STY+1,X'20'
   BNE "+18
   MVI STYPE,C'2'
   MVC FUNC,="CL34'(STORAGE GROUP PROCESSING)'
   B STOK
   CLI SMF85STY+1,X'21'
   BNE "+18
   MVI STYPE,C'3'
   MVC FUNC,="CL34'(DASD SPACE MANAGEMENT PROCESSING)'

Figure B-10  SMF85TH assembler source (part 2 of 10)
Figure B-11  SMF85TH assembler source (part 3 of 10)
<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETDO MVC YYDDD(2),=C'19'</td>
<td>01370099</td>
</tr>
<tr>
<td>B SETDZ</td>
<td>01380099</td>
</tr>
<tr>
<td>SETD1 MVC YYDDD(2),=C'20'</td>
<td>01390099</td>
</tr>
<tr>
<td>*</td>
<td>01400099</td>
</tr>
<tr>
<td>SETDZ EQU *</td>
<td>01410099</td>
</tr>
<tr>
<td>*</td>
<td>CONVERT THE TIME FROM HUNDREDS OF SEC SINCE MIDNIGHT</td>
</tr>
<tr>
<td>LA R5,100</td>
<td>PREPARE TO DIVIDE BY 100</td>
</tr>
<tr>
<td>LA R6,0</td>
<td>01430099</td>
</tr>
<tr>
<td>L R7,SMF85TME</td>
<td>GET THE TIME</td>
</tr>
<tr>
<td>DR R6,R5</td>
<td>-&gt; SECS IN R7, HUNS IN R6</td>
</tr>
<tr>
<td>CVD R6,DWORD</td>
<td>01460099</td>
</tr>
<tr>
<td>OI DWORD+7,X'0F'</td>
<td>FIX THE SIGN FOR PRINTING</td>
</tr>
<tr>
<td>UNPK HUS,DWORD+6(2)</td>
<td>01480099</td>
</tr>
<tr>
<td>*</td>
<td>DC F'0'</td>
</tr>
<tr>
<td>*</td>
<td>NOW GET THE SECS</td>
</tr>
<tr>
<td>LA R5,60</td>
<td>PREPARE TO DIVIDE BY 60</td>
</tr>
<tr>
<td>LA R6,0</td>
<td>01520099</td>
</tr>
<tr>
<td>DR R6,R5</td>
<td>-&gt; MINS IN R7, SECS REMAINDER IN R6</td>
</tr>
<tr>
<td>CVD R6,DWORD</td>
<td>01550099</td>
</tr>
<tr>
<td>OI DWORD+7,X'0F'</td>
<td>FIX THE SIGN FOR PRINTING</td>
</tr>
<tr>
<td>UNPK SS,DWORD+6(2)</td>
<td>01560099</td>
</tr>
<tr>
<td>*</td>
<td>NOW GET THE MINS</td>
</tr>
<tr>
<td>LA R6,0</td>
<td>01590099</td>
</tr>
<tr>
<td>DR R6,R5</td>
<td>-&gt; HRS IN R7, MINS REMAINDER IN R6</td>
</tr>
<tr>
<td>CVD R6,DWORD</td>
<td>01610099</td>
</tr>
<tr>
<td>OI DWORD+7,X'0F'</td>
<td>FIX THE SIGN FOR PRINTING</td>
</tr>
<tr>
<td>CVD R7,DWORD</td>
<td>DO HOURS</td>
</tr>
<tr>
<td>OI DWORD+7,X'0F'</td>
<td>FIX THE SIGN FOR PRINTING</td>
</tr>
<tr>
<td>UNPK HH,DWORD+6(2)</td>
<td>01650099</td>
</tr>
<tr>
<td>*</td>
<td></td>
</tr>
<tr>
<td>PUT PRINTDCB,PRINTL0</td>
<td>01670099</td>
</tr>
<tr>
<td>*</td>
<td>COPY SGN</td>
</tr>
<tr>
<td>MVC SGN,ST32SGN</td>
<td>01690099</td>
</tr>
<tr>
<td>*</td>
<td>COPY VSMO</td>
</tr>
<tr>
<td>MVC VSNO,ST32VSNO</td>
<td>01700099</td>
</tr>
<tr>
<td>*</td>
<td>COPY VSM1</td>
</tr>
<tr>
<td>MVC VSN1,ST32VSN1</td>
<td>01720009</td>
</tr>
<tr>
<td>*</td>
<td>COPY MT</td>
</tr>
<tr>
<td>MVC MT,ST32MT</td>
<td>01750099</td>
</tr>
<tr>
<td>PUT PRINTDCB,PRINTL1</td>
<td>01770099</td>
</tr>
<tr>
<td>*</td>
<td></td>
</tr>
<tr>
<td>BINDEC PDWO</td>
<td>CONVERT</td>
</tr>
<tr>
<td>BINDEC PDWK</td>
<td>01790099</td>
</tr>
<tr>
<td>BINDEC PDRW</td>
<td>CONVERT</td>
</tr>
<tr>
<td>BINDEC PDRK</td>
<td>01800099</td>
</tr>
<tr>
<td>BINDEC PDDW</td>
<td>CONVERT</td>
</tr>
<tr>
<td>BINDEC PDDK</td>
<td>01810099</td>
</tr>
<tr>
<td>PUT PRINTDCB,PRINTL2</td>
<td>01820009</td>
</tr>
<tr>
<td>BINDEC PDWO</td>
<td>01830099</td>
</tr>
<tr>
<td>BINDEC PDWK</td>
<td>01840099</td>
</tr>
<tr>
<td>BINDEC PDDW</td>
<td>01850099</td>
</tr>
<tr>
<td>BINDEC PDDK</td>
<td>01860099</td>
</tr>
<tr>
<td>PUT PRINTDCB,PRINTL3</td>
<td>01870099</td>
</tr>
<tr>
<td>BINDEC PDWO</td>
<td>01880099</td>
</tr>
<tr>
<td>BINDEC PDWK</td>
<td>01890099</td>
</tr>
<tr>
<td>BINDEC PDDW</td>
<td>01900099</td>
</tr>
<tr>
<td>BINDEC PDDK</td>
<td>01910099</td>
</tr>
<tr>
<td>PUT PRINTDCB,PRINTL4</td>
<td>01920099</td>
</tr>
</tbody>
</table>

**Figure B-12  SMF85TH assembler source (part 4 of 10)**
Figure B-13  SMF85TH assembler source (part 5 of 10)
* CONVERT 32KD 02450099
  L R1,ST3232KD 02460099
  CVD R1,DWORD 02470099
  OI DWORD+7,X'0F' 02480099
  UNPK N32KD(11),DWORD+2(6) 02490099
  BINDEC NCE CONVERT 02500099
  PUT PRINTDCB,PRINTL9 02510099
* INTERPRET THE FLAGS 02520099
  UNPK FLGS(09),ST32FLGS(5) UNPK 1 MORE THAN NEEDED 02530099
  MVI FLGS+8,C' ' BLANK OUT THE EXTRA BYTE 02540099
  NC FLGS(08),-8X'0F' 02550099
  TR FLGS(8),TRTAB 02560099
  BINDEC NTE CONVERT 02570099
  BINDEC RCLD CONVERT 02580099
  BINDEC RCLK CONVERT 02590099
  BINDEC LOBI CONVERT 02600099
  BINDEC LOBO CONVERT 02610099
  PUT PRINTDCB,PRINTL10 02620099
* CONVERT PUWO 04681000
  L R1,ST32PUWO 04682000
  CVD R1,DWORD 04683000
  OI DWORD+7,X'0F' 04684000
  UNPK PUWO(11),DWORD+2(6) 04685000
* CONVERT PUWK 04686000
  L R1,ST32PUWK 04687000
  CVD R1,DWORD 04688000
  OI DWORD+7,X'0F' 04689000
  UNPK PUWK(11),DWORD+2(6) 04689100
* CONVERT PURO 04689200
  L R1,ST32PURO 04689300
  CVD R1,DWORD 04689400
  OI DWORD+7,X'0F' 04689500
  UNPK PURO(11),DWORD+2(6) 04689600
* CONVERT PURK 04689700
  L R1,ST32PURK 04689800
  CVD R1,DWORD 04689900
  OI DWORD+7,X'0F' 04690000
  UNPK PURK(11),DWORD+2(6) 04690100
* CONVERT PUDO 04690200
  L R1,ST32PUDO 04690300
  CVD R1,DWORD 04690400
  OI DWORD+7,X'0F' 04690500
  UNPK PUDO(11),DWORD+2(6) 04690600
* CONVERT PUDK 04690700
  L R1,ST32PUDK 04690800
  CVD R1,DWORD 04690900
  OI DWORD+7,X'0F' 04691000
  UNPK PUDK(11),DWORD+2(6) 04691100
* PUT PRINTDCB,PRINTL11 04691200
* DC F'O' CREATE AN ABEND TO LOOK AT THE RECORDS 02630099
  PUT PRINTDCB,PRINTBLK 02640099
* LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETs 02650099

Figure B-14   SMF85TH assembler source (part 6 of 10)
* WHEN BCT REACHES ZERO GO GET ANOTHER RECORD

```
LA R8,0(R8,R9)          02660099
BCT R10,SCOTRIP         02670099
B READ                  02680099
IGNORE DS OH EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS
B READ                  02700099
FINISH DS OH            02720099
SEGEND                  02730099
SMFIN DCB DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,
                      EODAD=FINISH             02740099
PRINTDCB DCB DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133  02750099
DWORD DS D              02760099
ORG DWORD                02770099
DC C'12345678'          02780099
TRWORK DS CL33          02790099
TRTAB DC C'0123456789ABCDEF'  0280099
PRINTBLK DC CL133' '    02810099
PRINTHDR DC CL133' ISMF TYPE 85 SUBTYPE 32-35 RECORDS' 02820099
PRINTLO DC CL133' SMFDTE/TME:'  02830099
ORG PRINTLO+38          02840099
YYDDD DC CL7' '         02850099
   DC CL1' '            02860099
   HH DC CL2' '         02870099
   DC C' '             02880099
   MM DC CL2' '         02890099
   DC C' '             02900099
   SS DC CL2' '         02910099
   DC C' '             02920099
   HUS DC CL3' '        02930099
   DC CL1' '           02940099
ORG                  02950099
PRINTL1 DC CL133' STYPE/SGN/VSN0/VSN1/MT:' 02960099
ORG PRINTL1+38         02970099
STYLE3 DC CL1'3' PREFIX TO SUBTYPES 32-35 02980099
STYLE DC CL1' ' CONVERTED 02990099
   DC CL1' '           03000099
   SGN DC CL8' '       03010099
   DC C' '            03020099
   VSN0 DC CL6' '      03030099
   DC C' '            03040099
   VSN1 DC CL6' '      03050099
   DC C' '            03060099
   MT DC CL2' '        03070099
   DC C' '            03080099
   FUNC DC CL34' '     03090099
ORG                  03100099
*                      03110099
```

Figure B-15  SMF85TH assembler source (part 7 of 10)
Figure B-16   SMF85TH assembler source (part 8 of 10)
Figure B-17  SMF85TH assembler source (part 9 of 10)
**Figure B-18** SMF85TH assembler source (part 10 of 10)

```
PRINTL9 DC CL133' DTUP/DTDE/4KIN/4KDE/32KI/32KD/NCE:' 04190099
          ORG PRINTL9+38 04200099
DTUP   DC CL12' ' CONVERTED 04210099
          DC CL1' ' 04220099
DTDE   DC CL12' ' CONVERTED 04230099
          DC CL1' ' 04240099
N4KIN  DC CL12' ' CONVERTED 04250099
          DC CL1' ' 04260099
N4KDE  DC CL12' ' CONVERTED 04270099
          DC CL1' ' 04280099
N32KI  DC CL12' ' CONVERTED 04290099
          DC CL1' ' 04300099
N32KD  DC CL12' ' CONVERTED 04310099
          DC CL1' ' 04320099
NCE    DC CL12' ' CONVERTED 04330099
          ORG 04340099
PRINTL10 DC CL133' FLGS/NTE/RCLD/RCLK/LOBI/LOBD:' 04350099
          ORG PRINTL10+38 04360099
FLGS   DC CL9' ' INTERPRETED AS 0 OR 1 04370099
          DC CL4' ' 04380099
NTE    DC CL12' ' CONVERTED 04390099
          DC CL1' ' 04400099
RCLD   DC CL12' ' CONVERTED 04410099
          DC CL1' ' 04420099
RCLK   DC CL12' ' CONVERTED 04430099
          DC CL1' ' 04440099
LOBI   DC CL12' ' CONVERTED 04450099
          DC CL1' ' 04460099
LOBD   DC CL12' ' CONVERTED 04470099
          ORG 04480099
PRINTL11 DC CL133' PUWO/PUWK/PURO/PURK/PUDO/PUDK:' 06561000
          ORG PRINTL11+38 06562000
PUWO   DC CL12' ' CONVERTED 06565000
          DC CL1' ' 06566000
PUWK   DC CL12' ' CONVERTED 06567000
          DC CL1' ' 06568000
PURO   DC CL12' ' CONVERTED 06569000
          DC CL1' ' 06569100
PURK   DC CL12' ' CONVERTED 06569200
          DC CL1' ' 06569300
PUDO   DC CL12' ' CONVERTED 06569400
          DC CL1' ' 06569500
PUDK   DC CL12' ' CONVERTED 06569600
SMFDSECT DSECT 04490099
   IFASMFR (85) THIS INCLUDES CBRSMF MACRO 04500099
   END 04510099
```
Step 3: Store the JCL to assemble and link the source in the PDS

Cut and paste the contents of Figure B-19 into your PDS MHLRES1.SMF85TH.SOURCE as member SMF85THJ. The result should contain 32 lines.

Run the job when the member has been created.

This will create data set MHLRES1.SMF85TH.LOAD.

Note: If this JCL needs to be rerun, change the lines:

```jsp
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TH.LOAD,DISP=(),UNIT=SYSALLDA,SPACE=(CYL,(1,1,1))
```

To read:

```jsp
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TH.LOAD,DISP=SHR (),UNIT=SYSALLDA,SPACE=(CYL,(1,1,1))
```

i.e. to stop it trying to make the data set again.
B.3 SMF Record type 85 subtype 38 data display program

Program SMG85TO displays the contents of selected fields of SMF record Type 85 subtype 38 data. It is not intended to provide a comprehensive report on OAM activity but rather to verify that retrieval from Tape to DASD is occurring when an object is recalled.

There are three steps to build the program which needs to be done once, after which it can be executed several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

**Step 1: Create a PDS/PDSE to hold the members**
In this example the PDS is called MHLRES1.SMF85TO.SOURCE.

The LRECL/RECFM must be 80/FB—all other attributes can be chosen by you.

**Step 2: Store the program source in the PDS**
Cut and paste the contents of Figure B-20 on page 463 and all subsequent ones through to Figure B-23 on page 466, one after the other, into member SMFT85TO. The result should contain 188 lines.

*Note:* In Figure B-23 on page 466, the line labelled SMFIN has a continuation character. This must be in column 72.
Figure B-20  SMF85TO assembler source (part 1 of 4)
* OTHER REGISTER USES
* R12 OVERALL BASE REGISTER
* R8 RECORD TYPE/SUBTYPE CHECKING/WORKING
* R9 LENGTH OF PARTICULAR DSECT
* R10 NUMBER OF ENTRIES IN THE TRIPLET
*
* QSAM GET LOCATE PROCESSING IS USED
*
OPEN SMFIN
OPEN (PRINTDCB,(OUTPUT))
PUT PRINTDCB,PRINTHDR
READ GET SMFIN
* COPY PARAMETER POINTER
   LR R3,R1
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
   USING CBRSMF85,R3
* CHECK IF TYPE 85
   CLI SMF85RTY,X'55'
   BNE IGNORE
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS
   CHKSTYP1 DS OH
* CHECK IF SUBTYPE 38
   CLI SMF85STY+1,X'26'
   BNE IGNORE
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS
* IS TYPE 85 SUBTYPE 38, SO EXTRACT DATA
* R3 IS THE START OF THE WHOLE RECORD
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION
* ADD R8 TO R3
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS
   LA R4,SMF85END
   USING ST38,R4
   L R8,SMF85OSO
   LH R9,SMF85OSL
   LH R10,SMF85OSN
* PROCESS THE SUMMARY ENTRIES TRIPLET.
* FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START
* SECOND HW IS THE LENGTH OF EACH TRIPLET
* THIRD HW IS THE NUMBER OF TRIPLETS
* FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM
* THE SMF RECORD TYPE 85 SUBTYPE 38 RECORDS.
* COLN COMES FROM ST38COLN
* CNID COMES FROM ST38CNID
* ETC
* ST38FLGS IS INTERPRETED AS FLGO ON OR OFF
*
Figure B-21  SMF85TO assembler source (part 2 of 4)
* NOTE THAT THERE IS A PROBLEM WITH THE CBRSMF MACRO WHICH DOES
* NOT DESCRIBE THE OFFSET TO ST38FLGS CORRECTLY. THE CODE
* ACCOMODATES THIS, BUT IT WILL HAVE TO BE ADJUSTED WHEN THE
* CBRSMF MACRO IS FIXED.

**Figure B-22   SMF85TO assembler source (part 3 of 4)**

```
**********************************************************************
* SCOTRIP      DS    0H
  LA    R4,0(R3,R8)
  MVC    COLN,ST38COLN
* CONVERT CNID
  L     R1,ST38CNID
  CVD   R1,DWORD
  OI    DWORD+7,X'0F'
  UNPK  CNID(11),DWORD+2(6)
  PUT   PRINTDCB,PRINTL1
  MVC    OBJN,ST38OBJN
  MVC    SGN,ST38SGN
* CONVERT OLEN
  L     R1,ST38OLEN
  CVD   R1,DWORD
  OI    DWORD+7,X'0F'
  UNPK  OLEN(11),DWORD+2(6)
  PUT   PRINTDCB,PRINTL2
* MVC    VSN,ST38VSN
  MVC    MT,ST38MT
* CONVERT TKN
  L     R1,ST38TKN
  CVD   R1,DWORD
  OI    DWORD+7,X'0F'
  UNPK  TKN(11),DWORD+2(6)
  MVC    VT,ST38VT
  MVC    BT,ST38BT
* CONVERT FLAGS
  LA    R1,ST38FLGS
  MVC    FLGS,=CL20'FLG0 OFF'
  TM    0(R1),ST38FLGO  IS THE FLAG ON?
  BNO   FLG0OFF
  MVC    FLGS(08),=C'FLGO ON '
FLG0OFF EQU *
  PUT   PRINTDCB,PRINTL3
* DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS
WRITEIT  DS    0H
  PUT   PRINTDCB,PRINTBLK
```

Appendix B. Code samples DFSMS 1.9 465
* LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS
*
* WHEN BCT REACHES ZERO GO GET ANOTHER RECORD
LA   R8,0(R8,R9)
BCT  R10,SCOTRIP
B    READ
IGNORE DS OH EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS
B    READ
FINISH DS OH
SEGEND

SMFIN  DCB DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,
        C EODAD=FINISH
PRINTDCB DCB DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133
DWORD DS D
ORG DWORD
DC "C'12345678'"
PRINTBLK DC CL133' ' 
PRINTHDR DC CL133'1SMF TYPE 85 SUBTYPE 38 RECORDS'
PRINTL1 DC CL133' COLN/CNID:'
ORG PRINTL1+23
COLN DC CL44' ' 
DC C' ' 
CNID DC CL20' ' CONVERTED FROM BL4
ORG

*                      PRINTL2 DC CL133' OBJN/SGN/OLEN:'
                      ORG PRINTL2+23
OBJN DC CL44' ' 
DC CL1' ' 
SGN DC CL8' ' 
DC CL1' ' 
OLEN DC CL20' ' CONVERTED FROM BL4
ORG

*                     PRINTL3 DC CL133' VSN/MT/TKN/VT/BT/FLGS:'
                     ORG PRINTL3+23
VSN DC CL6' ' 
DC CL1' ' 
MT DC CL2' ' 
DC CL1' ' 
TKN DC CL20' ' CONVERTED FROM BL4
DC CL1' ' 
VT DC CL2' ' 
DC CL1' ' 
BT DC CL2' ' 
DC CL1' ' 
FLGS DC CL20' ' INTERPRETED
ORG

SMFDSECT DSECT
    IFASMFR (85) THIS INCLUDES CBRSMF MACRO
END
Step 3: Store the JCL to assemble and link the source in the PDS
Cut and paste the contents of Figure B-24 into your PDS MHLRES1.SMF85TO.SOURCE as member SMFT85OJ. The result should contain 33 lines.

```
// /*PRIORITY 15
// MHLRES10 JOB (1234567,COMMENT),MHLRES1,TIME=10,
// MSGCLASS=J,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=MHLRES1
// ASMHCL PROC
//ASM EXEC PGM=ASMA90,REGION=0M,
// PARM='OBJECT,NODECK'
//SYSIN DD DSN=SYS1.SAMPLIB(IEFES0),DISP=SHR
//SYSLIN DD DSN=&OBJ,DISP=(NEW,PASS),UNIT=SYSDA,
// SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120
//SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB
//SYSPRINT DD
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
//*
//LKED EXEC PGM=HEWL,REGION=2048K,COND=(8,LE,ASM),
// PARM='XREF,LIST,LET'
//SYSLIN DD DSN=&OBJ,DISP=(OLD,DELETE)
// DD DDNAME=SYSIN
//SYSLMOD DD DSN=&&LOADMOD(IEFES0),DISP=(MOD,PASS),UNIT=SYSDA,
// SPACE=(1024,(50,20,1))
//SYSPRINT DD
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
// PEND
// EXEC ASMHCL
//ASM.SYSIN DD DISP=SHR,DSN=MHLRES1.SMF85TO.SOURCE(SMF85TOA)
/*
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TO.LOAD,DISP=(,CATLG,DELETE),
// UNIT=SYSDA,SPACE=(CYL,(1,1,1))
//LKED.SYSIN DD *
SETSSI 00001800
NAME SMF85TO(R)
```

*Figure B-24  SMF85TO JCL to assemble and link the program*

This will create data set MHLRES1.SMF85TO.LOAD.

**Note:** If this JCL needs to be rerun, change the lines:

```
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TO.LOAD,DISP=(,CATLG,DELETE),
// UNIT=SYSDA,SPACE=(CYL,(1,1,1))
```

To read:

```
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TO.LOAD,DISP=SHR (,CATLG,DELETE),
// * UNIT=SYSDA,SPACE=(CYL,(1,1,1))
```

To stop it from trying to make the data set again.
B.4 SMF Record type 85 subtype 40 data display program

Program SMG85TJ displays the contents of selected fields of SMF record Type 85 subtype 40 data. It is not intended to provide a comprehensive report on OAM activity but rather to verify that immediate backup is occurring.

There are three steps to build the program which needs to be done once, after which it can be executed several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

Step 1: Create a PDS/PDSE to hold the members
In this example the PDS is called MHLRES1.SMF85TJ.SOURCE.

The LRECL/RECFM must be 80/FB—all other attributes can be chosen by you.

Step 2: Store the program source in the PDS
Cut and paste the contents of Figure B-25 on page 469 and all subsequent ones through to Figure B-28 on page 472, one after the other, into member SMFT85JA. The result should contain 173 lines.
Figure B-25   SMF85TJ assembler source (part 1 of 4)
* OTHER REGISTER USES
  * R12  OVERALL BASE REGISTER
  * R8   RECORD TYPE/SUBTYPE CHECKING/WORKING
  * R9   LENGTH OF PARTICULAR DSECT
  * R10  NUMBER OF ENTRIES IN THE TRIPLET
  *
* QSAM GET LOCATE PROCESSING IS USED
  *
* OPEN SMFIN 00600000
  OPEN (PRINTDCB,(OUTPUT)) 00610000
  PUT PRINTDCB,PRINTHDR 00620000
READ GET SMFIN 00630000
* COPY PARAMETER POINTER 00640000
  LR R3,R1 00650000
* R3 -> SMF RECORD 00660000
  USING CBRSMF85,R3 00670000
* CHECK IF TYPE 85 00680000
  CLI SMF85RTY,X'55' 00690000
  BNE IGNORE 00700000
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS 00710000
  CHKSTYP1 DS OH 00720000
* CHECK IF SUBTYPE 40 00730000
  CLI SMF85STY+1,X'28' 00740000
  BNE IGNORE 00750000
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS 00760000
  CHKSTYP1 DS OH 00770000
* IS TYPE 85 SUBTYPE 40, SO EXTRACT DATA 00780000
* R3 IS THE START OF THE WHOLE RECORD 00790000
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS. 00800000
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION 00810000
* ADD R8 TO R3 00820000
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS 00830000
  LA R4,SMF85END 00840000
  USING ST40,R4 00850001
  L R8,SMF85OSO 00860000
  LH R9,SMF85OSL 00870000
  LH R10,SMF85OSN 00880000
* PROCESS THE SUMMARY ENTRIES TRIPLET. 00890000
* FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START 00900000
* SECOND HW IS THE LENGTH OF EACH TRIPLET 00910000
* THIRD HW IS THE NUMBER OF TRIPLETS 00920000
* FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM 00930000
* THE SMF RECORD TYPE 85 SUBTYPE 40 RECORDS. 00940000
* STRD COMES FROM ST40STRD 00950000
* ENDD COMES FROM ST40ENDD 00960000
* ETC 00970000
* 00980000

Figure B-26  SMF85TJ assembler source (part 2 of 4)
SCOTRIP DS OH 00990000
LA R4,0(R3,R8) 01000000
LA R5,ST40END POINT TO THE VOLUME ARRAY 01010007
USING ST40VOLD,R5 01020007
* MOVE STRD
MVC STRD,ST40STRD 01030001
* MOVE ENDD
MVC ENDD,ST40ENDD 01040001
* CONVERT VOLN
LH R1,ST40VOLN 01050001
LR R6,R1 SAVE FOR LATER WHEN PRINTING THE VOLS 01060001
CVD R1,DWORD 01070001
OI DWORD+7,X'0F' 01080001
UNPK VOLN(11),DWORD+2(6) 01090001
* CONVERT PCTV
LH R1,ST40PCTV 01100001
CVD R1,DWORD 01110001
OI DWORD+7,X'0F' 01120001
UNPK PCTV(11),DWORD+2(6) 01130001
* CONVERT LIM
LH R1,ST40LIM 01140001
CVD R1,DWORD 01150001
OI DWORD+7,X'0F' 01160001
UNPK LIM(11),DWORD+2(6) 01170001
MVC SUBL,ST40SUBL 01180001
PUT PRINTDCB,PRINTL1 01190001
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS 01200001
VLOOP DS OH 01210001
MVC VSN,ST40VSN 01220001
PUT PRINTDCB,PRINTL2 01230001
LA R5,6(R5) 01240001
BCT R6,VLOOP 01250001
WRITEIT DS OH 01260001
PUT PRINTDCB,PRINTBLK 01270001
* LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS 01280001
* WHEN BCT REACHES ZERO GO GET ANOTHER RECORD 01290001
LA R8,0(R8,R9) 01300001
BCT R10,SCOTRIP 01310001
B READ 01320001
IGNORE DS OH EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS 01330001
B READ 01340001
FINISH DS OH 01350001
SENEG 01360001
SMFIN DCB DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP, 01370001
EODAD=FINISH 01380001
PRINTDCB DCB DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133 01390001
DWORD DS D 01400001
ORG DWORD 01410001
DC C'12345678' 01420001
PRINTBLK DC CL133' ' 01430001

Figure B-27  SMF85TJ assembler source (part 3 of 4)
| PRINTHDR DC | CL133’1SMF TYPE 85 SUBTYPE 40 RECORDS' | 01480001 |
| PRINTL1 DC | CL133' STRD/ENDD/VOLN/PCTV/LIM/SUBL:' | 01490011 |
| ORG | PRINTL1+30 | 01500014 |
| STRD | DC | CL10' ' | 01510001 |
| DC | C' ' | 01520000 |
| ENDD | DC | CL10' ' | 01530001 |
| DC | C' ' | 01540001 |
| VOLN | DC | CL12' ' CONVERTED FROM BINARY | 01550009 |
| DC | C' ' | 01560001 |
| PCTV | DC | CL12' ' CONVERTED FROM BINARY | 01570009 |
| DC | C' ' | 01580001 |
| LIM | DC | CL12' ' CONVERTED FROM BINARY | 01590009 |
| DC | C' ' | 01591011 |
| SUBL | DC | CL1' ' ASIS | 01592011 |
| ORG | | 01600000 |
| * | | 01610000 |
| PRINTL2 DC | CL133' VSN:' | 01620004 |
| ORG | PRINTL2+5 | 01630004 |
| VSN | DC | CL6' ' | 01640004 |
| ORG | | 01650000 |
| * | | 01660000 |
| SMFDSECT DSECT | (85) THIS INCLUDES CBRSMF MACRO | 01670000 |
| IFASMFR | END | 01680000 |
| | END | 01690000 |

*Figure B-28 SMF8STJ assembler source (part 4 of 4)*
Step 3: Store the JCL to assemble and link the source in the PDS
Cut and paste the contents of Figure B-29 into your PDS MHLRES1.SMF85TJ.SOURCE as member SMFT85JJ. The result should contain 32 lines.

Run the job when the member has been created.

```
//MHLRES1O JOB (1234567,COMMENT),MHLRES1,TIME=10,
// MSGCLASS=J,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=MHLRES1
/*JOBPARM S=* 
//ASMHCL PROC 
//ASM EXEC PGM=ASMA90,REGION=0M,
// PARM='OBJECT,NODECK'
//SYSIN DD DSN=SYS1.SAMPLIB(IEFESO),DISP=SHR 
//SYSLIN DD DSN=&OBJ,DISP=(NEW,PASS),UNIT=SYSDA,
// SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120 
//SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB 
//SYSPRINT DD SYOUT=* 
//SYSLIN DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5)) 
/* 
//LKED EXEC PGM=HEWL,REGION=2048K,COND=(8,LE ASM),
// PARM='XREF,LST,LK'
//SYSLIN DD DSN=&OBJ,DISP=(OLD,DELETE)
// DD DDNAME=SYSIN 
//SYSLMOD DD DSN=&&LOADMOD(IEFESO),DISP=(MOD,PASS),UNIT=SYSDA,
// SPACE=(1024,(50,20,1)) 
//SYSPRINT DD SYOUT=* 
//SYSLIN DD DSN=&&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5)) 
//PEND 
// EXEC ASMHCL 
//ASM.SYSIN DD DISP=SHR,DSN=MHLRES1.SMF85TJ.SOURCE(SMF85TJ) 
/* 
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TJ.LOAD,DISP=SHR TLG,DELETE), 
// UNIT=SYSALLDA,SPACE=(CYL,(1,1,1)) 
//LKED.SYSIN DD * 
SETSSI 00001800 
NAME SMF85TJ(R) 
```

Figure B-29 SMF85TJ JCL to assemble and link the program

This will create data set MHLRES1.SMF85TJ.LOAD.

**Note:** If this JCL needs to be rerun, change the lines:

```
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TJ.LOAD,DISP=(,CATLG,DELETE), 
// UNIT=SYSDA,SPACE=(CYL,(1,1,1)) 
```

To read:

```
//LKED.SYSLMOD DD DSN=MHLRES1.SMF85TJ.LOAD,DISP=SHR (,CATLG,DELETE), 
/* UNIT=SYSDA,SPACE=(CYL,(1,1,1)) 
```

To stop it from trying to make the data set again.
SMF examples to go in their main sections DFSMS V1R9

In this section we provide SMF examples for DFSMS V1R9.

SMF Record 85 Subtype 1-7 Records for OAM OSREQ macro usage

OAM writes SMF Record type 85 subtype 2/3/6 to document the OSREQ macro’s use of the LOB support.

We have written a simple program called SMF85TA to scan the SMF records and summarize activity. The program itself and how to construct it is documented in B.1, “SMF Record type 85 subtype 1-7 data display program” on page 442.

In Figure B-30 we show the JCL to extract the SMF records and run the program.

If you do not want output from all the types that the program can process, change the SMF selection statement to only include those subtypes that you do want.

For example, change:

```
OUTDD(OUTDD,TYPE(85(1,2,3,4,5,6,7)))
```

To:

```
OUTDD(OUTDD,TYPE(85(2,3,4,5,6)))
```

To only select subtypes 2-6 and exclude subtypes 1 and 7.

**Note:** This JCL assumes that an extract is being taken from the active SMF data set which is then passed to the SMFT85I program. The JCL could be changed to make a permanent extract of the SMF data, or to read an already created SMF data extract.

```
//MHLRES1O JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES1
//SMFEXTR EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=* 
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1 
//OUTDD DD DSN=&SMFT85, 
// SPACE=(CYL,(10,5)), 
// RECFM=VB,LRECL=5096, 
// DISP=(,PASS,DELETE), 
// UNIT=SYSDA 
//SYSIN DD * 
// INDD(DUMPIN,OPTIONS(DUMP))
// OUTDD(OUTDD,TYPE(85(1,2,3,4,5,6,7)))
/*
 // EXEC PGM=SMF85TA
 //STEPLIB DD DISP=SHR,DSN=MHLRES1.SMF85TA.LOAD
 //SYSUDUMP DD SYSOUT=A 
 //SMFIN DD DISP=SHR,DCB=BFTTEK=A, 
 // DSN=&SMFT85 
 //PRINT DD SYSOUT=A,RECFM=UA
```

Figure B-30  SMF85TA program execution JCL
The SMF data is mapped by the CBRSMF macro usually found in SYS1.MACLIB. The program uses, where possible, similar field names as in the CBRSMF macro so the user can check the description there at any time. In DFSMS V1R9, updates were made to the SMF records to expand the bit definitions in the FLAG fields.

After executing the program, the output can be reviewed in conjunction with the contents of the current CBRSMF. Extracts from CBRSMF are provided in Figure B-34 on page 479 through Figure B-38 on page 483.

**SMF85TA Program execution results**

In Figure B-31 on page 476 and Figure B-32 on page 477 we show an example of output from execution of program SMF85TA with all records selected in the SMFEXTR step.

A typical sequence of events is illustrated by Subtype 1 OSREQ ACCESS, some activity, and ending with subtype 7 OSREQ UNACCESS. In the case of ACCESS and UNACCESS there is no data to track. All related activities use the same TOKen, for example D6E2D4C97F690A00.

In the examples shown, the activity tracked is Subtype 2 OS STORE. In this case, the length of the data is recorded as 00052428550, and related to the activity, FLAGs are recorded as 80400000. Using information from the CBRSMF macro we were able to analyze the meaning of the flags. From the data relating to ST2FLAGS we see that 80 indicates ST2FLGS0 is on and 40 indicates that STFLGS9 is on. These mean:

- **ST2FLGS0** - OBJECT STORE TO DASD
- **ST2FLGS9** - WHEN ON, THE OBJECT IS STORED TO LOB STORAGE STRUCTURE

A similar process can be followed to check that the OAM functions are performing as expected as far as the OSREQ macro are concerned.

After the OAM cycle of ACCESS, activity, UNACCESS has been established, the ACCESS and UNACCESS records can be omitted. An example of this follows the full example.
<table>
<thead>
<tr>
<th>SMFDTE/TME:</th>
<th>2008073 16:24:00.047</th>
</tr>
</thead>
<tbody>
<tr>
<td>STYPE:</td>
<td>1 OSREQ ACCESS</td>
</tr>
<tr>
<td>COLN/OBJN:</td>
<td></td>
</tr>
<tr>
<td>SGN/SCN/MCN/LEN/TTOP/TOK:</td>
<td>00000000000000000000000000000000 D6E2D4C97F690A00</td>
</tr>
<tr>
<td>VSN/MT/RS/FLGS:</td>
<td>00000000 00000000 00000000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SMFDTE/TME:</th>
<th>2008073 16:24:00.098</th>
</tr>
</thead>
<tbody>
<tr>
<td>STYPE:</td>
<td>2 OSREQ STORE</td>
</tr>
<tr>
<td>COLN/OBJN:</td>
<td>TOGRP1.COLSTD.TL2TST</td>
</tr>
<tr>
<td>SGN/SCN/MCN/LEN/TTOP/TOK:</td>
<td>TOGRP1 OBJDASD TOGRP1D1 00052428550 D6E2D4C97F690A00</td>
</tr>
<tr>
<td>VSN/MT/RS/FLGS:</td>
<td>00000000 00000000 80400000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SMFDTE/TME:</th>
<th>2008073 16:24:01.004</th>
</tr>
</thead>
<tbody>
<tr>
<td>STYPE:</td>
<td>7 OSREQ UNACCESS</td>
</tr>
<tr>
<td>COLN/OBJN:</td>
<td></td>
</tr>
<tr>
<td>SGN/SCN/MCN/LEN/TTOP/TOK:</td>
<td>00000000000000000000000000000000 D6E2D4C97F690A00</td>
</tr>
<tr>
<td>VSN/MT/RS/FLGS:</td>
<td>00000000 00000000 80400000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SMFDTE/TME:</th>
<th>2008073 16:24:02.005</th>
</tr>
</thead>
<tbody>
<tr>
<td>STYPE:</td>
<td>1 OSREQ ACCESS</td>
</tr>
<tr>
<td>COLN/OBJN:</td>
<td></td>
</tr>
<tr>
<td>SGN/SCN/MCN/LEN/TTOP/TOK:</td>
<td>00000000000000000000000000000000 D6E2D4C97F690A00</td>
</tr>
<tr>
<td>VSN/MT/RS/FLGS:</td>
<td>00000000 00000000 80400000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SMFDTE/TME:</th>
<th>2008073 16:24:02.055</th>
</tr>
</thead>
<tbody>
<tr>
<td>STYPE:</td>
<td>2 OSREQ STORE</td>
</tr>
<tr>
<td>COLN/OBJN:</td>
<td>TOGRP1.COLSTD.TL2TST</td>
</tr>
<tr>
<td>SGN/SCN/MCN/LEN/TTOP/TOK:</td>
<td>TOGRP1 OBJDASD TOGRP1D1 00052428200 D6E2D4C97F690A00</td>
</tr>
<tr>
<td>VSN/MT/RS/FLGS:</td>
<td>00000000 00000000 80400000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SMFDTE/TME:</th>
<th>2008073 16:24:02.058</th>
</tr>
</thead>
<tbody>
<tr>
<td>STYPE:</td>
<td>7 OSREQ UNACCESS</td>
</tr>
<tr>
<td>COLN/OBJN:</td>
<td></td>
</tr>
<tr>
<td>SGN/SCN/MCN/LEN/TTOP/TOK:</td>
<td>00000000000000000000000000000000 D6E2D4C97F690A00</td>
</tr>
<tr>
<td>VSN/MT/RS/FLGS:</td>
<td>00000000 00000000 80400000</td>
</tr>
</tbody>
</table>

Figure B-31   SMF85TA output - all subtypes part (1 of 2)
Figure B-32   SMF85TA output - all subtypes part (2 of 2)
In Figure B-33 we show an example of output from execution of program SMF85TA with records 1 and 7 not selected in the SMFEXTR step. Using this method to leave out the OSREQ ACCESS and UNACCESS data serves to reduce the amount of data that appears in your output, thereby simplifying things.

In Figure B-34 on page 479 to Figure B-38 on page 483 we show the SMF records that relate to this output. The Flags field as shown in the output reflects the flag bits as mapped by the flag fields in the SMF records corresponding to the particular subtype.

SMF record type 85 subtype 1 is used to map subtype records 1 and related subtypes 2, 3, 4, 5, 6, and 7.

Note: You should not use the contents of the CBRSMF macro as presented here. When you assemble the program, or wish to refer to the macro, it will be found in SYS1.MACLIB and will contain the most recent version.
ST1 DSECT SUBTYPES 1 - 7
STICOLN DS CL44' ' COLLECTION NAME
STIOBJN DS CL44' ' OBJECT NAME
STISGN DS CL8' ' STORAGE GROUP NAME
STISCN DS CL8' ' STORAGE CLASS NAME
STIMCN DS CL8' ' MANAGEMENT CLASS NAME
STIOFF DS BL4'0' OFFSET FOR PARTIAL OBJECT
* RETRIEVE (SUBTYPE 3), ZERO FOR ALL OTHERS.
* STILEN DS BL4'0' LENGTH,
* SUBTYPE 1 - UNUSED
* SUBTYPE 2 - LENGTH OF OBJECT STORED
* SUBTYPE 3 - NUMBER OF BYTES RETRIEVED
* SUBTYPE 4 - NUMBER OF QEL ELEMENTS RETURNED.
* SUBTYPE 5 - UNUSED
* SUBTYPE 6 - LENGTH OF OBJECT DELETED
* SUBTYPE 7 - UNUSED
STITOK DS CL16' ' OSREQ TRACKING TOKEN, SUPPLIED
* WITH TTOKEN KEYWORD ON OSREQ
* MACRO
STITOK DS CL8' ' OSREQ ACCESS TOKEN
STIVSN DS CL6' ' VOLUME SERIAL NUMBER
STIVMT DS CL2' ' VOLUME MEDIA TYPE
STIRC DS BL4'0' OSREQ RETURN CODE, IN REGISTER 15
* FOLLOWING OSREQ MACRO
STIRS DS BL4'0' OSREQ REASON CODE, IN REGISTER 15
* FOLLOWING OSREQ MACRO
STIFLGS DS BL4'0' PROCESSING FLAGS, MEANING
* DEPENDENT ON RECORD SUBTYPE.

Figure B-34 SMF record type 85 subtype 1 significant fields (extract from CBRSMF macro) (1 of 5)
Figure B-35   SMF record type 85 subtype 1 significant fields (extract from CBRSMF macro) (2 of 5)

***********************************************************************
* SUBTYPE 2 - OSREQ STORE FLAGS
* ***********************************************************************
ST2FLGS0 EQU X'80' OBJECT STORE TO DASD
ST2FLGS1 EQU X'40' OBJECT STORE TO OPTICAL
ST2FLGS2 EQU X'20' OBJECT STORE TO TAPE
ST2FLGS3 EQU X'10' UNUSED
ST2FLGS4 EQU X'08' UNUSED
ST2FLGS5 EQU X'04' WHEN ON, THE OSREQ STORE
* REQUEST RESULTED IN THE MOUNTING
* OF A SHELF-RESIDENT REMOVABLE
* MEDIA VOLUME (TAPE OR OPTICAL)
* BY A HUMAN OPERATOR. ONLY VALID
* IF BIT 1 OR 2 IS ON.
ST2FLGS6 EQU X'02' WHEN ON, THE OSREQ STORE
* REQUEST RESULTED IN THE MOUNTING
* OF A LIBRARY-RESIDENT REMOVABLE
* MEDIA VOLUME (TAPE OR OPTICAL)
* BY A HUMAN OPERATOR. ONLY VALID
* IF BIT 1 OR 2 IS ON.
ST2FLGS7 EQU X'01' WHEN ON, THE OSREQ STORE
* REQUEST WAS SATISIFIED USING
* AN ALREADY MOUNTED REMOVEABLE
* MEDIA VOLUME (TAPE OR OPTICAL).
* ONLY VALID
* IF BIT 1 OR 2 IS ON.
ST2FLGS8 EQU X'80' WHEN ON, AN IMMEDIATE BACKUP COPY
* WAS SCHEDULED FOR THIS OBJECT. @L5A
ST2FLGS9 EQU X'40' WHEN ON, THE OBJECT IS STORED
* TO LOB STORAGE structure @L5A
ST2FLGS10 EQU X'20' WHEN ON, OBJECT STORED ON SUBLEVEL 1
* VOLUME. @L6A
ST2FLGS11 EQU X'10' WHEN ON, OBJECT STORED ON SUBLEVEL 2
* VOLUME. @L6A
**SUBTYPE 3 - OSREQ RETRIEVE FLAGS**

<table>
<thead>
<tr>
<th>Flag Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'80'</td>
<td>Primary copy of object retrieved from DASD.</td>
</tr>
<tr>
<td>X'40'</td>
<td>Primary copy of object retrieved from optical.</td>
</tr>
<tr>
<td>X'20'</td>
<td>Primary copy of object retrieved from tape.</td>
</tr>
<tr>
<td>X'10'</td>
<td>Either the first or the second backup copy of the object was retrieved from optical as result of <code>VIEW=BACKUP</code> or <code>VIEW=BACKUP2</code> being specified on the OSREQ macro. Refer to bit 10 to indicate which backup copy was retrieved.</td>
</tr>
<tr>
<td>X'08'</td>
<td>Either the first or the second backup copy of the object was retrieved from tape as result of <code>VIEW=BACKUP</code> or <code>VIEW=BACKUP2</code> being specified on the OSREQ macro. Refer to bit 10 to indicate which backup copy was retrieved.</td>
</tr>
<tr>
<td>X'04'</td>
<td>Either the first or the second backup copy of the object was retrieved from optical as a result of the primary copy of the object residing on an unreadable optical disk volume and the automatic access to backup was active. Refer to bit 10 to indicate which backup copy was retrieved.</td>
</tr>
<tr>
<td>X'02'</td>
<td>Either the first or the second backup copy of the object was retrieved from tape as a result of the primary copy of the object residing on an unreadable optical disk volume and the automatic access to backup was active. Refer to bit 10 to indicate which backup copy was retrieved.</td>
</tr>
<tr>
<td>X'01'</td>
<td>Request resulted in the mounting of a shelf-resident removable media volume (tape or optical) by a human operator. Only valid if bit 1, 2, 3, 5 or 6 is on.</td>
</tr>
</tbody>
</table>

*Figure B-36  SMF record type 85 subtype 1 significant fields (extract from CBRSMF macro) (3 of 5)*
**ST3FLGS8 EQU X'80'**  
*WHEN ON, THE OSREQ RETRIEVE REQUEST RESULTED IN THE MOUNTING OF A LIBRARY-RESIDENT REMOVABLE MEDIA VOLUME (TAPE OR OPTICAL) BY A HUMAN OPERATOR. ONLY VALID IF BIT 1, 2, 3, 5 OR 6 IS ON.*

**ST3FLGS9 EQU X'40'**  
*WHEN ON, THE OSREQ RETRIEVE REQUEST WAS SATISIFIED USING AN ALREADY MOUNTED REMOVEABLE MEDIA VOLUME (TAPE OR OPTICAL). ONLY VALID IF BIT 1, 2, 3, 5 OR 6 IS ON.*

**ST3FLGS10 EQU X'20'**  
*WHEN ON, THE SECOND BACKUP COPY OF THE OBJECT WAS RETRIEVED. @L4C*

**ST3FLGS11 EQU X'10'**  
*WHEN ON, A RECALL WAS SCHEDULED FOR THIS OBJECT @L4A*

**ST3FLGS12 EQU X'08'**  
*WHEN ON, A RECALL WAS EXPLICITLY SPECIFIED ON THE OSREQ RETRIEVE REQUEST @L4A*

**ST3FLGS13 EQU X'04'**  
*WHEN ON, THE PRIMARY COPY OF THE OBJECT WAS RETRIEVED FROM A LOB STORAGE STRUCTURE @L5A*

**ST3FLGS14 EQU X'02'**  
*WHEN ON, OBJECT RETR. FROM SUBLEVEL 1 VOLUME. @L6A*

**ST3FLGS15 EQU X'01'**  
*WHEN ON, OBJECT RETR. FROM SUBLEVEL 2 VOLUME. @L6A*

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

**SUBTYPE 4 - OSREQ QUERY FLAGS**

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

**ST5FLG0 EQU X'80'**  
*WHEN ON, MANAGEMENT CLASS SPECIFIED ON OSREQ CHANGE.*

**ST5FLG1 EQU X'40'**  
*WHEN ON, STORAGE CLASS SPECIFIED ON OSREQ CHANGE.*

**ST5FLG2 EQU X'20'**  
*WHEN ON, RETENTION PERIOD SPECIFIED ON OSREQ CHANGE.*

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

**SUBTYPE 5 - OSREQ CHANGE FLAGS**

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

* Figure B-37  SMF record type 85 subtype 1 significant fields (extract from CBRSMF macro) (4 of 5)*
OAM also writes SMF Record type 85 subtype 32/33/34/35 to document the use of its functions. In DFSMS V1R9 the tape support has been extended to provide two levels referred to as sublevel 1 and sublevel 2. Sublevel 1 is what used to be provided, and sublevel 2 is the new capability.

The SMF Record type 85 subtype 32/33/34/25 have been expanded to track use across all the storage hierarchies.

We have written a simple program called SMF85TH to scan the SMF records and summarize activity. The program itself and how to construct it is documented in “SMF Record type 85 subtype 32-35 data display program” on page 451.

---

**SMF Record 85 Subtype 32-35 Records for OAM OSMC usage**

OAM also writes SMF Record type 85 subtype 32/33/34/35 to document the use of its functions. In DFSMS V1R9 the tape support has been extended to provide two levels referred to as sublevel 1 and sublevel 2. Sublevel 1 is what used to be provided, and sublevel 2 is the new capability.

The SMF Record type 85 subtype 32/33/34/25 have been expanded to track use across all the storage hierarchies.

We have written a simple program called SMF85TH to scan the SMF records and summarize activity. The program itself and how to construct it is documented in “SMF Record type 85 subtype 32-35 data display program” on page 451.
In Figure B-39 we show the JCL to extract the SMF records and run the program.

**Note:** This JCL assumes that an extract is being taken from the active SMF data set, which is then passed to the SMFT85I program. The JCL could be changed to make a permanent extract of the SMF data, or to read of an already created SMF data extract.

If you do not want output from all the types that the program can process, change the SMF selection statement to only include those subtypes that you do want.

For example, change

```
OUTDD(OUTDD,TYPE(85(32,33,34,35)))
```

To

```
OUTDD(OUTDD,TYPE(85(35)))
```

To only select subtype 35.

```
//MHLRES1O JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES1
//SMFEXTR EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=* 
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1
//OUTDD DD DSN=&SMFT85,
// SPACE=(CYL,(10,5)),
// RECFM=VB,LRECL=5096,
// DISP=(,PASS,DELETE),
// UNIT=SYSDA
//SYSIN DD *
  INDD(DUMPIN,OPTIONS(DUMP))

/*
 // EXEC PGM=SMF85TH
 //STEPLIB DD DISP=SHR,DSN=MHLRES1.SMF85TH.LOAD
 //SYSUDUMP DD SYSOUT=A
 //SMFIN DD DISP=SHR,DCB=BFTEK=A,
 //DSN=&SMFT85
 //PRINT DD SYSOUT=A,RECFM=UA
 */
```

**Figure B-39**  SMF85TH program execution JCL

In Figure B-40 on page 485 and Figure B-41 on page 486 we show an example of output from execution of program SMF85TH. You can see examples of subtypes 32 and 33. In the subtype 32 records in support of the tape sublevel 2 support, data is shown in the PUWO and PUWK fields indicating the number of objects written to TSL2 (4) and the number of kilobytes (204800) written to TSL2.
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUWO/PUWK/PURO/PURK/PUDO/PUDK:</td>
<td>00000000010 00000512000 00000000000 00000000000 00000000000 00000000000</td>
</tr>
<tr>
<td>FLGS/NTE/RCLD/RCLK/LOBI/LOBD:</td>
<td>41000000 0000000 0000000 0000000 0000000 0000010</td>
</tr>
<tr>
<td>BOWO/BOWK/BORO/BORK/BODO/BODK:</td>
<td>0000000 0000000 0000000 0000000 0000000 0000000</td>
</tr>
<tr>
<td>PTWO/PTWK/PTRO/PTRD/PTDO/PTDK:</td>
<td>0000000 0000000 0000000 0000000 0000000 0000000</td>
</tr>
<tr>
<td>POWO/POWK/PORO/PORD/PODO/PODK:</td>
<td>0000000 0000000 0000000 0000000 0000000 0000000</td>
</tr>
<tr>
<td>PDWO/PDWK/PDRO/PDRK/PDDO/PDDK:</td>
<td>0000000 0000000 0000000 0000000 0000000 0000000</td>
</tr>
<tr>
<td>STYPE/SGN/VSN0/VSN1/MT:</td>
<td>32 TOGRP1 (STORAGE GROUP PROCESSING)</td>
</tr>
<tr>
<td>SMFDTE/TME:</td>
<td>2008073 15:36:19.084</td>
</tr>
</tbody>
</table>

Figure B-40 SMF85TH output (part 1 of 2)
### SMF85TH Output (Part 2 of 2)

#### SMFDT/TME:
```
2008073 16:16:16.077
```

#### STYPE/SGN/VSN1/VSN1/MT:
```
32 TOGRP1 (STORAGE GROUP PROCESSING)
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### PDU/PK/PDRK/PDRK/PDDK/PDDK:
```
000000 000000 000000 000000 000000 000000
```

#### Figure B-41

In Figure B-42 on page 487 to Figure B-46 on page 491 we show the SMF records that relate to this output.

---

DFSMS V1.10 and EAV Technical Guide

486
SMF record type 85 subtype 32 is used to map subtype records 32, 33, 34, and 35.

**Note:** You should not use the contents of the CBRSMF macro as presented here. When you assemble the program, or want to refer to the macro, it will be found in SYS1.MACLIB.

---

```
1 DSECT SUBTYPES 32 - 35
ST32SGN DS CL8' ' STORAGE GROUP NAME
ST32VSNO DS CL6' ' VOLUME SERIAL NUMBER OF OPTICAL
  VOLUME. ONLY VALID FOR SUBTYPES
  34 AND 35, CONTAINS BLANKS FOR
  OTHER SUBTYPES.
ST32VSN1 DS CL6' ' VOLUME SERIAL NUMBER OF OPPOSITE
  SIDE OF OPTICAL DISK
  VOLUME. ONLY VALID FOR SUBTYPES
  34 AND 35, CONTAINS BLANKS FOR
  OTHER SUBTYPES.
ST32OMT DS CL2' ' OPTICAL MEDIA TYPE. ONLY VALID
  FOR SUBTYPES 34 AND 35, CONTAINS
  BLANKS FOR OTHER SUBTYPES.
  RESERVED
***********************************************************************
* COUNTS OF PRIMARY OBJECTS (AND KILOBYTES) WRITTEN, READ AND
* DELETED FROM DASD.
***********************************************************************
ST32PDWO DS BL4'0' NUMBER OF PRIMARY OBJECTS
  WRITTEN TO DASD.
ST32PDWK DS BL4'0' NUMBER OF KILOBYTES PRIMARY
  OBJECTS WRITTEN TO DASD.
ST32PDRO DS BL4'0' NUMBER OF PRIMARY OBJECTS
  READ FROM DASD.
ST32PDRK DS BL4'0' NUMBER OF KILOBYTES PRIMARY
  OBJECTS READ FROM DASD.
ST32PDDO DS BL4'0' NUMBER OF PRIMARY OBJECTS
  DELETED FROM DASD.
ST32PDDK DS BL4'0' NUMBER OF KILOBYTES PRIMARY
  OBJECTS DELETED FROM DASD.
***********************************************************************
* COUNTS OF PRIMARY OBJECTS (AND KILOBYTES) WRITTEN, READ AND
* DELETED FROM OPTICAL.
***********************************************************************
ST32POWO DS BL4'0' NUMBER OF PRIMARY OBJECTS
  WRITTEN TO OPTICAL.
ST32POWK DS BL4'0' NUMBER OF KILOBYTES PRIMARY
  OBJECTS WRITTEN TO OPTICAL.
ST32PORO DS BL4'0' NUMBER OF PRIMARY OBJECTS
  READ FROM OPTICAL.
ST32PORK DS BL4'0' NUMBER OF KILOBYTES PRIMARY
  OBJECTS READ FROM OPTICAL.
ST32PODO DS BL4'0' NUMBER OF PRIMARY OBJECTS
  DELETED FROM OPTICAL.
ST32PODK DS BL4'0' NUMBER OF KILOBYTES PRIMARY
  OBJECTS DELETED FROM OPTICAL.
```

*Figure B-42  SMF record type 85 subtype 32 significant fields (extract from CBRSMF macro) (1 of 5)*
***********************************************************************
* COUNTS OF PRIMARY OBJECTS (AND KILOBYTES) WRITTEN, READ AND
* DELETED FROM TAPE.
***********************************************************************
ST32PTWO DS  BL4'0'  NUMBER OF PRIMARY OBJECTS
  * WRITTEN TO TAPE.
ST32PTWK DS  BL4'0'  NUMBER OF KILOBYTES OF PRIMARY
  * OBJECTS WRITTEN TO TAPE.
ST32PTRO DS  BL4'0'  NUMBER OF PRIMARY OBJECTS
  * READ FROM TAPE.
ST32PTRK DS  BL4'0'  NUMBER OF KILOBYTES OF PRIMARY
  * OBJECTS READ FROM TAPE.
ST32PTDO DS  BL4'0'  NUMBER OF PRIMARY OBJECTS
  * LOGICALLY DELETED FROM TAPE.
ST32PTDK DS  BL4'0'  NUMBER OF KILOBYTES OF PRIMARY
  * OBJECTS LOGICALLY DELETED FROM
  * TAPE.
***********************************************************************
* COUNTS OF BACKUP OBJECTS (AND KILOBYTES) WRITTEN, READ AND
* DELETED FROM OPTICAL.
***********************************************************************
ST32BOWO DS  BL4'0'  NUMBER OF BACKUP OBJECTS
  * WRITTEN TO OPTICAL.
ST32BOWK DS  BL4'0'  NUMBER OF KILOBYTES OF BACKUP
  * OBJECTS WRITTEN TO OPTICAL.
ST32BORO DS  BL4'0'  NUMBER OF BACKUP OBJECTS
  * READ FROM OPTICAL.
ST32BORK DS  BL4'0'  NUMBER OF KILOBYTES OF BACKUP
  * OBJECTS READ FROM OPTICAL.
ST32BODO DS  BL4'0'  NUMBER OF BACKUP OBJECTS
  * DELETED FROM OPTICAL.
ST32BODK DS  BL4'0'  NUMBER OF KILOBYTES OF BACKUP
  * OBJECTS DELETED FROM OPTICAL.
***********************************************************************
* COUNTS OF BACKUP OBJECTS (AND KILOBYTES) WRITTEN, READ AND
* DELETED FROM TAPE.
***********************************************************************
ST32BTWO DS  BL4'0'  NUMBER OF BACKUP OBJECTS
  * WRITTEN TO TAPE.
ST32BTWK DS  BL4'0'  NUMBER OF KILOBYTES OF BACKUP
  * OBJECTS WRITTEN TO TAPE.
ST32BTRO DS  BL4'0'  NUMBER OF BACKUP OBJECTS
  * READ FROM TAPE.
ST32BTRK DS  BL4'0'  NUMBER OF KILOBYTES OF BACKUP
  * OBJECTS READ FROM TAPE.
ST32BTDO DS  BL4'0'  NUMBER OF BACKUP OBJECTS
  * LOGICALLY DELETED FROM TAPE.
ST32BTDK DS  BL4'0'  NUMBER OF KILOBYTES OF BACKUP
  * OBJECTS LOGICALLY DELETED FROM
  * TAPE.
Figure B-44   SMF record type 85 subtype 32 significant fields (extract from CBRSMF macro) (3 of 5)
**Counts of Activity Against the Object Storage Database**
* (Object Directory Table, 4K Object Storage Table and 32K Object Storage Table).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST32DTUP</td>
<td>Number of rows updated in the Object Directory Table.</td>
<td>BL4'0'</td>
</tr>
<tr>
<td>ST32DTDE</td>
<td>Number of rows deleted from the Object Directory Table.</td>
<td>BL4'0'</td>
</tr>
<tr>
<td>ST324KIN</td>
<td>Number of rows inserted into the 4K Object Storage Table.</td>
<td>BL4'0'</td>
</tr>
<tr>
<td>ST324KDE</td>
<td>Number of rows deleted from the 4K Object Storage Table.</td>
<td>BL4'0'</td>
</tr>
<tr>
<td>ST3232KI</td>
<td>Number of rows inserted into the 32K Object Storage Table.</td>
<td>BL4'0'</td>
</tr>
<tr>
<td>ST3232KD</td>
<td>Number of rows deleted from the 32K Object Storage Table.</td>
<td>BL4'0'</td>
</tr>
<tr>
<td>ST32NCE</td>
<td>Number of optical cartridges expired. Valid only for subtype 32.</td>
<td>BL4'0'</td>
</tr>
<tr>
<td>ST32FLGS</td>
<td>Processing flags</td>
<td></td>
</tr>
<tr>
<td>ST32FLG0</td>
<td>When on, this process was invoked automatically under software control.</td>
<td>X'80'</td>
</tr>
<tr>
<td>ST32FLG1</td>
<td>When on, this process was invoked by a MODIFY OAM,START command.</td>
<td>X'40'</td>
</tr>
<tr>
<td>ST32FLG2</td>
<td>When on, this process was invoked using an ISMF Line Operator.</td>
<td>X'20'</td>
</tr>
<tr>
<td>ST32FLG3</td>
<td>When on, vol recovery was invoked w/ BACKUP1 keyword or defaulted to Backup1</td>
<td>X'10'</td>
</tr>
<tr>
<td>ST32FLG4</td>
<td>When on, vol recovery was invoked w/ BACKUP2 keyword</td>
<td>X'08'</td>
</tr>
<tr>
<td>ST32FLG5</td>
<td>When on, vol recovery or movevol was specified with delete option</td>
<td>X'04'</td>
</tr>
<tr>
<td>ST32FLG6</td>
<td>When on, vol recovery or movevol was specified with recycle option</td>
<td>X'02'</td>
</tr>
<tr>
<td>ST32FLG7</td>
<td>When on, indicated processing object storage group</td>
<td>X'01'</td>
</tr>
<tr>
<td>ST32FLG8</td>
<td>When on, indicated processing backup object storage group</td>
<td>X'80'</td>
</tr>
<tr>
<td>ST32FLG9</td>
<td>When on, cycle endtime exceeded</td>
<td>X'40'</td>
</tr>
</tbody>
</table>

*Figure B-45  SMF record type 85 subtype 32 significant fields (extract from CBRSMF macro) (4 of 5)*
SMF Record 85 Subtype 40 Records for OAM OSMC Command Recycle

OAM writes SMF Record type 85 subtype 40 to document the use of the RECYCLE facility.

We have written a simple program called SMF85TJ to scan the SMF records and summarize activity. The program itself and how to construct it is documented in B.4, “SMF Record type 85 subtype 40 data display program” on page 468.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Size</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST32NTE</td>
<td>Number of tape volumes expired</td>
<td>BL4</td>
<td>0L3A</td>
</tr>
<tr>
<td>ST32RCLD</td>
<td>Number of recalled objects</td>
<td>BL4</td>
<td>000A</td>
</tr>
<tr>
<td>ST32RCLK</td>
<td>Number of kilobytes of recalled objects</td>
<td>BL4</td>
<td>000A</td>
</tr>
<tr>
<td>ST32LOBI</td>
<td>Number of LOB rows inserted</td>
<td>BL4</td>
<td>000A</td>
</tr>
<tr>
<td>ST32LOBD</td>
<td>Number of LOB rows deleted</td>
<td>BL4</td>
<td>000A</td>
</tr>
<tr>
<td>ST32PUWO</td>
<td>Number of primary objects written to TSL2</td>
<td>BL4</td>
<td>0L6A</td>
</tr>
<tr>
<td>ST32PUWK</td>
<td>Number of kilobytes of primary objects written to TSL2</td>
<td>BL4</td>
<td>0L6A</td>
</tr>
<tr>
<td>ST32PURO</td>
<td>Number of primary objects read from TSL2</td>
<td>BL4</td>
<td>0L6A</td>
</tr>
<tr>
<td>ST32PURK</td>
<td>Number of kilobytes of primary objects read from TSL2</td>
<td>BL4</td>
<td>0L6A</td>
</tr>
<tr>
<td>ST32PUDO</td>
<td>Number of primary objects logically deleted from TSL2</td>
<td>BL4</td>
<td>0L6A</td>
</tr>
<tr>
<td>ST32PUOK</td>
<td>Number of kilobytes of primary objects logically deleted from TSL2</td>
<td>BL4</td>
<td>0L6A</td>
</tr>
</tbody>
</table>

Figure B-46   SMF record type 85 subtype 32 significant fields (extract from CBRSMF macro) (5 of 5)
In Figure B-47 we show the JCL to extract the SMF records and run the program.

**Note:** This JCL assumes that an extract is being taken from the active SMF data set which is then passed to the SMFT85I program. The JCL could be changed to make a permanent extract of the SMF data, or to read of an already created SMF data extract.

```plaintext
//MHLRES1O JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES1 
// EXEC PGM=IFASMFDP 
//SYSPRINT DD SYSOUT=* 
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN2 
//*DUMPIN DD DISP=SHR,DSN=SMFDATA.ALLRECS.G3196V00 
//OUTDD DD DSN=&SMFT85, 
// SPACE=(CYL,(10,5)), 
// RECFM=VB,LRECL=5096, 
// DISP=(,PASS,DELETE), 
// UNIT=SYSDA 
//SYSIN DD * 
   INDD(DUMPIN,OPTIONS(DUMP)) 
   OUTDD(OUTDD,TYPE(85(40))) 
/* 
// EXEC PGM=SMF85TI 
//STEPLIB DD DISP=SHR,DSN=MHLRES1.SMF85TI.LOAD 
//SYSUDUMP DD SYSOUT=A 
//SMFIN DD DISP=SHR,DCB=BFTEK=A, 
// DSN=&SMFT85 
//PRINT DD SYSOUT=A,RECFM=UA 
```

*Figure B-47  SMF85TJ program execution JCL*

In Figure B-48 we show an example of output from execution of program SMF85TI. This example shows that volume THS020 was successfully processed by the OSMC command migration facility and Tape Sublevel 2 was used.

```plaintext
SMF TYPE 85 SUBTYPE 40 RECORDS 
STRD/ENDD/VOLN/PCTV/LIM/SUBL:2008-03-13 2008-03-13 0000000001 0000000099 0000000001 2 
VSN:THS020 
```

*Figure B-48  SMF85TJ output*
In Figure B-49 we show the SMF records that relate to this output.

**Note:** You should not use the contents of the CBRSMF macro as presented here. When you assemble the program, or want to refer to the macro, it can be found in SYS1.MACLIB.

```
***********************************************************************
*                                                                     *
*   OAM SUBTYPE SECTION, FOR SUBTYPE 40                                *
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- ---------------------------------------------------       *
*        40 OSMC COMMAND RECYCLE                                     *
*                                                                     *
***********************************************************************
ST40     DSECT                   SUBTYPE 40
ST40STRD DS     CL10' '          DATE RECYCLE COMMAND STARTED     @L5A
ST40ENDD DS     CL10' '          DATE RECYCLE COMMAND ENDED       @L5A
ST40VOLN DS     BL2'0'           NUMBER OF VOLSERS COMPLETED      @L5A
ST40PCTV DS     BL2'0'           PERCENT VALID USED FOR COMMAND   @L5A
ST40LIM  DS     BL2'0'           LIMIT USED FOR COMMAND           @L5A
ST40SUBL DS     CL1' '           TAPE SUBLEVEL USED FOR COMMAND:
*                                A=ALL,
*                                -=TSL NOT SPECIFIED ON CMD,
*                                1=TAPE SUBLEVEL 1,
*                                2=TAPE SUBLEVEL 2.
*                                @L6A
DS     BL1'0'           RESERVED                         @L6C
ST40END  DS     0C               END OF BASE SECTION              @L5A
***********************************************************************
* SUBTYPE 40 VOLUME ARRAY SECTION                                     *
***********************************************************************
ST40VOLD DSECT                   ARRAY OF VOLS COMPLETED RECYCLE @L5A
ST40VSN  DS     40CL6' '         VOLSER                          @L5A
```

*Figure B-49*  SMF record type 85 subtype 40 significant fields (extract from CBRSMF macro)*
Appendix C. Code samples DFSMS V1.10

This appendix contains sample code that can be useful when implementing DFSMS V1.10.

Each sample program is documented separately so that they can be used individually as required.
C.1 SMF Record type 85 subtype 1-10 data display program

Program SMG85TP displays the contents of selected fields of SMF record Type 85 subtypes 1/2/3/4/5/6/7/8/9/10 data. It is not intended to provide a comprehensive report on OAM activity but rather to verify that immediate backup is occurring.

There are three steps to build the program which needs to be done once, after which it can be executed several times. It is not necessary to have in depth assembler experience, but familiarity with JCL is required.

Step 1: Create a PDS/PDSE to hold the members
In this example the PDS is called MHLRES1.SMF85TP.SOURCE.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

Step 2: Store the program source in the PDS
Make sure that your ISPF EDIT menu settings has the screen format defined as DATA.

Cut and paste the contents of Figure B-1 on page 443 and all subsequent ones through to Figure B-7 on page 449 one after the other into member SMFT85AA. The result should contain 333 lines.

C.2 VTOC management: Enhancement for EAV

The Volume Table of Contents (VTOC) plays a pivotal role in the storage and retrieval of data on DASD volumes. The VTOC management macros have been updated with z/OS V1R10 to provide for use of EAV volumes and also to prevent problems that might arise from the use of versions of the macros from prior to z/OS V1R10.

Information about data sets on a DASD volume is contained in records on the VTOC and they are mapped, in the system storage, by the Data Set Control Block (DSCB). Different DSCBs are used depending on what type of data set is being described.

The VTOC is manipulated by the Direct Access Device Storage Management (DADSM) and Common Vtoc Access Facility (CVAF) macros

For the purposes of illustrating the changes required in invocation of the macros, discussion in the section is centered around the basic DSCB that represents a data set, which is the Format-1 DSCB that can have a Format-3 DSCB to describe additional extents of the data set. To accommodate an EAV format volume cylinder addressable area, the equivalent of the Format-1 DSCB is the Format-8 DSCB, which is always paired with a Format-9 DSCB. The Format-8/Format-9 DSCB pair provides for additional information about a data set to be recorded, including provision to retrieve additional extent information more rapidly. Format-3 DSCBs can be used with Format-8/Format-9 DSCBs.

There are activities that have to be updated to access EAV related resources, such as OPEN and read the VTOC of a volume and OPEN an EAS data set using a DCB (this is normally done via an ACB and related services).

The updated macros, supported by the underlying service routines are described next.
DADSM macros:
- LSPACE - provides information about volume size, free space on the volume, free space on the VTOC and INDEX, volume fragmentation, and VTOC status. Also provided is information about the size of the track-managed space and its free space statistics. See C.3, “VTOC management macros extended for EAV - LSPACE” on page 497.
- OBTAIN: Reads one or more DSCBs from the VTOC. See C.4, “VTOC management macros extended for EAV: OBTAIN” on page 510.
- PARTREL: Releases unused space from a sequential or partitioned data set or a PDSE - not applicable to EAV volumes at this stage.

CVAF macros:
- CVAFDIR - directly accesses one or more DSCBs. See C.6, “VTOC management macros extended for EAV: CVAFDIR” on page 530.
- CVAFDSM - volume free space information. See C.7, “VTOC management macros extended for EAV: CVAFDSM” on page 560
- CVAFFILT - reads sets of DSCBs for one or more DASD data sets. See C.8, “VTOC management macros extended for EAV: CVAFFILT” on page 572
- CVAFSEQ - retrieves the following items. See C.9, “VTOC management macros extended for EAV - CVAFSEQ” on page 599:
  • Data set names from an active VTOC index
  • DSCBs in physical-sequential order
  • DSCBs in data set name order (index required)
- CVAFTST - determines if a DASD volume has an active VTOC index - not affected by EAV support

OPEN with DCB related activities:

C.3 VTOC management macros extended for EAV - LSPACE

The LSPACE macro returns information about a volume, including free space in the EAV and non-EAV areas.

The information returned for an EAV volume is different from that for a non-EAV volume, so the LSPACE macro invocation has to be adjusted to support detail about EAV volumes. However, LSPACE coded for z/OS releases prior to V1R10 will work on V1R10 without risk of overlaying storage.

The LSPACE macro requires a Parameter List, designated with z/OS 1.10 as PLISTVER=1 or PLISTVER=2. PLISTVER=1 represents the basic parameter list for z/OS prior to release V1R10. When PLISTVER=2 is specified, related larger areas to return data to are required.

LSPACE from z/OS releases prior to V1R10 functions as if PLISTVER=1 had been specified.
C.3.1 EAV Migration Assistance Tracker considerations: LSPACE

The EAV Migration Assistance Tracker will detect use of the LSPACE service that has not been updated for EAV compatibility.

Any appearance on the tracker list should be investigated and resolved as use of LSPACE against an EAV volume in due course will not report on the Track Managed area. The program running in this way will not, however, fail.

For information about the EAV Migration Tracker, refer to Appendix C of DFSMS Advanced Services, SC26-7400, and in this document to “Migration Assistance Tracker” on page 641.

In Figure C-1 we show information taken from the tracker entries that relate to program LSPACE which has issued the LSPACE macro. The cause such an entry and an example of resolution is discussed in this section.

<table>
<thead>
<tr>
<th>TRACKING INFORMATION</th>
<th>VALUE</th>
<th>JOBNAME</th>
<th>PROGNAME</th>
<th>OFF</th>
<th>ASID</th>
<th>NUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS-I:3 LSPACE DATA=</td>
<td>2003</td>
<td>MHLRES1L</td>
<td>LSPACEO</td>
<td>AC 42</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

*Figure C-1  Example of tracker entry for LSPACE*

C.3.2 LSPACE macro use examples

The LSPACE macro can be coded to return data in various ways. In order to illustrate the type of changes required to support EAV volumes, a very simple scenario is illustrated:

- Program to issue LSPACE against a volume in pre-z/OS V1R10 form
- Program to issue LSPACE against a volume in z/OS V1R10 form

C.3.2.1 Program LSPACEO: Use of LSPACE not enhanced

Program LSPACEO issues the LSPACE macro against a volume that is defined through the JCL LSPACEDD DD statement. The data set on the DD statement is insignificant and it is not opened.

In Figure C-2 on page 499 through to Figure C-4 on page 501 we show source code to issue the LSPACE macro.

The LSPACE DATA= option is used to return the available data to the area named MYDATA.

The information is extracted and formatted for printing.
Figure C-2  Source of LSPACEO program (1 of 3)
*        PUT   PRINTDCB,PRINTL4                                         00570038
*        DC   H'0'        CAUSE ABEND                                  00580038
RETURN   DS   OH                                                       00590038
SEGEND                                                         00600038
WORKAREA DS  OD   LOCATE WORK AREA (265 BYTES)               00610038
EYEEYE   DC    CL8'TIOTTIOT'                                            00620038
ATIOT    DS   F             ADDRESS OF THE TIOT                        00630038
UCBAD    DS   F             ADDRESS OF THE UCB WE WANT                 00640038
MYDATA   DS   OCL36                                                    00650038
LSPDRETN DS   C                                                        00660038
LSPDSTAT DS   C                                                        00670038
LSPRSV1 DS   CL2                                                      00680038
LSPDNEXT DS   CL4                                                      00690038
LSPDCYL DS   CL4                                                      00700038
LSPDTRK DS   CL4                                                      00710038
LSPDLCYL DS   CL4                                                      00720038
LSPDLTRK DS   CL4                                                      00730038
LSPDFOS DS   CL4                                                      00740038
LSPDVIRS DS   CL4                                                      00750038
LSPDFRAG DS   CL4                                                      00760038
*                                                                       00770038
LSPACEED DC   CL8'LSPACEEDD'                                            00780038
*                                                                       00790038
DWORD    DS   D                                                        00800038
PWORD    DC    CL11' '                                                  00810038
*                                                                       00820038
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133               00830038
*                                                                       00840038
PRINTL0 DC    CL133' '                                                  00850038
ORG   PRINTL0+1                                                  00860038
DC    CL30' '                                                  00870038
FSPCEH DC    C' FREE SPACE ON VOLUME'                                 00880038
DC    C' '                                                     00890038
VOLSER   DC    CL6' '                                                   00900038
ORG                                                            00910038
PRINTL1 DC    CL133' '                                                  00920038
ORG   PRINTL1+1                                                  00930038
TOTALH   DC    C'TOTAL'                                                 00940038
ORG   PRINTL1+22                                                  00950038
TRACKSH DC    C'ADD TRACKS'                                            00960038
ORG   PRINTL1+51                                                  00970038
CYLSH    DC    C'CYLS'                                                 00980038
ORG   PRINTL1+68                                                  00990038
FREEXTC DC    C'FREE XTS'                                              01000038
ORG                                                            01010038
*                                                                       01020038
PRINTL2 DC    CL133' '                                                  01030038
ORG   PRINTL2+1                                                  01040038
SIZECC DC    CL20'SIZE ..:'                                            01050038
DC    C' '                                                   01060038
SIZET    DC    CL22' '                                                 01070038
DC    C' '                                                   01080038
SIZEC    DC    CL20' '                                                 01090038
DC    C' '                                                   01100038

Figure C-3   Source of LSPACEO program (2 of 3)
In Figure C-5 we show the JCL to execute program LSPACEO. The volume whose SPACE is requested is defined through the LSPACEDD DD statement. The data set named is not significant.

```
//MHLRES1L JOB (1234567,COMMENT),UAALF0,TIME=10, 
// MSGLEVEL=1,CLASS=A, 
// NOTIFY=&SYSUID 
/*JOBPARM S=*
//LSPACEO PROC 
//ASM      EXEC PGM=LSPACEO,REGION=0M 
//STEPLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD 
//LSPACEDD DD DISP=SHR,DSN='MHLRES1.EAV.ASM' 
//SYSUDUMP DD SYSOUT=* 
//PRINT    DD SYSOUT=* 
// PEND 
// EXEC LSPACEO 
```

In Figure C-6 we show the output from running program LSPACEO. This shows that the are 1375 CYLS free on the volume, of which the largest is 1314 CYLS. There is also 99 additional tracks on the volume. There are 18 free extents in the VTOC.

```
FREE SPACE ON VOLUME MLD20C
TOTAL ADD TRACKS CYLS FREE XTS
SIZE . .: 0000000999 000001375 000000018
LARGEST.: 0000000000 0000001314
```

In Figure C-7 we show similar output from ISPF option 3.4 In this case the free space available as expressed in Cyls has been converted to Tracks, and the additional free tracks have been added to the total.
C.3.2.2 Program LSPACEE: Use of LSPACE enhanced

Program LSPACEE issues the LPACE macro against a volume that is defined through the JCL LSPACEE DD statement. The data set on the DD statement is insignificant and it is not opened.

In Figure C-8 on page 503 through to Figure C-11 on page 506 we show source code to issue the LSPACE macro using two of the new options provide for EAV support.

The LSPACE EXPDATA= option is used to return the available data to the area named MYDATAE which has been enlarged appropriately. The use of the EXPDATA keyword requires the use of the PLISTVER=2 setting which indicates to the LSPACE service routine that it is safe to return additional data. It is of course necessary for the programmer to have provided the enlarged area.

The information is extracted and formatted for printing.

Here is a summary of the changes made to the LSPACEO source code to create LSPACEE.

- LSPACE macro - changed DATA=MYDATA to EXPDATA=MYDATAE and added PLISTVER=2 (There was no need to change the name of the data area from MYDATA to MYDATAE, it was just done to emphasize that the change in length is required)
- Added test for volume having cylinder managed area or not
- Added code to interpret the additional data returned
- Added additional fields to what was MYDATA
- Added additional print functions for the additional data

Of these changes, the critical change is the addition of the additional fields in MYDATAE to increase the length and so avoid the storage overlay that would occur if no change was made in this area when the DATA= statement is changed to EXPDATA=.

The LSPACE macro has the option to return data in a Message area. Prior to z/OS V1R10 there was a MSG= and a EXPMSG option which required a larger return area. The additional data returned in support of the EAV environment required a further increase leading to provision of the XEXPMSG option. Conversion between MSG, EXPMSG and XEXPMSG requires attention to the size of the data area similar changes to those made for the DATA to EXPDATA expansion.
Appendix C. Code samples DFSMS V1.10

503

Figure C-8   Source of LSPACEE program (1 of 4)

LSPACEE SEGSTART
* ISSUE LSPACE AGAINST THE VOLUME POINTED TO BY A DDNAME
* AND COMPOSE A REPORT
* ALLOW FOR EAV VOLUME FORMAT
*
* OPEN (PRINTDCB,(OUTPUT))
EXTRACT ATIOT,FIELDS=(TIOT)
L R1,ATIOT TCBTIOT - POINT TO TIOT
LA R15,24(,R1) TIOENTRY
USING TIOENTRY,R15

TIOTLOOP CLI TIOENTRY,'X'00' END OF TIOT?
BE ABEND YES - IMPOSSIBLE
CLC TIOEDDNM,LSPACEDD DOES DDNAME MATCH?
BE TIOTEXIT YES - RETURN
SR R1,R1
IC R1,TIOELNGH GET LENGTH OF ENTRY
LA R15,0(R1,R15) POINT TO NEXT ENTRY
B TIOTLOOP

ABEND DC H'0'

TIOTEXIT L R1,TIOFSRT-1 GET UCB ADDRESS
LA R1,0(,R1) CLEAR HI ORDER BYTE
ST R1,UCBAD SAVE UCB ADDRESS
MVC VOLSER,28(R1)
*
LSPACE UCB=UCBAD,EXPDATA=MYDATAE,PLISTVER=2
TM LSPDRETN,LSPDCYLM
BO EAVYES
MVC EAVNO(2),='C'NO'
EAVYES EQU *
PUT PRINTDCB,PRINTL0
PUT PRINTDCB,PRINTL1V
L R1,LSPDTRK
CVD R1,DWORD
OI DWORD+7,X'0F'
UNPK PWORD(11),DWORD+2(6)
MVC SIZEZ,PWORD+1
L R1,LSPDTCYL
CVD R1,DWORD
OI DWORD+7,X'0F'
UNPK PWORD(11),DWORD+2(6)
MVC SIZEC,PWORD+1
L R1,LSPDNEXT
CVD R1,DWORD
OI DWORD+7,X'0F'
UNPK PWORD(11),DWORD+2(6)
MVC LARGET,PWORD+1
*
PUT PRINTDCB,PRINTL2
L R1,LSPDLTRK
CVD R1,DWORD
OI DWORD+7,X'0F'
UNPK PWORD(11),DWORD+2(6)
MVC LARGET,PWORD+1
L R1,LSPDLCYL
CVD R1,DWORD

Figure C-8   Source of LSPACEE program (1 of 4)
Figure C-9  Source of LSPACEE program (2 of 4)
* TRACK MANAGED AREA

LSPDVNXT DS CL4 01110001
LSPDVCL DS CL4 01120001
LSPDVTTK DS CL4 01130001
LSPREVCL DS CL4 01140007
LSPDVLT KS CL4 01150001
LSPDVMR DS CL4 01160001
LSPDVFRG DS CL4 01170001

* VOLUME AREA

LSPDTRKS DS CL4 01180003
LSPDCYLS DS CL4 01200003
LSPDRSV8 DS CL60 01210003


LSPACEED DC CL8'LSPACEED' 01220000

DWORD DS D 01230000
PWORD DC CL11' ' 01240000

PRINTDCB DCB DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133 01250000
PRINTL0 DC CL133' ' 01260000
ORG PRINTL0+1 01270000
DC CL30' ' 01280000

FSPCEH DC C' FREE SPACE ON VOLUME' 01290000
DC C' ' 01300000
VOLSER DC CL6' ' 01310000
DC C' ' 01320000

EAV DC C' WHICH HAS A CYLINDER MANAGED AREA' 01330000
EAVNO EQU EAV+11 01340000

TOTALHV DC C'TOTAL' 01350000
ORG PRINTL1V+22 01360000
TRACSHV DC C'ADD TRACKS' 01370000
ORG PRINTL1V+51 01380000
CYLSHV DC C'CYLS' 01390000
ORG PRINTL1V+68 01400000
FREEXTCV DC C'FREE XTS' 01410000

TOTALHT DC C'TRACK MANAGED' 01420000
ORG PRINTL1T+22 01430000
TRACSHV DC C'ADD TRACKS' 01440000
ORG PRINTL1T+51 01450000
CYLSHT DC C'CYLS' 01460000
ORG PRINTL1T+68 01470000
FREEXTCT DC C'FREE XTS' 01480000

* 01490000
As a result of the use of the EXPDATA keyword, LSPACE returns additional data that allows the part of an EAV volume that is track managed to be seen separately. If the volume is not EAV capable, or does not have a Cylinder managed area, there is no difference between the Total space and the Track managed space.

In Figure C-12 we show the JCL to execute program LSPACEE against data set MHLRES1.EAV.ASM then against MHLEAV.EDSBIG02.

```
PRINTL2 DC CL133 ' ' 01610000
           ORG PRINTL2+1 01620000
SIZECC DC CL20 ' SIZE . .:' 01630000
           DC ' ' 01640000
SIZET DC CL22 ' 01650000
           DC ' ' 01660000
SIZEC DC CL20 ' 01670000
           DC ' ' 01680000
FREEXN DC CL22 ' 01690000
           ORG 01700000
PRINTL3 DC CL133 ' ' 01710000
           ORG PRINTL3+1 01720000
LARGECC DC CL20 ' LARGEST . .:' 01730000
           DC ' ' 01740000
LARGET DC CL22 ' 01750000
           DC ' ' 01760000
LARGECE DC CL20 ' 01770000
           ORG 01780000
TIOT DSECT 01790000
           DS CL24 JOBNAME, ETC 01800000
TIOENTRY DS OC 01810000
TIOELNGH DS AL1 LENGTH OF THIS ENTRY 01820000
           DS XL3 01830000
TIOEDDNM DS CL8 DD NAME 01840000
TIOEDFCB DS CL3 TTR OF JFCB 01850000
           DS XL2 01860000
TIOEFURT DS AL3 ADDRESS OF UCB 01870000
           END 01880000
//MHLRES1L JOB (1234567,COMMENT),UAALF0,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*JOBPARM S=* 
//LSPACEE PROC
//ASM EXEC PGM=LSPACEE,REGION=0M 
//STEPLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//LSPACEEDD DD DISP=SHR,DSN='MHLRES1.EAV.ASM'
//SYSUDUMP DD SYSOUT=* 
//PRINT DD SYSOUT=* 
//PEND
// EXEC LSPACEE
// EXEC LSPACEE
//LSPACEEDD DD DISP=SHR,DSN='MHLEAV.EDSBIG02'
```

**Figure C-11**  Source of LSPACEE program (4 of 4)

**Figure C-12**  JCL to execute program LSPACEE against the volumes two data sets are on
In Figure C-13 we show the output from LSPACEE running against volume MLD20C. Volume MLD20C does not have a cylinder managed area (is not an EAV volume) so the output shows the same figures for the TOTAL and for the TRACK MANAGED areas.

| FREE SPACE ON VOLUME MLD20C WHICH HAS NO CYLINDER MANAGED AREA |
|----------------|----------------|------------|-------------|
| TOTAL ADD TRACKS CYLS FREE XTS |
| SIZE .: 0000000099 000001375 000000018 |
| LARGEST.: 0000000000 0000001314 |

**Figure C-13** Output from execution of program LSPACEE against volume MLD20C

The ISPF 3.4 output is the same as illustrated with the LSPACEO program, repeated here for convenience in Figure C-14.

| Volume .: MLD20C |
| Command ===> |
| Unit .: 3390 |

<table>
<thead>
<tr>
<th>Volume Data</th>
<th>VTOC Data</th>
<th>Free Space</th>
<th>Tracks</th>
<th>Cyls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tracks .: 150,255</td>
<td>Tracks .: 180</td>
<td>Size .: 20,724</td>
<td>1,375</td>
<td></td>
</tr>
<tr>
<td>%Used .: 86</td>
<td>%Used .: 1</td>
<td>Largest .: 19,710</td>
<td>1,314</td>
<td></td>
</tr>
<tr>
<td>Trks/Cyls: 15</td>
<td>Free DSCBS: 8,939</td>
<td>Free Extents .: 18</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure C-14** Output from ISPF 3.4 showing free space information about volume MLD20C

In Figure C-15 we show the output from LSPACEE running against volume MLDE65. Volume MLDE65 does have a cylinder managed area (is an EAV volume) so the output shows different figures for the TOTAL and for the TRACK MANAGED areas.

| FREE SPACE ON VOLUME MLDE65 WHICH HAS A CYLINDER MANAGED AREA |
|----------------|----------------|-------------|-------------|
| TOTAL ADD TRACKS CYLS FREE XTS |
| SIZE .: 0000000042 0000017614 0000000010 |
| LARGEST.: 0000000005 0000009269 |

**Figure C-15** Output from execution of program LSPACEE against volume MLDE65

In Figure C-16 we show similar output from ISPF option 3.4. In this case the free space available as expressed in Cyls has been converted to Tracks, and the additional free tracks have been added to the total.

In addition, ISPF has recognized that the volume is set up for EAV, and the Track Managed area is separately formatted.
C.3.2.3 Execution of sample LSPACE programs

The LSPACEO and LSPACEE programs are simple working examples. If you want to run a program, you need to assemble link it. It is not necessary to have in depth assembler experience, but familiarity with JCL is required.

The source and support macros are stored in one PDS, the JCL in one PDS, and the LOAD modules in another.

Required once in support of all the VTOC management programs, set up the macros. Refer to C.12, “Supporting macros for sample programs” on page 630.

To create the assembler PDS, use the following steps.

There are three steps to build the program which needs to be done once, after which it can be executed several times.

**Step 1: Create a PDS/PDSE to hold the source members**

In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

**Step 2: Create a PDS/PDSE to hold the LOAD MODULES**

In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 0/U. All other attributes can be chosen by you.

**Step 3: Store the program source in the PDS**

Make sure that your ISPF EDIT menu settings has the screen format defined as DATA, the emulator line length is set to greater than 80, and that the edit session is in NONUM mode.

- Choose whether to build LSPACEE or LSPACEO. This example assumes that LSPACEE is being built.
- Cut and paste the contents of Figure C-8 on page 503 through to Figure C-11 on page 506 one after the other into member LSPACEEE. The result should contain 188 lines.

---

<table>
<thead>
<tr>
<th>Volume . : MLDE65</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command ==&gt;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Unit . : 3390</td>
</tr>
<tr>
<td>Free Space</td>
</tr>
<tr>
<td>VTOC Data</td>
</tr>
<tr>
<td>Tracks . : 750</td>
</tr>
<tr>
<td>Size . : 1,074,252</td>
</tr>
<tr>
<td>%Used . : 1</td>
</tr>
<tr>
<td>Largest . : 139,040</td>
</tr>
<tr>
<td>Free DSCBS: 37,491</td>
</tr>
<tr>
<td>Extents . : 10</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Volume Data</td>
</tr>
<tr>
<td>Tracks . : 1,135,260</td>
</tr>
<tr>
<td>Size . : 981,957</td>
</tr>
<tr>
<td>%Used . : 5</td>
</tr>
<tr>
<td>Largest . : 139,040</td>
</tr>
<tr>
<td>Trks/Cyls: 15</td>
</tr>
<tr>
<td>Free Extents . : 9</td>
</tr>
</tbody>
</table>

Figure C-16  Output from ISPF 3.4 showing free space information about volume MLDE65
In order to assemble LSPACEE the supporting macros are needed. Listing of the macros are provided. They need to be stored as members in MHLRES1.EAV.ASM. The same macros are used for several programs and subroutines. Refer to C.12, “Supporting macros for sample programs” on page 630 for information about storing the macros if this has not already been done.

**Step 4: Create a PDS/PDSE to hold the JCL members**

In this example the PDS is called MHLRES1.EAV.JCL.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

**Step 5: Store the JCL source in the PDS**

- Cut and paste the contents of Figure C-17 into member LSPACEE. The result should contain 29 lines.

```
//MHLRES1L JOB (1234567,COMMENT),UAALF0,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*/JOBPARM S=*
//ASMHCL PROC
//ASM EXEC PGM=ASMA90,REGION=0M,
//     PARM='OBJECT,NODECK'
//SYSLIN DD DSN=&OBJ,DISP=(NEW,PASS),UNIT=SYSDA,
//     SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120
//SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB
// DD DISP=SHR,DSN=SYS1.MODGEN
// DD DISP=SHR,DSN=MHLRES1.EAV.ASM
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.ASM
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
//*
//LKED EXEC PGM=HEWL,REGION=2048K,COND=(8,LE,ASM),
// PARM='XREF,LIST,LET'
//SYSLIN DD DSN=&OBJ,DISP=(OLD,DELETE)
// DD DDNAME=SYSIN
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=SYSIN,UNIT=SYSDA,SPACE=(CYL,(5,5))
// PEND
// EXEC ASMHCL
//ASM.SYSIN DD DISP=SHR,DSN=MHLRES1.EAV.ASM(LSPACEE)
/*/LKEO.SYSLMOD DD DSN=MHLRES1.EAV.LOAD,DISP=SHR
//LKEO.SYSIN DD *
NAME LSPACEE(R)
```

*Figure C-17  JCL to assemble and link LSPACEE*
Step 7: Run the job in member LSPACE
Check that the return code from the ASM and LKED steps is 0.

A return code of 8 on the ASM step is an indication that the macros required have not been stored in MHLRES1.EAV.ASM.

The LSPACE program is now ready for execution.

An exactly similar process can be followed to set up program LSPACEO

C.4 VTOC management macros extended for EAV: OBTAIN

The OBTAIN macro, in conjunction with the CAMLST macro, is used to locate information about a data set.

The OBTAIN macro as implemented in z/OS releases prior to z/OS V1R10, and if not extended by EAV keywords is not permitted to retrieve data set information from EAV formatted volumes. Use of an un-extended OBTAIN macro call against a data set on an EAV volume will result in return code 24 (dec) and the data requested will not be returned.

In order to provide insight into what programs in an installation are using OBTAIN macros, a tracking facility has been provided that will capture such instances but not interrupt processing.

Attention: The OBTAIN work area is defined to be 140 bytes long. Of that 140 bytes, in releases prior to z/OS V1.10, only 101 bytes of that area were used, the remainder being reserved. There have been reports of some programs defining just 101 bytes for the workarea. Such programs can abend when run on z/OS 1.10 because an additional 2 bytes are now returned that might overwrite the following data areas.

All programs that use OBTAIN should be checked to ensure that the OBTAIN work area is defined with the full 140 bytes. This is also significant when it comes to expanding the OBTAIN function for EAV support.

C.4.1 EAV Migration Assistance Tracker considerations: OBTAIN

The EAV Migration Assistance Tracker will detect use of the OBTAIN service that has not been updated for EAV compatibility.

Any appearance on the tracker list must be investigated and resolved because use of an un-extended OBTAIN against an EAV volume in due course will fail.

For information about the EAV Migration Tracker, refer to Appendix C of DFSMS Advanced Services, SC26-7400 and in this document to “Migration Assistance Tracker” on page 641.

In Figure C-18 we show information taken from the tracker entries that relate to program OBTDROV0 which has issued the OBTAIN macro. The cause of such an entry and an example of resolution is discussed in this section

----TRACKING INFORMATION---- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM
SMS-E:1 DADSM OBTAIN C10001 MHLRES1C OBTDROV0 3FC 42 1

Figure C-18 Example of tracker entry for OBTAIN
C.4.2 OBTAIN macro use examples

We have provided simple program examples to illustrate the various scenarios:

- Unmodified OBTAIN issued for data set on non-EAV volume
- Unmodified OBTAIN issued for data set on EAV volume
- Updated OBTAIN issued for data set on non-EAV volume
- Updated OBATIN issued for data set on EAV volume

We provide examples of the minimal changes needed to support use of EAV volumes.

The examples consist of a subroutine to issue the OBTAIN macro, in unmodified and modified format data driver program to call the subroutine.

C.4.2.1 OBTMODO subroutine: Non-EAV

Subroutine OBTMODO retrieves the contents of selected fields of the DSCB for a data set. It is not intended to provide a comprehensive report on DSCB contents but rather to provide the basis to illustrate conversion to EAV capability. OBTMODO is written in reentrant code because that is the usual way such routines would be written.

In Figure C-19 on page 512 to Figure C-21 on page 514 we show the source listing for the subroutine to issue the OBTAIN macro.
Figure C-19  OBTMODO subroutine source listing (1 of 3)
NONVSAM  LA    R5,DSORGS  R5 -> PS PO DA IS ** 00540000
TM    DS1DSORG,DS1DSGPS IS DSORG PS ? 00550000
BO    STODSRG 00560000
LA    R5,4(R5) 00570000
TM    DS1DSORG,DS1DSGPO IS DSORG PO ? 00580000
BO    STODSRG 00590000
LA    R5,4(R5) 00600000
TM    DS1DSORG,DS1DSGDA IS DSORG DA ? 00610000
BO    STODSRG 00620000
LA    R5,4(R5) 00630000
TM    DS1DSORG,DS1DSGIS IS DSORG IS ? 00640000
BO    STODSRG 00650000
LA    R5,4(R5) 00660000
STODSRG MVC 0(4,R4),0(R5) ATTRIB(1) = DSORG 00670000
* 00680000
IC    R5,DS1RECFM 00690000
SRL   R5,4 00700000
N     R5,=F'12'  R5 =  0 4 8 12 00710000
LA    R5,RECFMS(R5) 00720000
MVC   4(4,R4),0(R5) ATTRIB(2) = RECFM 00730000
MVC   8(4,R4),0(R5) ATTRIB(3) = RECFM 00740000
* 00750000
LA    R5,5(R4) 00760000
TM    DS1RECFM,DS1RECFB BLOCKED ? 00770000
BNO   REC2 00780000
MVI   0(R5),C'B' 00790000
MVI   0(R5),C'B' 00800000
LA    R5,1(R5) 00810000
* 00820000
REC2  TM    DS1RECFM,DS1RECFS SPANNED ? 00830000
BNO   REC3 00840000
MVI   0(R5),C'S' 00850000
MVI   10(R4),C'S' 00860000
LA    R5,1(R5) 00870000
* 00880000
REC3  TM    DS1RECFM,DS1RECFA ANSI CONTROL CHAR ? 00890000
BNO   REC4 00900000
MVI   0(R5),C'A' 00910000
MVI   11(R4),C'A' 00920000
B    LRECL 00930000
* 00940000
REC4  TM    DS1RECFM,DS1RECMC MACHINE CONTROL CHAR ? 00950000
BNO   LRECL 00960000
MVI   0(R5),C'M' 00970000
MVI   11(R4),C'M' 00980000
* 00990000
LRECL  XC    12(8,R4),12(R4) ZERO LRECL AND BLKSIZE 01000000
MVC   14(2,R4),DS1LRECL MOVE LRECL TO CALLERS VECTOR 01010000
MVC   18(2,R4),DS1BLKL MOVE BLKSIZE TO CALLERS VECTOR 01020000
* 01030000

Figure C-20  OBTMODO subroutine source listing (2 of 3)
Subroutine operation

ZCSSTART and ZCSEND are macros to set up the routine and establish the reentrant environment, and can be ignored as far as the logic is concerned.

Each time the OBTAIN macro is issued, it uses control blocks generated by the CAMLST macro (which is also used by other macros). LOCATE is issued once to locate the data set from the system catalog, then OBTAIN is issued to retrieve the information from the DSCB.

There is only one DSCB work area defined in this version of the routine.

Data is passed back to the calling program for interpretation and display.

In Figure C-22 on page 515 and Figure C-23 on page 516 we show the source of the driver and display program for subroutine OBTMODO.
Figure C-22  OBTDRVO program source to call OBTMODO routine (1 of 2)
MVDSN MVC DSN(0),O(R4) CODE TO EXECUTE 00550000
  DS  OF 00560000
DSNR DS CL44 00570000
RCR DS CL4 00580000
  DC C'ATTRATTR' 00590000
ATTRR DS CL40 00600000
  ORG ATTRR+12 00610000
LRECLR DS F 00620000
BLKR DS F 00630000
FMTR DS CL2 00640000
DSCBS DS H 00650002
  ORG 00660000
DWORD DS D 00670000
PWORD DC CL11' ' 00680000
  * 00690000
PRINTDCB DCB DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133 00700000
  * 00710000
PRINTL0 DC CL133' ' 00720000
  ORG PRINTL0+1 00730000
DSNAMEH DC CL44'DSNAMC' 00740000
  DC C' ' 00750000
  RCODEH DC CL4RC' 00760000
  DC C' ' 00770000
  ATTRBH DC CL40ATTRBS' 00780000
  ORG ATTRBH+12 00790000
  DC CL10'LAECBL' 00800000
  DC C' ' 00810000
  DC CL10'BLKSIZ' 00820000
DSCBFMTH DC CL2'DS' 00830000
  DC C' ' 00840000
  DSCBNUMH DC CL10'DSCBS' 00850000
  ORG 00860000
  * 00870000
PRINTL1 DC CL133' ' 00880000
  ORG PRINTL1+1 00890000
DSNAME DC CL44' ' 00900000
  DC C' ' 00910000
  RCODE DC CL4' ' 00920000
  DC C' ' 00930000
  ATTRIB DC CL40' ' 00940000
  ORG ATTRIB+11 00950000
  DC C' ' 00960000
  LRECL DC CL10' ' 00970000
  DC C' ' 00980000
  BLKSIZ DC CL10' ' 00990000
DSCBFMT DC CL2'XX' 01000000
  DC C' ' 01010002
  DSCBNUM DC CL10' ' 01020002
  ORG 01030000
END 01040000

Figure C-23 OBTDRVO program source to call OBTMODO routine (2 of 2)
SEGSTART and SEGEND are macros to set up the routine and establish the environment, and can be ignored as far as the logic is concerned.

The program is intended to display the following information:

1. Data set name
2. Return code from OBTAIN
   a. 0 (x'00') - OK
   b. 4 (x'04') - required volume not mounted
   c. 8 (x'08') - DSCB not found on the volume specified
   d. 12 (x'0C') - Permanent error
   e. 16 (x'10') - Invalid workarea pointer
   f. 24 (x'18') - Data set has Format-8 DSCB and EADSCB=NOTOK
   g. 28 (x'1C') - Internal error and EADSCB=NOTOK
3. Data set attributes - Organization, RECFM, RECFM-X, LRECL, BLKSIZE
   a. Organization - PO, PS, VS etc.
   b. RECFM - F/FB/FBA/V/VA/VB/VBA etc. - replaced by **** for VS organization
   c. RECFM-X as for RECFM but the carriage control character, if present, is separated out
   d. LRECL - Logical record length - suppressed for VS origination
   e. BLKSIZE - Data set Blocksize - suppressed for VS organization
4. DSCB Format Identifier
   a. F1/F4/F8 etc.
5. Number of DSCBs associated with the data set
   a. Number of DSCBS associated with the data set

Additional information:

The data set whose attributes are wanted is passed as a PARM to the program.

After initialization, the basic logic is:

1. Call subroutine OBTMODO.
2. OPEN the DCB.
3. Print the Header.
4. Interpret the data and format.
5. Print the detail.
In Figure C-24 we show an example of a job executing the OBTDRVO program several times.

```
//MHLEAV1C JOB (1234567,COMMENT),UAALF0,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
//*JOBPARM S=*  
//CHKDSN PROC
//ASM      EXEC PGM=OBTDRVO,REGION=0M,
//             PARM='MHLRES1.EAV.JCL'
//STEPLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//SYSUDUMP DD SYSOUT=*  
//PRINT DD SYSOUT=*  
//PEND
// EXEC CHKDSN,PARM='MHLRES1.EAV.JCL'
// EXEC CHKDSN,PARM='MHLRES1.DDIR.D'
// EXEC CHKDSN,PARM='MHLRES1.DDIR.DUD'
// EXEC CHKDSN,PARM='MHLEAV.EXTKSDS.DATA'
// EXEC CHKDSN,PARM='MHLEAV.EXTGELA.TEST0005.DATA'
// EXEC CHKDSN,PARM='MHLEAV.EXTGELA.TEST0005.INDEX'
// EXEC CHKDSN,PARM='MHLEAV.PE.D195653.DATA'
// EXEC CHKDSN,PARM='MHLEAV.PE.D195653.INDEX'
```

Figure C-24   Example of JOB to examine several data sets using OBTDRVO

In Figure C-25 we show the results of running the job in Figure C-24.

```
DSNAME                    RC  ATTRIBS  LRECL  BLKSIZ  DS #DSCBS  
MHLEAV1.EAV.JCL          00  PO FB FB  0000000080 0000027920 1 0000000001
DSNAME                    RC  ATTRIBS  LRECL  BLKSIZ  DS #DSCBS  
MHLEAV1.DDIR.D           00  VS *******                      1 0000000001
DSNAME                    RC  ATTRIBS  LRECL  BLKSIZ  DS #DSCBS  
MHLEAV1.DDIR.DUD         12                     12
DSNAME                    RC  ATTRIBS  LRECL  BLKSIZ  DS #DSCBS  
MHLEAV.EXTKSDS.DATA      24                     12
DSNAME                    RC  ATTRIBS  LRECL  BLKSIZ  DS #DSCBS  
MHLEAV.EXTGELA.TEST0005.DATA  12
DSNAME                    RC  ATTRIBS  LRECL  BLKSIZ  DS #DSCBS  
MHLEAV.EXTGELA.TEST0005.INDEX  12
DSNAME                    RC  ATTRIBS  LRECL  BLKSIZ  DS #DSCBS  
MHLEAV.PE.D195653.DATA   24                     24
DSNAME                    RC  ATTRIBS  LRECL  BLKSIZ  DS #DSCBS  
MHLEAV.PE.D195653.INDEX  24                     24
```

Figure C-25   OBTDRVO output from sample

This output shows the following results:

- `MHLRES1.EAV.JCL` is a partitioned data set, and was retrieved successfully (RC=0).
- `MHLRES1.DDIR.D` is the data portion of a VSAM data set, not in an EAV volume, and was retrieved successfully (RC=0).
- `MHLRES1.DDIR.DUD` was not found (RC=12).
- `MHLEAV.EXTKSDS.DATA` is the data portion of a VSAM data set. It was located on an EAV volume (RC=24).
- `MHLEAV.EXTGELA.TEST0005.DATA` was not found (RC=12).
- `MHLEAV.EXTGELA.TEST0005.INDEX` was not found (RC=12).
MHLEAV.PE.D195653.DATA is the data portion of a VSAM data set. It was located on an EAV volume (RC=24).

MHLEAV.PE.D195653.DATA is the data portion of a VSAM data set. It was located on an EAV volume (RC=24).

To contrast the result of running a version of the program that has been enabled to retrieved data sets on EAV volumes, see Figure C-26.

```
Figure C-26  OBTDRVE output from sample

This output shows the following results:

- MHLEAV.PE.D195653.DATA is the data portion of a VSAM data set. It was located on an EAV volume (RC=24).
- MHLEAV.PE.D195653.DATA is the data portion of a VSAM data set. It was located on an EAV volume (RC=24).

```

The enhanced output is provided by the OBTMODE subroutine driven by the OBTDRVE program.
C.4.2.2 OBTMODE subroutine: EAV

Subroutine OBTMODE retrieves the contents of selected fields of the DSCB for a data set. It is not intended to provide a comprehensive report on DSCB contents but rather to provide the basis to illustrate conversion to EAV capability. OBTMODE is written in reentrant code because that is the usual way such routines would be written.

In Figure C-27 on page 521 to Figure C-29 on page 523, we show the source listing for the subroutine to issue the OBTAIN macro.

There are only two differences between OBTMODO and OBTMODE, but they are significant for the support of EAV volumes:

- The line:
  
  DSCB9 DS CL140 OBTAIN WORK AREA (DSCB9)
  
  Is added after the line with label DSCB1

- The line:
  
  OBTAIN CAMOBT
  
  Is changed to:

  OBTAIN CAMOBT,NUMBERDSCB=2,EADSCB=OK OBTAIN 2XDSCBS FROM VTOC

The NUMBERDSCB=2 indicates that two work areas have been provided. This relates to the original DSCB work area (DSCB1) with the addition of DSCB9. As a consequence of providing the additional work area, it is then safe to specify EADSCB=OK. Even though the work area is still labelled DSCB1, the Format-8 DSCB data is returned to the same area.

If a data set with only a format-1 DSCB is returned, then the DSCB9 area is unused.

Program OBTDRVE with its subroutine OBTMODE can be used on EAV and non-EAV DASD volumes.
Figure C-27  OBTMODE subroutine source listing (1 of 3)
NONVSAM LA R5,DSORG2 R5 -> PS PO DA IS **
TM DSIDSORG,DSIDSGPS IS DSORG PS ?
BO STODSRG
LA R5,4(R5)
TM DSIDSORG,DSIDSPO IS DSORG PO ?
BO STODSRG
LA R5,4(R5)
TM DSIDSORG,DSIDSDA IS DSORG DA ?
BO STODSRG
LA R5,4(R5)
TM DSIDSORG,DSIDSIS IS DSORG IS ?
BO STODSRG
LA R5,4(R5)
STODSRG MVC 0(R4),0(R5) ATTRIB(1) = DSORG
* IC R5,DS1RECFM
SRL R5,4
N R5,=F'12' R5 = 0 4 8 12
LA R5,RECFMS(R5) R5 -> * V F U
MVC 4(R4),0(R5) ATTRIB(2) = RECFM
MVC 8(R4),0(R5) ATTRIB(3) = RECFM
* LA R5,5(R4)
TM DS1RECFM,DS1RECFB BLOCKED ?
BNO REC2
MVI 0(R5),C'B'
MVI 9(R4),C'B'
LA R5,1(R5)
* REC2 TM DS1RECFM,DS1RECFS SPANNED ?
BNO REC3
MVI 0(R5),C'S'
MVI 10(R4),C'S'
LA R5,1(R5)
* REC3 TM DS1RECFM,DS1RECFA ANSI CONTROL CHAR ?
BNO REC4
MVI 0(R5),C'A'
MVI 11(R4),C'A'
B LRECL
* REC4 TM DS1RECFM,DS1RECMC MACHINE CONTROL CHAR ?
BNO LRECL
MVI 0(R5),C'M'
MVI 11(R4),C'M'
* LRECL XC 12(B,R4),12(R4) ZERO LRECL AND BLKSIZE
MVC 14(B,R4),DS1LRECL MOVE LRECL TO CALLERS VECTOR
MVC 18(B,R4),DS1BLKL MOVE BLKSIZE TO CALLERS VECTOR
* Figure C-28 OBTMODE subroutine source listing (2 of 3)
Subroutine operation

ZCSSTART and ZCSEND are macros to set up the routine and establish the reentrant environment, and can be ignored as far as the logic is concerned.

Each time the OBTAIN macro is issued, it uses control blocks generated by the CAMLST macro (which is also used by other macros). LOCATE is issued once to locate the data set from the system catalog, then OBTAIN is issued to retrieve the information from the DSCB.

There are two DSCB work areas defined in this version of the routine, DSCB1 and DSCB9.

Attention: It is essential that the defined DSCB work areas be 140 bytes long each because each one is assumed to start 140 bytes from the start of the previous one. Only the location of the first area is passed to the OBTAIN service routine.

Data is passed back to the calling program for interpretation and display.

In Figure C-30 on page 524 and Figure C-31 on page 525, we show the source of the driver and display program for subroutine OBTMODE.

The only difference between the OBTDREVE and OBTDREVE programs is the name, and the subroutine they call.
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00010001</td>
<td>OBTDRVE SEGSTART</td>
</tr>
<tr>
<td>00020000</td>
<td>* DRIVER PROGRAM TO CALL OBTMODE (DSNR, RCR, ATTRR)</td>
</tr>
<tr>
<td>00030001</td>
<td>* DSNR 44 INPUT DSN</td>
</tr>
<tr>
<td>00040000</td>
<td>* RCR 4 RETURN CODE</td>
</tr>
<tr>
<td>00050000</td>
<td>* ATTRR 40 ATTRIBUTES</td>
</tr>
<tr>
<td>00060000</td>
<td>EJECT</td>
</tr>
<tr>
<td>00070000</td>
<td>L R2,0(R1) PARM INPUT</td>
</tr>
<tr>
<td>00080000</td>
<td>LH R3,0(R2) LENGTH OF PARM</td>
</tr>
<tr>
<td>00090000</td>
<td>LA R4,2(R2) POINT TO PARM DATA</td>
</tr>
<tr>
<td>00100000</td>
<td>MVI DSN,C' '</td>
</tr>
<tr>
<td>00110000</td>
<td>MVC DSN+1(43),DSN BLANK IT</td>
</tr>
<tr>
<td>00120000</td>
<td>SH R3,=H'1' REDUCE LENGTH FOR EXECUTE</td>
</tr>
<tr>
<td>00130000</td>
<td>EX R3,MYDSN</td>
</tr>
<tr>
<td>00140000</td>
<td>MVC DSNR,DSN</td>
</tr>
<tr>
<td>00150000</td>
<td>CALL OBTMODE,(DSNR,RCR,ATTRR)</td>
</tr>
<tr>
<td>00160001</td>
<td>OPEN (PRINTDCB,(OUTPUT))</td>
</tr>
<tr>
<td>00170000</td>
<td>PUT PRINTDCB,PRINTL0</td>
</tr>
<tr>
<td>00180000</td>
<td>MVC DSNAME,DSNR</td>
</tr>
<tr>
<td>00190000</td>
<td>MVC ATTRIB(12),ATTRR</td>
</tr>
<tr>
<td>00200000</td>
<td>MVC DSCBFMT,FMTR</td>
</tr>
<tr>
<td>00210000</td>
<td>L R1,RCR</td>
</tr>
<tr>
<td>00220000</td>
<td>CVD R1,DWORD</td>
</tr>
<tr>
<td>00230000</td>
<td>OI DWORD+7,X'0F'</td>
</tr>
<tr>
<td>00240000</td>
<td>UNPK PWORD(11),DWORD+2(6)</td>
</tr>
<tr>
<td>00250000</td>
<td>MVC RCODE,PWORD+9</td>
</tr>
<tr>
<td>00260000</td>
<td>CH R1,=H'0'</td>
</tr>
<tr>
<td>00270000</td>
<td>BNE TRUNCATE</td>
</tr>
<tr>
<td>00280000</td>
<td>LH R1,DSCBS</td>
</tr>
<tr>
<td>00290000</td>
<td>CVD R1,DWORD</td>
</tr>
<tr>
<td>00300000</td>
<td>OI DWORD+7,X'0F'</td>
</tr>
<tr>
<td>00310000</td>
<td>UNPK PWORD(11),DWORD+2(6)</td>
</tr>
<tr>
<td>00320000</td>
<td>MVC DSCBNUM,PWORD+1</td>
</tr>
<tr>
<td>00330000</td>
<td>CLC ATTRR(2),=CL2'VS'</td>
</tr>
<tr>
<td>00340000</td>
<td>BE TRUNCATE</td>
</tr>
<tr>
<td>00350000</td>
<td>L R1,LRECLR</td>
</tr>
<tr>
<td>00360000</td>
<td>CVD R1,DWORD</td>
</tr>
<tr>
<td>00370000</td>
<td>OI DWORD+7,X'0F'</td>
</tr>
<tr>
<td>00380000</td>
<td>UNPK PWORD(11),DWORD+2(6)</td>
</tr>
<tr>
<td>00390000</td>
<td>MVC LRECL,PWORD+1</td>
</tr>
<tr>
<td>00400000</td>
<td>L R1,BLKR</td>
</tr>
<tr>
<td>00410000</td>
<td>CVD R1,DWORD</td>
</tr>
<tr>
<td>00420000</td>
<td>OI DWORD+7,X'0F'</td>
</tr>
<tr>
<td>00430000</td>
<td>UNPK PWORD(11),DWORD+2(6)</td>
</tr>
<tr>
<td>00440000</td>
<td>MVC BLKSIZ,PWORD+1</td>
</tr>
<tr>
<td>00450000</td>
<td>TRUNCATE DS OH</td>
</tr>
<tr>
<td>00460000</td>
<td>S PUT PRINTDCB,PRINTL1</td>
</tr>
<tr>
<td>00470000</td>
<td>* DC H'0' CAUSE ABEND</td>
</tr>
<tr>
<td>00480000</td>
<td>RETURN DS OH</td>
</tr>
<tr>
<td>00490000</td>
<td>SEGEND</td>
</tr>
<tr>
<td>00500000</td>
<td>WORKAREA DS OD LOCATE WORK AREA (265 BYTES)</td>
</tr>
<tr>
<td>00510000</td>
<td>DSN DS CL44</td>
</tr>
</tbody>
</table>

Figure C-30  OBTDRVE program source to call OBTMODE routine (1 of 2)
**MVDSN**    MVC   DSN(O),0(R4)  CODE TO EXECUTE 00550000
  DS    OF 00560000
**DSNR**    DS    CL44 00570000
**RCR**    DS    CL4 00580000
  DC    C'ATTRATTR' 00590000
**ATTRR**    DS    CL40 00600000
  ORG   ATTRR+12 00610000
**LRECLR**    DS    F 00620000
**BLKR**    DS    F 00630000
**FMTR**    DS    CL2 00640000
**DSCBS**    DS    H 00650000
  ORG   00660000
**DWORD**    DS    D 00670000
**PWORD**    DC    CL11' ' 00680000
  * 00690000
**PRINTDCB**    DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133 00700000
  * 00710000
**PRINTL0**    DC    CL133' ' 00720000
  ORG   PRINTL0+1 00730000
**DSNAMEH**    DC    CL44'DSNAME' 00740000
  DC    C' ' 00750000
**RCODEH**    DC    CL4'RC' 00760000
  DC    C' ' 00770000
**ATTRIH**    DC    CL40'ATTRIBS' 00780000
  ORG   ATTRIH+12 00790000
  DC    CL10'LRECL' 00800000
  DC    C' ' 00810000
  DC    CL10'BLKSIZ' 00820000
**DSCBFMTH**    DC    CL2'DS' 00830000
  DC    C' ' 00840000
**DSCBNUMH**    DC    CL10'#DSCBS' 00850000
  ORG   00860000
  * 00870000
**PRINTL1**    DC    CL133' ' 00880000
  ORG   PRINTL1+1 00890000
**DSNAME**    DC    CL44' ' 00900000
  DC    C' ' 00910000
**RCODE**    DC    CL4' ' 00920000
  DC    C' ' 00930000
**ATTRIB**    DC    CL40' ' 00940000
  ORG   ATTRIB+11 00950000
  DC    C' ' 00960000
**LRECL**    DC    CL10' ' 00970000
  DC    C' ' 00980000
**BLKSIZ**    DC    CL10' ' 00990000
**DSCBFMT**    DC    CL2'XX' 01000000
  DC    C' ' 01010000
**DSCBNUM**    DC    CL10' ' 01020000
  ORG   01030000
**END** 01040000

---

**Figure C-31**   **OBTDRVE** program source to call **OBTMODE** routine (2 of 2)
SEGSTART and SEGEND are macros to set up the routine and establish the environment, and can be ignored as far as the logic is concerned.

The program is intended to display the following information:

1. Data set name
2. Return code from OBTAIN
   a. 0 (x'00') - OK
   b. 4 (x'04') - required volume not mounted
   c. 8 (x'08') - DSCB not found on the volume specified
   d. 12 (x'0C') - Permanent error
   e. 16 (x'10') - Invalid workarea pointer
   f. 24 (x'18') - Data set has Format-8 DSCB and EADSCB=NOTOK
   g. 28 (x'1C') - Internal error and EADSCB=NOTOK
3. Data set attributes - Organization, RECFM, RECFM-X, LRECL, BLKSIZE
   a. Organization - PO, PS, VS etc.
   b. RECFM - F/FB/FBA/V/VA/VB/VBA etc. - replaced by **** for VS organization
   c. RECFM-X as for RECFM but the carriage control character, if present, is separated out
   d. LRECL - Logical record length - suppressed for VS origination
   e. BLKSIZE - Data set Blocksize - suppressed for VS organization
4. DSCB Format Identifier
   a. F1/F4/F8 etc.
5. Number of DSCBs associated with the data set
   a. Number of DSCBS associated with the data set

Additional information:

The data set whose attributes are wanted is passed as a PARM to the program.

After initialization, the basic logic is:
1. Call subroutine OBTMODE
2. OPEN the DCB
3. Print the Header
4. Interpret the data and format
5. Print the detail
6. Exit
In Figure C-32 we show an example of a job executing the OBTDRAVE program several times.

```
//MHLRES1C JOB (1234567,COMMENT),UAALF0,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*JOBPARM S=* 
//CHKDSN PROC
//ASM EXEC PGM=OBTDRAVE,REGION=0M,
//             PARM='MHLRES1.EAV.JCL'
//STEPLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//SYSUDUMP DD SYSOUT=* 
//PRINT DD SYSOUT=* 
//PEND
// EXEC CHKDSN,PARM='MHLRES1.EAV.JCL'
// EXEC CHKDSN,PARM='MHLRES1.DDIR.D'
// EXEC CHKDSN,PARM='MHLRES1.DDIR.DUD'
// EXEC CHKDSN,PARM='MHLEAV.EXTKSDS.DATA'
// EXEC CHKDSN,PARM='MHLEAV.EXTGELA.TEST0005.DATA'
// EXEC CHKDSN,PARM='MHLEAV.EXTGELA.TEST0005.INDEX'
// EXEC CHKDSN,PARM='MHLEAV.PE.D195653.DATA'
// EXEC CHKDSN,PARM='MHLEAV.PE.D195653.INDEX'
```

*Figure C-32  Example of JOB to examine several data sets using OBTDRAVE*

In Figure C-33 we show the result of running the JCL to execute program OBTDRAVE several times.

```
DSNAME                                      RC  ATTRIBS  LRECL  BLKSIZE  DS #DSCBS
MHLRES1.EAV.JCL                              00  PO  FB  FB  0000000000 0000027920 1 0000000001
MHLRES1.DDIR.D                               00  VS  ********                      1 0000000001
MHLRES1.DDIR.DUD                             12
MHLEAV.EXTKSDS.DATA                         RC  ATTRIBS  LRECL  BLKSIZE  DS #DSCBS
MHLEAV.EXTGELA.TEST0005.DATA                 12
MHLEAV.EXTGELA.TEST0005.INDEX                12
MHLEAV.PE.D195653.DATA                      00  VS  ********                      8 0000000002
MHLEAV.PE.D195653.INDEX                     00  VS  ********                      8 0000000002
```

*Figure C-33  OBTDRAVE output from sample*

This output shows the following results:

- **MHLRES1.EAV.JCL** is a partitioned data set, and was retrieved successfully (RC=0).
- **MHLRES1.DDIR.D** is the data portion of a VSAM data set, not in an EAV volume and was retrieved successfully (RC=0).
- **MHLRES1.DDIR.DUD** was not found (RC=12).
- **MHLEAV.EXTKSDS.DATA** is the data portion of a VSAM data set. It was located on an EAV volume successfully. It has a Format-8 DSCB and there are 2 DSCBs in the chain (RC=00).
- **MHLEAV.EXTGELA.TEST0005.DATA** was not found (RC=12).
- **MHLEAV.EXTGELA.TEST0005.INDEX** was not found (RC=12).
MHLEAV.PE.D195653.DATA is the data portion of a VSAM data set. It was located on an EAV volume successfully. It has a Format-8 DSCB and there are 2 DSCBs in the chain (RC=00).

MHLEAV.PE.D195653.DATA is the data portion of a VSAM data set. It was located on an EAV volume successfully. It has a Format-8 DSCB and there are 2 DSCBs in the chain (RC=00).

The enhanced output is provided by the OBTMODE subroutine driven by the OBTDRVE program.

C.4.2.3 Execution of sample OBTAINT programs
The subroutines and driver programs are working examples. If you want to run a program, you need to assemble its subroutine, and assemble the program and link the subroutine with it. It is not necessary to have in depth assembler experience, but familiarity with JCL is required.

The source and support macros are stored in one PDS, the JCL in one PDS, and the LOAD modules in another.

To create the assembler PDS, use the following steps.

There are three steps to build the program which needs to be done once, after which it can be executed several times.

Step 1: Create a PDS/PDSE to hold the source members
In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

Step 2: Create a PDS/PDSE to hold the LOAD MODULES
In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 0/U. All other attributes can be chosen by you.

Step 3: Store the program source in the PDS
Make sure that your ISPF EDIT menu settings has the screen format defined as DATA, the emulator line length is set to greater than 80, and that the edit session is in NONUM mode.

Choose whether to build OBTDRVE or OBTDRVO. This example assumes that OBTDRVE is being built.

Cut and paste the contents of Figure C-30 on page 524 and Figure C-31 on page 525 one after the other into member OBTDRVE. The result should contain 104 lines.

Cut and paste the contents of Figure C-27 on page 521 through to Figure C-29 on page 523 one after the other into member OBTMODE. The result should contain 126 lines.

In order to assemble OBTDRVE and OBTMODE, supporting macros are needed. Listing of the macros are provided. They need to be stored as members in MHLRES1.EAV.ASM. The same macros are used for both drivers and subroutines. Refer to “Store the macro source in a PDS” on page 630 for information about storing the macros.
Step 4: Create a PDS/PDSE to hold the JCL members

In this example the PDS is called MHLRES1.EAV.JCL.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

Step 5: Store the JCL source in the PDS

- Cut and paste the contents of Figure C-34 on page 529 into member OBTMODE. The result should contain 28 lines.
- Edit member OBTMODE and change the references to name to OBTMODE (on the ASM.SYSIN DD statement, and after the LKED.SYSIN statement).

```
//MHLRES1C JOB (1234567,COMMENT),UAALFO,TIME=10,
  // MSGLEVEL=1,CLASS=A,
  // NOTIFY=&SYSUID
  /*JOBPARM S**
  //ASMHCL PROC
  //ASM EXEC PGM=ASMA90,REGION=0M,
  //      PARM='OBJECT,NODECK'
  //SYSLIN DD DSN=&&OBJ,DISP=(NEW,PASS),UNIT=SYSDA,
  //       SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120
  //SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB
  //       DD DISP=SHR,DSN=SYS1.MODGEN
  //       DD DISP=SHR,DSN=MHLRES1.EAV.ASM
  //SYSPRINT DD SYSOUT=*     //SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
  //SYSLIB DD DDNAME=SYSIN
  //SYSPRINT DD SYSOUT=*     //SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
  //SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
  //PEND
  // EXEC ASMHCL
  //ASM.SYSIN DD DISP=SHR,DSN=MHLRES1.EAV.ASM(name)
  /*
  //LKED.SYSLMOD DD DSN=MHLRES1.EAV.LOAD,DISP=SHR
  //LKED.SYSIN DD *
  NAME name(R)
```

Figure C-34 JCL to assemble and link OBTMODE and OBTDRVE

- Cut and paste the contents of Figure C-34 on page 529 into member OBTDRVE (or copy the member just created called OBTMODE). The result should contain 28 lines.
- Edit member OBTDRVE and change the references to name to OBTDRVE (on the ASM.SYSIN DD statement, and after the LKED.SYSIN statement (or if you used a copy of OBTMODE, change reference to OBTMODE to OBTDRVE)).

Step 7: Run the job in member OBTMODE

Check that the return code from the ASM and LKED steps is 0.

A return code of 8 on the ASM step is an indication that the macros required have not been stored in MHLRES1.EAV.ASM.
Step 8: Run the job in member OBTDREVE
Check that the return code from the ASM and LKED steps is 0.

A return code of 8 on the ASM step is an indication that the macros required have not been stored in MHLRES1.EAV.ASM.

A return code of 8 on the LKED step is an indication that the subroutine required has not been stored in MHLRES1.EAV.LOAD.

After achieving return codes of 0 the program is ready to use.

C.5 VTOC management macros extended for EAV: REALLOC

The Realloc macro is extended for EAV support by the addition of the F9DSCB and NUMF9 operands.

In order to use the REALLOC macro, a Format-3 or a Format-9 DSCB must be provided. Because it is necessary to use the OBTAIN macro to retrieve a typical DSCB, refer to C.4, “VTOC management macros extended for EAV: OBTAIN” on page 510 for information about handling code changes to support the OBTAIN macro with EAV volumes.

C.6 VTOC management macros extended for EAV: CVAFDIR

The CVAFDIR macro and underlying services is provided to perform the following functions directly on the VTOC or VTOC index.

For the standard VTOC you can use the CVAFDIR macro to perform the following functions:
- Read or write one or more DSCBs by specifying the name of the data set they represent.
- Read or write one or more DSCBs by specifying their addresses.

In addition, for an indexed VTOC, you can use the CVAFDIR macro to perform the following functions:
- Read or write VTOC index records. (This allows calling programs to modify the VTOC index.)
- Read and retain in virtual storage the first high-level VIER, and VIERS used during an index search.
- Read and retain in virtual storage the space map VIRs.
- Free VIRs retained in virtual storage.

Attention: The CVAFDIR macro can be used to write to the VTOC. It is essential that all the elements of the VTOC be properly constructed, so any use of CVAFDIR to update the VTOC must be carefully managed. It is particularly important that proper serialization on the VTOC be set up. When setting up serialization on the VTOC the effects on other activities against the VTOC must be considered to avoid locking other processes out.
The CVAFDIR macro as implemented in z/OS releases prior to z/OS V1R10, and if not extended by EAV keywords is not permitted to retrieve information from EAV formatted volumes. Use of an un-extended CVAFDIR macro call against a VTOC on an EAV volume will result in a non zero return code, a non-zero status byte and the data requested will not be returned.

Changes to a program issuing the CVAFDIR macro call are required to allow access to EAV volumes. Existing programs without the EAV enabling operand on the CVAFDIR call will be tracked by the Migration Assistance Tracker but will not fail if run against a non-EAV volume. Such programs will fail if run against an EAV volume.

We have provided an example of a program that issues the CVAFDIR macro in the format used prior to z/OS 1.10, and as extended to support EAV to illustrate the type of change required to support EAV.

The purpose of this example program is to:

- Demonstrate what happens when an enhanced macro call is issued against a non-EAV volume
- Demonstrate what happens when an enhanced macro call is issued against an EAV volume
- Demonstrate what happens when an un-enhanced macro call is issued against a non-EAV volume
- Demonstrate what happens when an un-enhanced macro call is issued against an EAV volume
- Demonstrate that the Migration Assistance Tracker picks up the CVAFDIR un-enhanced macro against a non-EAV volume
- Illustrate the code changes required to support EAV format volumes.

The example program uses data prepared by running a program that issues the CVAFFILT macro to retrieve information from a volume so the example is a two step process.
In Figure C-35 we show the JCL to examine volumes MLDC65, MHLS1A, and MLDC66.

```plaintext
//MHLRESID JOB (1234567,COMMENT),UAALF0,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*/JOBPARM S=* 
//CVDIRIT PROC
//RUNF EXEC PGM=CVFLT0T,REGION=0M
//STEPLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//CVAFDD DD DISP=SHR,UNIT=SYSALLDA,Vol=SER=&VOLSER
//FILTDD DD DSN=MHLRES1.EAV.DSNCCHHR,
//        DISP=(NEW,CATLG,DELETE),
//        UNIT=3390,
//        SPACE=(TRK,(1,1)),
//        DCB=(LRECL=49,BLKSIZE=4900,RECFM=FB,DSORG=PS)
//SYSUDUMP DD SYSOUT=* 
//OUTDD DD SYSOUT=* 
//RUNG EXEC PGM=IEBGENER,REGION=0M
//SYSPRINT DD SYSOUT=* 
//SYSUT1 DD DSN=MHLRES1.EAV.DSNCCHHR,
//        DISP=SHR 
//SYSUT2 DD SYSOUT=* 
//SYSIN DD DUMMY
//RUND EXEC PGM=CVDIR1E,REGION=0M
//STEPLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//CVAFDD DD DISP=SHR,UNIT=SYSALLDA,Vol=SER=&VOLSER
//FILTDD DD DSN=MHLRES1.EAV.DSNCCHHR,
//        DISP=(SHR,DELETE,DELETE)
//SYSUDUMP DD SYSOUT=* 
//OUTDD DD SYSOUT=* 
//PEND
//VMLDC65 EXEC CVDIR1T,VOLSER=MLDC65 
///*
//VMHLS1A EXEC CVDIR1T,VOLSER=MHLS1A,COND=EVEN
//VMLDC66 EXEC CVDIR1T,VOLSER=MLDC66,COND=EVEN
```

Figure C-35 Example of JOB CVDIR1EJ to examine volumes using CVDIR1E

The first step, labelled RUNF uses CVAFFILT to extract a data set name and its CCHHR.

The second step labelled RUNG uses IEBGENER to print the extracted data set name and its CCHHR.

The third step labelled RUND uses CVAFDIR to READ the DSCB

The process is applied to three volumes using the inline JCL procedure CVDIR1E. The second and third invocations have COND=EVEN coded because the program might abend.
In Figure C-36 we show the result of running the JCL to execute program CVDIR1E on volume MLDC65 (EAV). This a a program that uses the CVAFFILT macro to get the data set name and its CCHHR from the designated volume, and write the name and CCHR in a data set to pass to subsequent steps.

<table>
<thead>
<tr>
<th>CVFLT0T START OF OUTPUT MESSAGES</th>
<th>VOLUME: MLDC65</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVAFFILT CALL: EADSCB=OK</td>
<td>LIST/ORDER/F1-F8</td>
</tr>
<tr>
<td>CV4EADOK BIT SET / EADSCB=OK</td>
<td></td>
</tr>
<tr>
<td>RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL</td>
<td>CVSTAT CODE: X&quot;00&quot; DEC&quot;000&quot;</td>
</tr>
<tr>
<td>MHLEAV.EXTKSDS.DATA DSN AND CCHHR RECORD WRITTEN</td>
<td></td>
</tr>
<tr>
<td>VERIFY CVAFFILT ACCESS=RLSE EADSCB=NOTOK IGNORED</td>
<td>RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL</td>
</tr>
<tr>
<td></td>
<td>CVSTAT CODE: X&quot;00&quot; DEC&quot;000&quot;</td>
</tr>
<tr>
<td>CVFLT0T END OF OUTPUT MESSAGES</td>
<td></td>
</tr>
</tbody>
</table>

Figure C-36  CVFLT0T output (step 1) from sample job CVDIR1EJ

This output shows the following results:

- The messages relate to volume MLDC65 (EAV).
  - CVAFFILT CALL: EADSCB=OK
    - Result is SUCCESS with CVSTAT X"00" (DEC(00))
  - CVAFFILT CALL: EADSCB=NOTOK KEYWORD CODED
    - Result is SUCCESS with CVSTAT X"00" (DEC(00)) showing that the CVAFFILT ACCESS=RLSE operand is not sensitive to whether the volume is EAS eligible or not.

In Appendix C-37, “IEBGENER output (step 2) from sample job CVDIR1EJ (SDSF with SET HEX ON)” on page 533 we show the display of the output from the IEBGENER execution which is step 2 of the procedure.

Figure C-37  IEBGENER output (step 2) from sample job CVDIR1EJ (SDSF with SET HEX ON)

This output show the following results:

- HEX display of the contents of the data set created by CVFLT0T. The data set name occupies columns 1-44 and the CCHHR columns 45-49
In Figure C-38 we show the display of the output from the CVDIR1E execution which is step 3 of the procedure.

<table>
<thead>
<tr>
<th>CVDIR1E START OF OUTPUT MESSAGES</th>
<th>VOLUME: MLDC65</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCESSING DSN: MHLEAV.EXTKSDDS.DATA</td>
<td>(CVAFDIR READ)</td>
</tr>
<tr>
<td>CV4EADOK BIT SET / EADSCB=OK</td>
<td></td>
</tr>
<tr>
<td>CV4MULTD BIT SET / MULTIPLEDSCBS=YES</td>
<td></td>
</tr>
<tr>
<td>RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL</td>
<td></td>
</tr>
<tr>
<td>CVSTAT CODE: X&quot;00&quot; DEC&quot;000&quot;</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROCESSING DSN: MHLEAV.EXTKSDDS.DATA</th>
<th>(CVAFDIR WRITE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CV4EADOK BIT SET / EADSCB=OK</td>
<td></td>
</tr>
<tr>
<td>CV4MULTD BIT NOT SET / MULTIPLEDSCBS=NO</td>
<td></td>
</tr>
<tr>
<td>RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL</td>
<td></td>
</tr>
<tr>
<td>CVSTAT CODE: X&quot;00&quot; DEC&quot;000&quot;</td>
<td></td>
</tr>
</tbody>
</table>

VERIFY CVAFDIR ACCESS=RLSE EADSCB=NOTOK IGNORED
RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
CVSTAT CODE: X"00" DEC"000"

CVDIR1E END OF OUTPUT MESSAGES

This output shows the following results:

- The messages relate to volume MLDC65 (EAV).
  - CVAFDIR READ CALL: EADSCB=OK
    - Result is SUCCESS with CVSTAT X"00" (DEC(00))
  - CVAFDIR WRITE CALL: EADSCB=OK
    - Result is SUCCESS with CVSTAT X"00" (DEC(00))
  - CVAFDIR CALL: EADSCB=NOTOK KEYWORD CODED
    - Result is SUCCESS with CVSTAT X"00" (DEC(00)) showing that the CVAFDIR ACCESS=RLSE operand is not sensitive to whether the volume is EAS eligible or not.
C.6.1 EAV Migration Assistance Tracker considerations: CVAFDIR

The EAV Migration Assistance Tracker will detect use of the CVAFDIR service that has not been updated for EAV compatibility.

Any appearance on the tracker list must be investigated and resolved because use of an un-extended OBTAIN against an EAV volume in due course will fail.

For information about the EAV Migration Tracker, refer to Appendix C of DFSMS Advanced Services, SC26-7400, and in this document to “Migration Assistance Tracker” on page 641.

In Figure C-39 we show the output from the Migration Assistance Tracker that relates to the run of CVDIR1X run against volume MHLS1A (a non EAV volume). It shows that CVDIR1X needs attention before it is used against an EAV format volume. CVDIR1X is a version of the CVDIR1E program described next but without EAS data set coding.

![Figure C-39 CVDIR1X Migration Assistance Tracker entry](image)

**Source code listing**

The CVDIR1EJ example job uses two programs

- CVFLT0T
- CVDIR1E

CVFLT0T is used as a utility to extract the data set CCHHR. It is coded using CVAFFILT with EADSCB=OK. For information about coding for the CVAFFILT macro in general, see C.8, “VTOC management macros extended for EAV - CVAFFILT” on page 572.

Program CVDIR1E is coded with EADSCB=OK and we follow its source listing with information on what would be changed to create a version of the program that is not EAV capable.

In Figure C-40 on page 536 through to Figure C-48 on page 544 we show assembler source for a program to issue the CVAFFILT macro to get the CCHHR for a data set.
CVFLTOT TITLE 'CVAF CVAFFILT TEST MODULE' 00010002
CVFLTOT CSECT 00020002
CVFLTOT AMODE 31 00030002
CVFLTOT RMODE 24 00040002
*********************************************************************** 00050002
* * CVFLTOT - CVAFFILT TESTCASE * 00060002
* * CVAFFILT MACRO RUN IN AMODE31/RMODE24 * 00070002
* * CVAFFILT TEST DSN'S / PROCESSING USED: * 00080002
* * 1) DATASETS AS DESCRIBED BELOW: * 00090002
* * SEQ01 - MHLEAV.EXTKSDS.DATA - SEQ DS WITH 1 EXTENT * 00100002
* * THIS MODULE WILL ISSUE A CVAFFILT CALL FOR AN ORDERED FCL * 00110002
* * REQUESTING ONLY THE FMT1 OR FMT8 FOR THE DATASET. THIS CALL * 00120002
* * WILL USE 1 BUFFERS, ONE FOR EACH DATASET. WE WILL THEN PROCESS * 00130002
* * THE DSCBS RETURNED FOR THE DATASETS AND CREATE AN OUTPUT FILE * 00140002
* * THAT CONSISTS OF A RECORD FOR EACH OF THE DATASETS THAT CONSIST * 00150002
* * OF THE DSN AND CCHHR. THIS OUTPUT FILE WILL BE USED AS INPUT * 00160002
* * TO THE CVAFDIR MODULES LATER IN THIS TESTCASE. * 00170002
* NOTE: THIS MODULE REQUIRES THE USE OF THE CVSTCHK MODULE WHICH * 00180002
* IS CALLED TO VERIFY THE CVSTAT CODE RETURNED. * 00190002
* THE TESTCASE $$STAT CREATES THIS MODULE IN LINKLIB. * 00200002
* OUTPUT IN OUTDD DATASET SHOULD MATCH THE FOLLOWING: * 00210002
*---------------------------------------------------------------------* 00220002
* EXAMPLE - NON EAV VOLUME OR EAV VOLUME * 00230002
* -------------------------------------- * 00240002
* CVFLTOT START OF OUTPUT MESSAGES * 00250002
* CVAFFILT CALL: EADSCB=OK LIST/ORDER/F1-F8 * 00260002
* CV4EADOK BIT SET / EADSCB=OK * 00270002
* RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL * 00280002
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED * 00290002
* MHLEAV.EXTKSDS.DATA DSN AND CCHHR RECORD WRITTEN * 00300002
* VERIFY CVAFFILT ACCESS=RLSE EADSCB=NOTOK IGNORED * 00310002
* VERIFY CVAFFILT ACCESS=RLSE EADSCB=NOTOK IGNORED * 00320002
* RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL * 00330002
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED * 00340002
* CVFLTOT END OF OUTPUT MESSAGES * 00350002
* *********************************************************************** 00360002

Figure C-40  Source of CVFLTOT program (part 1 of 9)
Appendix C. Code samples DFSMS V1.10

Figure C-41   Source of CVFLT0T program (part 2 of 9)

* CVFLT0T - LOGIC NOTES *
  * THIS MODULE WILL PERFORM THE FOLLOWING: *
  * INITIALIZATION *
  * - OBTAIN THE NECESSARY INFORMATION FROM THE DASD VOLUME *
  * - OPEN THE OUTPUT FILES AND WRITE THE NECESSARY OUTPUT MESSAGES *
  * MAINLINE *
  * - INITIALIZE THE NECESSARY BUFFER LIST FOR CVAFFILT (1 ENTRY) *
  * - INITIALIZE A FCL TO READ LIST/ORDER/F1-F8 *
  * - ISSUE CVAFFILT READ (EADSCB=OK) FOR FCL LIST/ORDER/F1-F8 *
  * - CHECK THE RETURN CODE AND CVSTAT CODE FROM CVAFFILT *
  * - MOVE DSN / CCHHR FOR DSNS TO FILTOUT FOR LATER CVAFDIR CALLS *
  * FINALIZATION *
  * - CLOSE THE OUTPUT FILE *
  * - EXIT *
  * CVFLT0T - JOB INFORMATION *
    * NORMAL END OF JOB: *
      * - RC=00 AND OUTDD OUTPUT AS DETAILED ABOVE *
    * ABNORMAL END OF JOB: *
      * - ABEND 100 - ERROR OPENING VTOC ON THE DASD VOLUME THAT IS *
        * ASSOCIATED WITH THE CVAFDD DD STATEMENT *
      * - ABEND 101 - ERROR OPENING THE OUTDD DATASET *
      * - ABEND 102 - ERROR OPENING THE FILTDD DATASET *
      * - ABEND 103 - ERROR CLOSING THE OUTDD DATASET *
      * - ABEND 104 - ERROR CLOSING THE FILTDD DATASET *
    * HOUSEKEEPING *
      * - SAVE CALLER'S REGISTERS AND ESTABLISH A NEW REGISTER SAVE AREA *
        * STM R14,R12,12(R13) STANDARD LINKAGE CONVENTION *
        * BALR R10,0 R10 IS IMPLIED BASE REG *
        * USING BASE,R10,R11,R12 R11 AND R12 ALSO BASE REGS *
        * BASE L R11,BASEAD1 SET UP ADDRESSING FOR R11 *
        * L R12,BASEAD2 SET UP ADDRESSING FOR R12 *
        * B CV000000 BRANCH AROUND DECLARES *
        * BASEAD1 DC A(BASE+4096) ADDRESSING FOR BASE R11 *
        * BASEAD2 DC A(BASE+8192) ADDRESSING FOR BASE R12 *
        * CV000000 DS OH HOUSEKEEPING... *
        * ST R13,SAVE+4 SAVE PTR TO CALLER'S SAVE AREA *
        * LA R14,SAVE GET ADDRESS OF THE NEW SAVE AREA *
        * ST R14,8(R13) CHAIN CALLER'S AREA TO Ours *
        * LR R13,R14 ESTABLISH THE NEW SAVE AREA *

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

Housekeeping
- Standard linkage convention
- R10 is implied base register
- R11 and R12 are also base registers
- Set up addressing for R11
- Set up addressing for R12
- Branch around declares
- Addressing for base R11
- Addressing for base R12
- Establish the new save area

***********************************************************************
* INITIALIZATION

INITIAL DS OH INITIALIZATION SECTION
BAL R14,IVOLRTN INVOKE RTN TO IDENTIFY THE VOLUME(S)
OPEN (OUTFILE,(OUTPUT)) OPEN THE OUTPUT MESSAGE FILE
TM OUTFILE+48,X'10' IF OPEN OF OUTPUT FILE NOT OK
BO OK1
    ABEND 101 ISSUE USER ABEND 101
OK1 DS OH
OPEN (FILTOUT,(OUTPUT)) OPEN THE OUTPUT DSN/CCHHR FILE
TM FILTOUT+48,X'10' IF OPEN OF OUTPUT FILE NOT OK
BO OK2
    ABEND 102 ISSUE USER ABEND 102
OK2 DS OH

* MAINLINE

MAINLINE DS OH MAINLINE SECTION
* 1 BUFFERS, FCL1LIST/FCL1ORDR/FCL1EQF1 EADSCB=OK
* 01260002
* 01270002
BAL R14,BUF1RTN INVOKE RTN TO INIT BUFFERS (1)
BAL R14,FCL1RTN INVOKE RTN TO INIT FCL (LIST)
PUT OUTFILE,STRTMSG WRITE A RECORD TO THE OUTPUT FILE
PUT OUTFILE,BLNKLINE WRITE A RECORD TO THE OUTPUT FILE

* TEST THAT EADSCB=NOTOK IS IGNORED FOR RLSE CALL
* 01360002
* 01370002
PUT OUTFILE,RLSEOKM WRITE RLSE OK MESSAGE
* 01390002

* FINALIZATION

FINAL DS OH FINALIZATION SECTION
PUT OUTFILE,BLNKLINE WRITE A RECORD TO THE OUTPUT FILE
PUT OUTFILE,ENDMSG WRITE A RECORD TO THE OUTPUT FILE
CLOSE (OUTFILE) CLOSE OUTPUT FILE
C R15,RCODE00 IF FILE CLOSE IS NOT OK
BE OK3
    ABEND 103 ISSUE USER ABEND 103
OK3  DS    OH                                                       01560002
C   R15,RCODE00  IF FILE CLOSE IS NOT OK                           01570002
BE  OK4                                                     01580002
ABEND 104        ISSUE USER ABEND 104                             01590002
OK4  DS    OH                                                       01600002
L   R13,4(R13)       RESTORE REGISTER                              01610002
LM  R14,R12,12(R13)  RESTORE CALLERS REGISTERS                     01620002
LA  R15,0         SET RC TO 0                                     01630002
BR  R14         RETURN TO CALLER                                  01640002
*                                                                01650002
*********************************************************************** 01660002
*                          IDVOLRTN                                   * 01670002
*                          - OBTAIN THE NECESSARY INFORMATION FROM THE DASD VOLUME  * 01680002
IDVOLRTN  DS    OH                                               01690002
ST  R14,IDVLSAVE       STORE C(R14) INTO SAVE AREA                01700002
RDJFCB (VTOCDCB,(INPUT)) READ JFCB / OPEN VTOC                   01710002
MVI  JFCB1,X'04'     PUT IN ID FOR FORMAT 4                      01720002
MVC  JFCB1+1(43),JFCB1 SETUP FOR VTOC OPEN                      01730002
OPEN (VTOCDCB,(INPUT)),TYPE=J OPEN VTOC (OPEN TYPE=J)            01740002
TM  VTOCDCB+48,X'10'  IF OPEN OF VTOC NOT OK                      01750002
BO  OK5                                                  01760002
ABEND 100        ISSUE USER ABEND 100                            01770002
OK5  DS    OH                                                       01780002
SLR  RDEB,RDEB        INIT REG1 FOR DEB PTR                       01790002
SLR  RUCB,RUCB        INIT REG2 FOR UCB PTR                       01800002
ICM  RDEB,B'0111',VTOCDCB+45  GET DEB ADDRESS                   01810002
ST  RDEB,DEBADD       SAVE DEB ADDRESS INTO R1                   01820002
ICM  RUCB,B'0111',33(RDEB)    GET UCB ADDRESS                   01830002
ST  RUCB,UCBADD       SAVE UCB ADDRESS INTO R2                   01840002
IDVLEXIT  DS    OH                                             01850002
L   R14,IDVLSAVE       LOAD C(IDVLSAVE) INTO R14                 01860002
BR  R14        EXIT                                           01870002
*********************************************************************** 01880002
*                          TSTRCRTN                                   * 01890002
*                          - TEST RETURN CODE FROM CVAFFILT          * 01900002
TSTRCRTN  DS    OH                                             01910002
ST  R14,TSTRSAVE       STORE C(R14) INTO SAVE AREA                01920002
LA  R9,CVPLDEF         ESTABLISH ADDR - CVPLDEF                  01930002
USING CVPLMAP,R9      ADDR TO THE CVPL (FOR CVSTAT)               01940002
L   R15,RETCODE       LOAD R15 WITH RC SAVED FROM LAST CALL     01950002
C   R15,RCODE16       0    RC MESSAGE                             01960002
BNL  PRCERMSG         4    RC MESSAGE                             01970002
B   PROCESS(R15)     8    RC MESSAGE                             01980002
SPACE                                                            01990002
PROCESS  EQU   *                                                 02000002
B   PRC00MSG         0    RC MESSAGE                             02010002
B   PRC04MSG         4    RC MESSAGE                             02020002
B   PRC08MSG         8    RC MESSAGE                             02030002
B   PRC12MSG        12    RC MESSAGE                             02040002
B   PRC16MSG        16    RC MESSAGE                             02050002
SPACE                                                            02060002

Figure C-43   Source of CVFLT0T program (part 4 of 9)
PRC00MSG   PUT OUTFILE,RC00MSG   WRITE RC00 MESSAGE 02100002  
            B   PUTMSGE                      02110002
PRC04MSG   PUT OUTFILE,RC04MSG   WRITE RC04 MESSAGE 02120002  
            B   PUTMSGE                      02130002
PRC08MSG   PUT OUTFILE,RC08MSG   WRITE RC08 MESSAGE 02140002  
            B   PUTMSGE                      02150002
PRC12MSG   PUT OUTFILE,RC12MSG   WRITE RC12 MESSAGE 02160002  
            B   PUTMSGE                      02170002
PRC16MSG   PUT OUTFILE,RC16MSG   WRITE RC16 MESSAGE 02180002  
            B   PUTMSGE                      02190002
PRCERMSG   PUT OUTFILE,RCERMSG   WRITE RC ERROR MESSAGE 02200002  
            PUTMSGE DS OH                    02210002
            MVC WCYSTCD(1),CVSTAT          MOVE CVSTAT CODE TO WCYSTCD 02220002  
            BAL R14,CALLCVST               INVOKE CALLCVST TO CALL CVSTCHK MOD 02230002  
            DROP R9                        DROP R9 02240002
TSTREXIT DS OH                      EXIT FROM TSTRCRTN 02250002  
L R14,TSTRSAVE LOAD C(TSTRSAVE) INTO R14 02260002  
            BR R14                         EXIT 02270002
*********************************************************************** 02280002
*                           CALLCVST                               * 02290002
*     - ROUTINE TO CALL CVSTCHK FOR BINARY SEARCH OF CVSTAT TABLE   * 02300002
CALLCVST DS OH                 CALL TO CVSTCHK ROUTINE 02310002  
            ST R14,CALLSAVE              STORE C(R14) INTO SAVE AREA 02320002  
            LA R1,PARADDR1              LOAD R1 WITH PARMLIST FOR CALL 02330002  
            L R15,=V(CVSTCHK)          SET UP CALL TO CVSTCHK 02340002  
            BALR R14,R15               INVOKE CVSTCHK EXTERNAL SUBPROGRAM 02350002  
            PUT OUTFILE,WFMTREC1       WRITE A RECORD TO THE OUTPUT FILE 02360002
CALLEXIT DS OH                 EXIT FROM CALLCVST 02370002  
L R14,CALLSAVE LOAD C(CALLSAVE) INTO R14 02380002  
            BR R14                         EXIT 02390002
*********************************************************************** 02400002
*                           BUF1RTN                                * 02410002
*     - INITIALIZE BUFFER LIST HEADER (BFLH)                      * 02420002
*     - INITIALIZE BUFFER LIST ELEMENTS (BFLE) (1 ELEMENTS)      * 02430002
BUF1RTN  DS OH                  BUFFER LIST INITIALIZATION ROUTINE 02440002  
            ST R14,BUF1SAVE             STORE C(R14) INTO SAVE AREA 02450002  
            XC BFLHDEF(BFLSIZE),BFLHDEF  CLEAR BUFR LIST AREA 02460002  
            LA R1,BFLHDEF              R1 - BUFFER LIST HEADER 02470002  
            USING BFLMAP,R1            ESTABLISH ADDRESABILITY 02480002  
            MVI BFLHNOE,C1             SET NUMBER OF BUFFER ELEMENTS (1) 02490002  
            OI BFLHFL,BFLHDSCB       IDENTIFY AS DSCB BUFR ELEMENT LIST 02500002  
            LA R2,BFLHDEF+BFLHLM     R2 - FIRST BUFFER LIST ELEMENT 02510002  
            USING BFLE,R2              ESTABLISH ADDRESABILITY 02520002  
            LA R3,DSCBDEF             R3 - FIRST DSCB BUFFER 02530002  
            LA R4,C1                  R4 = NBR OF ELEMENTS AND BUFRS (1) 02540002
BF10010  DS OH                  DO UNTIL BUFNR = 0 02550002  
            OI BFLEFL,BFLECHR         REQUEST CCHR ON RETURN 02560002  
            MVI BFLELTH,DSCBSIZ   SET BUFR LENGTH TO FULL DSCB SIZE 02570002  
            ST R3,BFLEBUF            SET ADDR(DSCB BUFFER) 02580002  
            LA R2,BFLELN(R2)        R2 - NEXT BUFFER LIST ELEMENT 02590002  
            LA R3,DSCBSIZ(R3)       R3 - NEXT DSCB BUFFER 02600002  
            BCT R4,BF10010           ENDDO 02610002  
            DROP R1,R2              DROP TEMP USINGS 02620002
BUF1EXIT DS OH                 EXIT FROM BUF RTN 02630002  
L R14,BUF1SAVE LOAD C(SAVE AREA) INTO R14 02640002  
            BR R14                         EXIT 02650002

Figure C-44  Source of CVFLT0T program (part 5 of 9)
Appendix C. Code samples DFSMS V1.10

Figure C-45  Source of CVFLT0T program (part 6 of 9)
CVR1EXIT DS OH     EXIT FROM CVAFRD1          03200002
         L R14,CVR1SAVE LOAD C(SAVE AREA) INTO R14  03210002
         BR R14     EXIT                        03220002
*********************************************************************** 03230002
* PROSCBS     * 03240002
* - PROCESS RETURNED DSCBS AND BUFFER LIST ENTRIES AND CREATE     * 03250002
* OUTPUT FILE (FILTOUT) RECORDS OF DSN / CCHHR (49 BYTES)     * 03260002
PRDSCBS DS OH     PROCESS DSCBS - CREATE OUTPUT RECORDS 03270002
         ST R14,PRDSSAVE STORE C(R14) INTO SAVE AREA  03280002
         LA R2,BFLEDEF LOAD R2 WITH ADDR OF BUF LIST ENTRIES 03290002
         USING BFLE,R2 SET UP ADDRESSABILITY TO BUF LIST 03300002
         LA R3,BFLEBUF LOAD R3 WITH STARTING BUFFER ADDRESS 03310002
         L R4,CONE LOAD R4 WITH NUMBER OF BUFFERS (1)  03320002
*********************************************************************** 03330002
BCT1 DS OH     03340002
         CLC 0(44,R3),SEQ01 SELECT ON DSN IN BUFFER 03350002
         BNE NOTEQ1 03360002
         MVC ODSN(44),SEQ01 MOVE DSN TO OUTPUT RECORD 03370002
         MVC OCHHR,BFLEARG MOVE CCHHR FOR OUTPUT RECORD 03380002
         PUT OUTFILE,SEQ01M WRITE A REPORT REC TO OUTFILE 03390002
         PUT FILTOUT,OUTREC WRITE A OUTPUT REC TO FILTOUT 03400002
         NOTEQ1 DS OH 03410002
         LA R2,BFLELN(R2) INCREMENT R2 TO NEXT BUFF ENTRY 03420002
         LA R3,DSCBSIZ(R3) INCREMENT R3 TO NEXT DSCB BUFFER 03430002
         BCT R4,BCT1 03440002
         DROP R2 DROP R2 03450002
         * 03460002
PROSEXIT DS OH EXIT FROM PROSCBS 03470002
         L R14,PRDSSAVE LOAD C(SAVE AREA) INTO R14  03480002
         BR R14     EXIT                        03490002
*********************************************************************** 03500002
* WORKING STORAGE     * 03510002
DS OD     03520002
DC CL36 'CVFLT0T-WORKING STORAGE BEGINS HERE' 03530002
*********************************************************************** 03540002
* EQUATES     * 03550002
R0 EQU 0       03560002
R1 EQU 1       03570002
RDEB EQU 1     REG1 FOR DEB ADDRESS 03580002
R2 EQU 2       03590002
RUCB EQU 2     REG2 FOR UCB ADDRESS 03600002
R3 EQU 3       03610002
R4 EQU 4       03620002
R5 EQU 5       03630002
R6 EQU 6       03640002
R7 EQU 7       03650002
R8 EQU 8       03660002
R9 EQU 9       03670002
R10 EQU 10     03680002
R11 EQU 11     03690002
R12 EQU 12     03700002
R13 EQU 13     03710002
R14 EQU 14     03720002
R15 EQU 15     03730002

Figure C-46 Source of CVFLT0T program (part 7 of 9)
BUFNBR EQU 1           BUFFERS TO BE USED          03740002
C1 EQU 1              SET NBR OF BUFFERS TO 1          03750002

*********************************************************************** 03760002
*                         SAVE AREAS                                  *
SAVE     DC 18F'0'       MAIN PROGRAM SAVE AREA          03770002
IDLSAVE DC F'0'         IDENTIFY VOLUME ROUTINE SAVE AREA     03790002
BUFFSAVE DC F'0'        INITIALIZE BUFFER ROUTINE SAVE AREA   03800002
CALLSAVE DC F'0'        CALL CVSTAT ROUTINE SAVE AREA        03810002
FCLSAVE DC F'0'         INITIALIZE FCL1 ROUTINE SAVE AREA     03820002
TSTRSAVE DC F'0'        TEST RETURN CODE ROUTINE SAVE AREA    03830002
CVRSAVE DC F'0'         CVAFFILT READ 1 DSN ROUTINE SAVE AREA  03840002
PROSAVE DC F'0'         PROCESS DSCBS ROUTINE SAVE AREA       03850002

*********************************************************************** 03860002
*                         CONSTANTS                                   *
CZERO    DC F'0'        CONSTANT - 0              03870002
CONE     DC F'1'        CONSTANT - 1              03880002
CTWO     DC F'2'        CONSTANT - 2              03890002
RCODE00 DC F'0'        RETURN CODE 0          03900002
RCODE04 DC F'4'        RETURN CODE 4          03910002
RCODE08 DC F'8'        RETURN CODE 8          03920002
RCD112 DC F'12'       RETURN CODE 12         03930002
RCD116 DC F'16'        RETURN CODE 16         03940002
SEQ01    DC CL44'MHLEAV.EXTKSDS.DATA'         03950002
CFCLID   DC CL4'FCL '   03960002

*********************************************************************** 03970002
*                    PROGRAM MESSAGES                                 *
BLNKLINE DC CL80' '    03980002
STRTMSG  DC CL80'CVFLT0T START OF OUTPUT MESSAGES'      04000002
VOLTEXT ORG STRTMSG+35 04010002
DC C'VOLUME: '       04020004
VOLSER DC CL6' '      04030004
ENDMSG   DC CL80'CVFLT0T END OF OUTPUT MESSAGES'      04040002
CALLMR11 DC CL80' CVAFFILT CALL: EADSCB=OK LIST/ORDER/F1-F8' 04050002
RC00MSG  DC CL80' RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL ' 04060002
RC04MSG  DC CL80' RC04 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT ' 04070002
RC08MSG  DC CL80' RC08 VERIFIED - INVALID VTOC STRUCTURE ' 04080002
RC012MSG DC CL80' RC12 VERIFIED - CVAFFILT PARM LIST ERROR / KEY ' 04090002
RC016MSG DC CL80' RC16 VERIFIED - I/O ERROR ' 04010002
RCERMSG  DC CL80' UNEXPECTED RETURN CODE RETURNED FROM CVAFFILT ' 04020002
SEQ01M   DC CL80'MHLEAV.EXTKSDS.DATA DSN AND CCHHR RECORD WRITTEN' 04030002
RLSEOKM  DC CL80' VERIFY CVAFFILT ACCESS=RLSE EADSCB=NOTOK IGNORED' 04040002
OKMSG    DC CL80' CV4EADOK BIT SET / EADSCB=OK ' 04050002
NOTOKMSG DC CL80' CV4EADOK BIT IS NOT SET / EADSCB=NOTOK ' 04060002

OUTREC   DS 0CL49      FILTOUT / FILTDD OUTPUT RECORD FORMAT 04070002
ODSN     DS CL44       DSN 04080002
OCCHHR   DS XL5        CCHHR 04090002

*********************************************************************** 04100002
*                         WORK AREAS                                  *
DEBADD DC F'0'        DEB ADDRESS SAVE AREA          04110002
UCBADD DC F'0'        UCB ADDRESS SAVE AREA          04120002
RETCODE DC F'999'     RETURN CODE SAVE AREA          04130002
WFMTREC1 DS CL133' '  WORK FORMAT RECORD FOR OUTPUT 04140002
WCSTCD DC XL1'FF'     CVSTAT CODE WORK AREA          04150002

Figure C-47 Source of CVFLT0T program (part 8 of 9)
Figure C-48  Source of CVFLT0T program (part 9 of 9)
In Figure C-49 on page 546 through to Figure C-59 on page 556, we show assembler source for a program to issue the CVAFDIR macro to get the DSCB for a data set and write it back to the VTOC.

**EAV support considerations**

In this source, the following coding relates to EAV support:

- In routine READ1RTN which starts at line 02720000
  - The CVAFDIR call at line 02770004 shows the specification of EADSCB=OK.

- In routine WRTE1RTN which starts at line 03580000
  - The CVAFDIR call at line 03670004 shows the specification of EADSCB=OK.

- In order to set up the CVAFDSM calls, the VTOC was opened and read directly. In order to allow a VTOC on an EAV volume to be READ, the DCB coded for that purpose had to be extended by the addition of the DCBE macro which is used to specify EADDCB=OK
  - Line 05420000 shows the addition of ,DCBE=VTOCDCBE
  - Line 03960021 shows the addition of VTOCDCBE DCBD EADSCB=OK
CVDIRIE TITLE 'CVAF CVAFDIR TEST MODULE' 00010004
CVDIRIE CSECT 00020004
CVDIRIE AMODE 24 00030004
CVDIRIE RMODE 24 00040004
***********************************************************************
* CVDIRIE - MODULE THAT ISSUES THE CVAFDIR MACRO TO READ DSCBS * 00050000
* FOR A GIVEN DATASET AND THEN WRITES OUT THE BUFFER * 00060000
* WITHOUT UPDATING THE BUFFER. RC 00 STAT000 EXPECTED. * 00070000
* THIS MODULE USES A PASSED DSN CVAFDIR READS (SEARCH). * 00080000
* THE CVAFDIR MACRO CALLS WILL USE THE FOLLOWING: * 00090000
* EADSCB=OK CODED * 00100000
* MULTIPLEDSCBS=YES CODED * 00110000
* MULTIPLEDSCBS=NO CODED * 00120000
* THIS MODULE WILL CREATE AN OUTPUT REPORT THAT LIST * 00130000
* ANY CVSTAT CODES ENCOUNTERED FOR THE CALLS. * 00140000
* OUTPUT IN OUTDD DATASET SHOULD BE THE FOLLOWING: * 00150000
*---------------------------------------------------------------------* 00160000
* EXPECTED OUTPUT * 00170000
* --------------- * 00180000
* CVDIRIE START OF OUTPUT MESSAGES * 00190000
* PROCESSING DSN: MHLEAV.EXTKSDS.DATA (CVAFDIR READ) * 00200000
* CV4EADOK BIT SET / EADSCB=OK * 00210000
* CV4MULTD BIT SET / MULTIPLEDSCBS=YES * 00220000
* RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL * 00230000
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED * 00240000
* PROCESSING DSN: MHLEAV.EXTKSDS.DATA (CVAFDIR WRITE) * 00250000
* CV4EADOK BIT SET / EADSCB=OK * 00260000
* CV4MULTD BIT NOT SET / MULTIPLEDSCBS=NO * 00270000
* RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL * 00280000
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED * 00290000
* VERIFY CVAFDIR ACCESS=RLSE EADSCB=NOTOK IGNORED * 00300000
* RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL * 00310000
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED * 00320000
* CVDIRIE END OF OUTPUT MESSAGES * 00330000
* * 00340000
* * 00350000
* * 00360000
* * 00370000
* * 00380000
* * 00390000
* * 00400000
* * 00410000
* * 00420000
* * 00430000
* * 00440000
* * 00450000
* * 00460000
***********************************************************************

Figure C-49  Source of CVDIR1E program (part 1 of 11)
Figure C-50  Source of CVDIR1E program (part 2 of 11)
Figure C-51  Source of CVDIR1E program (part 3 of 11)
Figure C-52 Source of CVDIR1E program (part 4 of 11)
SETU1RTN DS OH SETUP FOR CVAFDIR CALL
ST R14,SET1SAVE STORE C(R14) INTO SAVE AREA
LA R4,BUFLHDR GET ADDR OF BUF LIST HEADER
USING BFLHDR,R4 GET ADDRESSABILITY TO HEADER
MVI BFLHNOE,X'OC' INDICATE 12 ENTRIES
MVI BFLHKEY,BFLHDSCB INDICATE READ DSCB
LA R6,DSCBBUF LOAD R6 WITH ADDR OF 1ST DSCB BUFFER
LA R7,BUFLIST1 LOAD R7 WITH ADDR OF BUFLIST1
USING BFLE,R7 GET ADDRESSABILITY TO ENTRY

* INITIALIZE 1ST ENTRY

OI BFLEFL,BFLECHR INDICATE CCHHR TO BE READ
MVC BFLEARG(5),CCHHR0 SET ZEROS FOR ARGUMENT
MVI BFLELTH,DSCBL96 GET LENGTH OF BUFFER
ST R6,BFLEBUF PUT DSCB BUFB ADDR IN ENTRY
LA R7,12(,R7) INCREMENT ADDR TO NEXT ENTRY
LA R6,96(,R6) INCREMENT ADDR TO NEXT DSCB BUFFER
SR R5,R5 INIT R5 WITH ZERO
LA R5,1(R5) ADD 1 - ENTRY PROCESSED

SET1EXIT DS OH EXIT FROM SETU1RTN
L R14,SET1SAVE LOAD C(SAVE AREA) INTO R14
BR R14 EXIT

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

SEQ1RTN DS OH PROCESS SEQ01 DATASET ROUTINE
ST R14,SEQ1SAVE STORE C(R14) INTO SAVE AREA
MVC DSNAM(44),SEQ01 DSNNAME=MHLEAV.EXTKSDS.DATA
PUT OUTFILE,SEQ1RMSG WRITE A RECORD TO THE OUTPUT FILE
BAL R14,READ1RTN READ1RTN - EADSCB=OK CODED

* CVAFDIR RELEASE TO RESET CVAF PARAMETER LIST

CVAFDIR ACCESS=RLSE,IXRCDS=NOKEEP,BUFLIST=0,EADSCB=NOTOK,
MF=(E,CVAFDIR)

C R15,RCODE00 IF RC FROM RLSE NOT ZERO T CODE
BE OK6

PUT OUTFILE,RLSERRM WRITE RLSE ERROR MESSAGE

****** Source of CVDIR1E program (part 5 of 11)
Figure C-54  Source of CVDIR1E program (part 6 of 11)
**TSTRCRTN**

- Test return code from CVAFDIR
- Format and print messages as needed
- Invoke CVSTAT module to check CVSTAT code

**ST**

- R14,TSTRSAVE: Store C(R14) into save area
- R15,RETCODE: Load R15 with saved return code

**C**

- R15,RCODE16

**BNL**

- PRCERMSG

**B**

- PROCESS(R15)
- SPACE

**PROCESS**

- EQU *

**SPACE**

**PRC00MSG**

- PUT OUTFILE,RC00MSG: Write RC00 message

**B**

- PUTMSGE

**PRC04MSG**

- PUT OUTFILE,RC04MSG: Write RC04 message

**B**

- PUTMSGE

**PRC08MSG**

- PUT OUTFILE,RC08MSG: Write RC08 message

**B**

- PUTMSGE

**PRC12MSG**

- PUT OUTFILE,RC12MSG: Write RC12 message

**B**

- PUTMSGE

**PRC16MSG**

- PUT OUTFILE,RC16MSG: Write RC16 message

**B**

- PUTMSGE

**PRCERMSG**

- PUT OUTFILE,RCERMSG: Write RC error message

**PUTMSGE**

- DS OH

**LA**

- R1,CVAFDIR: Load R1 with address of CVAFDIR

**USING**

- CVPL,R1: Establish addressability to PLIST

**MVC**

- WCVSTCD(1),CVSTAT: Move CVSTAT code to WCVSTCD

**DROP**

- R1

**BAL**

- R14,CALLCVST: Invoke CALLCVST to call CVSTCHK MOD

**TSTEXIT**

- DS OH

**L**

- R14,TSTRSAVE: Load C(TSTRSAVE) into R14

**BR**

- R14: Exit

Figure C-55 Source of CVDIR1E program (part 7 of 11)
Figure C-56  Source of CVDIR1E program (part 8 of 11)
CALLEXIT DS OH EXIT FROM CALLCVST 04120000
          L R14,CALLSAVE LOAD C(CALLSAVE) INTO R14 04130000
          BR R14 EXIT 04140000
*********************************************************************** 04150000
* WORKING STORAGE * 04160000
          DS OD 04170000
          DC CL36 'CVDIR1E-WORKING STORAGE BEGINS HERE' 04180004
*********************************************************************** 04190000
* EQUATES * 04200000
          R0 EQU 0 04210000
          R1 EQU 1 04220000
          R2 EQU 2 04230000
          R3 EQU 3 04240000
          R4 EQU 4 04250000
          R5 EQU 5 04260000
          R6 EQU 6 04270000
          R7 EQU 7 04280000
          R8 EQU 8 04290000
          R9 EQU 9 04300000
          R10 EQU 10 04310000
          R11 EQU 11 04320000
          R12 EQU 12 04330000
          R13 EQU 13 04340000
          R14 EQU 14 04350000
          R15 EQU 15 04360000
*********************************************************************** 04370000
* SAVE AREAS * 04380000
          SAVE DC 18F '0' MAIN PROGRAM SAVE AREA 04390000
          IDVLSAVE DC F '0' IDENTIFY VOLUME ROUTINE SAVE AREA 04400000
          CALLSAVE DC F '0' CALL CVSTAT ROUTINE SAVE AREA 04410000
          TSTRSAVE DC F '0' TEST RETURN CODE ROUTINE SAVE AREA 04420000
          REAISAVE DC F '0' CVAFDIR READ 1 ROUTINE SAVE AREA 04430000
          SETSAVE DC F '0' SETUP ROUTINE SAVE AREA 04440000
          SEQSAVE DC F '0' SEQ01 ROUTINE SAVE AREA 04450000
          WRTSAVE DC F '0' CVAFDIR WRITE 1 ROUTINE SAVE AREA 04460000
*********************************************************************** 04470000
* PROGRAM FLAGS / SWITCHES * 04480000
          BOTH EQU X '18' 04490000
*********************************************************************** 04500000
* CONSTANTS * 04510000
          CCHHR0 DS XL5 '00' CCHHR ARGUMENT - ZERO 04520000
          NBRENT DC F '12' CONSTANT - 12 NUMBER OF ENTRIES 04530000
          RCODE0 DC F '0' RETURN CODE 0 04540000
          RCODE4 DC F '4' RETURN CODE 4 04550000
          RCODE8 DC F '8' RETURN CODE 8 04560000
          RCODE12 DC F '12' RETURN CODE 12 04570000
          RCODE16 DC F '16' RETURN CODE 16 04580000
          SEQ01 DC CL44 'MHLEAV.EXTKSDS.DATA' 04590000

Figure C-57  Source of CVDIR1E program (part 9 of 11)
*** PROGRAM BUFFERS ***

BUFLIST DS OF
  BUFFER LIST DECLARATIONS

BUFLHDR DC 2F'0'
  BUFFER LIST HEADER

BUFLIST1 DC 3F'0'
  BUFFER LIST ENTRY 1

BUFLIST2 DC 3F'0'
  BUFFER LIST ENTRY 2

BUFLIST3 DC 3F'0'
  BUFFER LIST ENTRY 3

BUFLIST4 DC 3F'0'
  BUFFER LIST ENTRY 4

BUFLIST5 DC 3F'0'
  BUFFER LIST ENTRY 5

BUFLIST6 DC 3F'0'
  BUFFER LIST ENTRY 6

BUFLIST7 DC 3F'0'
  BUFFER LIST ENTRY 7

BUFLIST8 DC 3F'0'
  BUFFER LIST ENTRY 8

BUFLIST9 DC 3F'0'
  BUFFER LIST ENTRY 9

BUFLISTA DC 3F'0'
  BUFFER LIST ENTRY 10

BUFLISTB DC 3F'0'
  BUFFER LIST ENTRY 11

BUFLISTC DC 3F'0'
  BUFFER LIST ENTRY 12

WARD LIST

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

*** PROGRAM MESSAGES ***

DSCBBUF DS XL96
  DSCB BUFFER DECLARATION 1 - 96 BYTE

DSCBBUF2 DS XL140
  DSCB BUFFER DECLARATION 2 - 140 BYTE

DSCBBUF3 DS XL140
  DSCB BUFFER DECLARATION 3 - 140 BYTE

DSCBBUF4 DS XL140
  DSCB BUFFER DECLARATION 4 - 140 BYTE

DSCBBUF5 DS XL140
  DSCB BUFFER DECLARATION 5 - 140 BYTE

DSCBBUF6 DS XL140
  DSCB BUFFER DECLARATION 6 - 140 BYTE

DSCBBUF7 DS XL140
  DSCB BUFFER DECLARATION 7 - 140 BYTE

DSCBBUF8 DS XL140
  DSCB BUFFER DECLARATION 8 - 140 BYTE

DSCBBUF9 DS XL140
  DSCB BUFFER DECLARATION 9 - 140 BYTE

DSCBBUFA DS XL140
  DSCB BUFFER DECLARATION 10- 140 BYTE

DSCBBUF B DS XL140
  DSCB BUFFER DECLARATION 11- 140 BYTE

DSCBBUF C DS XL140
  DSCB BUFFER DECLARATION 12- 140 BYTE

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

RR 04760000

BLNKLINE DC CL80 ' ' 

STRTMSG DC CL80 ' CVDIR1E START OF OUTPUT MESSAGES ' 

VOLTEXT ORG STRTMSG+35 

DC C 'VOLUME: ' 

VOLSER DC CL6 ' ' 

ORG 

ENDMSG DC CL80 ' CVDIR1E END OF OUTPUT MESSAGES ' 

RLSEMSG DC CL80 ' VERIFY CVAFDIR ACCESS=RLSE EADSCB=NOTOK IGNORED ' 

RC00MSG DC CL80 ' RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL ' 

RC04MSG DC CL80 ' RC04 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT ' 

RC08MSG DC CL80 ' RC08 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT '

SEQ1RMSG DS 1CL80 

DC CL37 ' PROCESSING DSN: MHLEAV.EXTKSDS.DATA' 

DC CL41 ' (CVAFDIR READ)' 

SEQ1WMSG DS 1CL80 

DC CL37 ' PROCESSING DSN: MHLEAV.EXTKSDS.DATA' 

DC CL41 ' (CVAFDIR WRITE)' 

RCDOOMSG DC CL80 ' RCDO VERIFIED - THE REQUEST WAS SUCCESSFUL ' 

RCO4MSG DC CL80 ' RCO4 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT ' 

RCO8MSG DC CL80 ' RCO8 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT '

Figure C-58  Source of CVDIR1E program (part 10 of 11)
Figure C-59  Source of CVDIR1E program (part 11 of 11)
C.6.1.1 Execution of sample CVDIR1EJ job
The CVFLT0T and CVDIR1E programs are working examples. If you want to run the CVDIR1EJ job, you need to assemble link the CVFLT0T and CVDIR1E programs. It is not necessary to have in depth assembler experience, but familiarity with JCL is required.

The source is stored in one PDS, the JCL in one PDS, and the LOAD modules in another.

Required once in support of all the CVAFxxxx VTOC management programs, set up the CVSTCHK subroutine. Refer to Figure C.13 on page 636.

To create the assembler PDS, use the following steps.

There are three steps to build each of the program load modules which needs to be done once, after which they can be executed several times.

**Step 1: Create a PDS/PDSE to hold the source members**
(Do this step if you have not already done it.)

In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

**Step 2: Create a PDS/PDSE to hold the LOAD MODULES**
(Do this step if you have not already done it.)

In this example the PDS is called MHLRES1.EAV.LOAD.

The LRECL/RECFM must be 0/U. All other attributes can be chosen by you.

**Step 3: Store the CVFLT0T program source in the PDS**
Make sure that your ISPF EDIT menu settings has the screen format defined as DATA, the emulator line length is set to greater than 80, and that the edit session is in NONUM mode.

- Cut and paste the contents of Figure C-40 on page 536 through to Figure C-48 on page 544 one after the other into member CVFLT0T. The result should contain 477 lines.

**Step 4: Create a PDS/PDSE to hold the JCL members**
(Do this step if you have not already done it.)

In this example the PDS is called MHLRES1.EAV.JCL.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.
Step 5: Store the JCL source to assemble and link CVFLT0T in the PDS

- Cut and paste the contents of Figure C-60 into member CVFLT0T. The result should contain 30 lines.

```jcl
//MHLRES1F JOB (1234567,COMMENT),UAALFO,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*JOBPARM S=* 
//ASMHCL PROC 
//ASM EXEC PGM=ASMA90,REGION=0M, 
 // PARM='OBJECT,NODECK' 
//SYSLIN DD DSN=&OBJ,DISP=(NEW,PASS),UNIT=SYSDA, 
 // SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120 
//SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB 
 // DD DISP=SHR,DSN=SYS1.MODGEN 
 // DD DISP=SHR,DSN=MHLRES1.EAV.ASM 
//SYSPRT DD SYSOUT=* 
//SYSLIB DD DISP=OLD,DELETE 
// DD DDNAME=SYSIN 
//SYSPRT DD SYSOUT=* 
//SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD 
//SYSLIB DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5)) 
// PEND 
 // EXEC ASMHCL 
 //ASM.SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.ASM(CVFLT0T) 
/* 
 //LKED.SYSLMOD DD DSN=MHLRES1.EAV.LOAD,DISP=SHR 
 //LKED.SYSIN DD STATUS=ائي(1) 
 NAME CVFLT0T(R)
```

Figure C-60   JCL to assemble and link CVFLT0T

Step 7: Run the job in member CVFLT0T
Check that the return code from the ASM and LKED steps is 0.

A return code of 8 on the LKED step is an indication that the subroutine required has not been stored in MHLRES1.EAV.LOAD.

Step 8: APF authorize the data set CVFLT0T has been linked into
(Do this step if you have not already done it.)

- Add the data set to the PARMLIB APF member (PROGxx) and IPL
- Or dynamically APF authorize the data set. Assuming it is MHLRES1.EAV.LOAD and that it is on volume MHLSE1, issue the command
  - setprog apf,add,dsname=MHLRES1.EAV.LOAD,volume=MHLSE1

The CVFLT0T program is now ready for execution, as part of job CVDIR1EJ.
Step 9: Store the CVDIR1E program source in the PDS
Make sure that your ISPF EDIT menu settings has the screen format defined as DATA, the emulator line length is set to greater than 80, and that the edit session is in NONNUM mode.

- Cut and paste the contents of Figure C-49 on page 546 through to Figure C-59 on page 556 one after the other into member CVDIR1E. The result should contain 561 lines.

Step 10: Create a PDS/PDSE to hold the JCL members
(Do this step if you have not already done it.)

In this example, the PDS is called MHLRES1.EAV.JCL.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

Step 11: Store the JCL source to assemble and link CVDIR1E in the PDS
- Cut and paste the contents of Figure C-61 into member CVDIR1E. The result should contain 30 lines.

```
//MHLRESIF JOB (1234567,COMMENT),UAALFO,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*/JOBPARM S=* /
//ASMHCL PROC
//ASM EXEC PGM=ASMA90,REGION=0M,
//   PARM='OBJECT,NODECK'
//SYSLIN DD DSN=&OBJ,DISP=(NEW,PASS),UNIT=SYSDA,
//   SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120
//SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB
// DD DISP=SHR,DSN=SYS1.MODGEN
// DD DISP=SHR,DSN=MHLRES1.EAV.ASM
//SYSPRINT DD SYSOUT=* 
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
//
//LKED EXEC PGM=HEWL,REGION=2048K,COND=(8,LE,ASM),
//   PARM='XREF,LIST,LET'
//SYSLIN DD DSN=&OBJ,DISP=(OLD,DELETE)
// DD DDNAME=SYSIN
//SYSPRINT DD SYSOUT=* 
//SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
// PEND
// EXEC ASMHCL
//ASM.SYSIN DD DISP=SHR,DSN=MHLRES1.EAV.ASM(CVFLT1E)
//*/
//LKED.SYSLMOD DD DSN=MHLRES1.EAV.LOAD,DISP=SHR
//LKED.SYSIN DD *
SETCODE AC(1)
NAME CVFLT1E(R)
```

Figure C-61  JCL to assemble and link CVFLT0T
Step 12: Run the job in member CVDIR1E
Check that the return code from the ASM and LKED steps is 0.

A return code of 8 on the LKED step is an indication that the subroutine required has not been stored in MHLRES1.EAV.LOAD.

Step 13: APF authorize the data set CVDIR1E has been linked into
(Do this step if you have not already done it.)

- Add the data set to the PARMLIB APF member (PROGxx) and IPL
- Or dynamically APF authorize the data set. Assuming it is MHLRES1.EAV.LOAD and that it is on volume MHLSE1, issue the command
  - `setprog apf,add,dsname=MHLRES1.EAV.LOAD, volume=MHLSE1`

The CVDIR1E program is now ready for execution as part of job CVDIR1EJ.

Use the JCL as shown in Figure C-35 on page 532.

C.7 VTOC management macros extended for EAV: CVAFDSM

The CVAFDSM macro, with its underlying services is provided to obtain volume information for an indexed or non-indexed VTOC. CVAFDSM itself is not new for EAV support but it has been extended for EAV.

The CVAFDSM macro as implemented in z/OS releases prior to z/OS V1R10, and if not extended by EAV keywords is not permitted to retrieve information from EAV formatted volumes. Use of an un-extended CVAFDSM macro call against a VTOC on an EAV volume will result in a non-zero return code, a non-zero status byte and the data requested will not be returned.

Changes to a program issuing the CVAFDSM macro call are required to allow access to EAV volumes. Existing programs without the EAV enabling operand on the CVAFDSM call will be tracked by the Migration Assistance Tracker but will not fail if run against a non-EAV volume. Such programs will fail if run against an EAV volume.

We have provided an example of a program that issues the CVAFDSM macro in the format used prior to z/OS 1.10, and as extended to support EAV to illustrate the type of change required to support EAV.

The purpose of this example program is to:

- Demonstrate what happens when an enhanced macro call is issued against a non-EAV volume
- Demonstrate what happens when an enhanced macro call is issued against an EAV volume
- Demonstrate what happens when an un-enhanced macro call is issued against a non-EAV volume
- Demonstrate what happens when an un-enhanced macro call is issued against an EAV volume
- Demonstrate that the Migration Assistance Tracker picks up the CVAFDSM un-enhanced macro against a non-EAV volume
- Illustrate the code changes required to support EAV format volumes.
In Figure C-62 we show the JCL to examine volumes MLDC65 and SBOX1T.

```
//MHLRESID JOB (1234567,COMMENT),UAALF0,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*JOBPARM S=* 
//CVDSM10 PROC 
//RUN EXEC PGM=CVDSM1O,REGION=0M
//STEPLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD 
//CVAFDD1 DD DISP=SHR,UNIT=3390,VOL=SER=MLDC65
//SYSUDUMP DD SYSOUT=* 
//OUTDD DD SYSOUT=* 
// PEND
// EXEC CVDSM10
// EXEC CVDSM10
//CVAFDD1 DD DISP=SHR,UNIT=3390,VOL=SER=SBOX1T
```

**Figure C-62**  Example of JOB to examine two volumes using CVDSM1O

In Figure C-63 we show the result of running the JCL to execute program CVDSM1O on two volumes MLDC65 (EAV) and SBOX1T (non-EAV).

```
CVDSM10 START OF OUTPUT MESSAGES VOLUME: MLDC65
CVAFDSM CALL: EADSCB=NOTOK IS THE DEFAULT
CV4EADOK BIT IS NOT SET / EADSCB=NOTOK
RC04 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT
CVSTAT CODE: X"52"    DEC"082"
CVAFDSM CALL: EADSCB=OK KEYWORD CODED
CV4EADOK BIT SET / EADSCB=OK
RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
CVSTAT CODE: X"00"    DEC"000"
CVDSM10 END OF OUTPUT MESSAGES

CVDSM10 START OF OUTPUT MESSAGES VOLUME: SBOX1T
CVAFDSM CALL: EADSCB=NOTOK IS THE DEFAULT
CV4EADOK BIT IS NOT SET / EADSCB=NOTOK
RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
CVSTAT CODE: X"00"    DEC"000"
CVAFDSM CALL: EADSCB=OK KEYWORD CODED
CV4EADOK BIT SET / EADSCB=OK
RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
CVSTAT CODE: X"00"    DEC"000"
CVDSM10 END OF OUTPUT MESSAGES
```

**Figure C-63**  CVDSM1O output from sample job
This output shows the following results:

- The first set of messages relate to volume MLDC65 (EAV).
  - CVAFDSM CALL: EADSCB=NOTOK IS THE DEFAULT
    • Result is FAILURE with CVSTAT X"52" (DEC(82))
  - CVAFDSM CALL: EADSCB=OK KEYWORD CODED
    • Result is SUCCESS with CVSTAT X"00" (DEC(00))

- The second set of messages relate to volume SBOX1T (non-EAV).
  - CVAFDSM CALL: EADSCB=NOTOK IS THE DEFAULT
    • Result is SUCCESS with CVSTAT X"00" (DEC(00))
  - CVAFDSM CALL: EADSCB=OK KEYWORD CODED
    • Result is SUCCESS with CVSTAT X"00" (DEC(00))

**C.7.1 EAV Migration Assistance Tracker considerations: CVAFDSM**

The EAV Migration Assistance Tracker will detect use of the CVAFDSM service that has not been updated for EAV compatibility.

Any appearance on the tracker list must be investigated and resolved because use of an un-extended OBTAIN against an EAV volume in due course will fail.

For information about the EAV Migration Tracker, refer to Appendix C of *DFSMS Advanced Services*, SC26-7400 and in this document to “Migration Assistance Tracker” on page 641.

In Figure C-64 we show the output from the Migration Assistance Tracker that relates to the run of CVDSM1O run against volume SBOX1T. It shows that CVDSM1O needs attention before it is used against an EAV format volume.

![Figure C-64](image-url)

**Source code listing**

In Appendix C-65, “Source of CVDSM1O program (1 of 8)” on page 563 through to Appendix C-72, “Source of CVDSM1O program (8 of 8)” on page 570 we show assembler source for a program to issue the CVAFDSM macro.

In this source, the following coding relates to EAV support:

- In routine CVAFDSM2 which starts at line 02700021
  - the CVAFDSM call which starts at line X02820021 shows the addition of EADSCB=OK

- In routine CVAFDSM1 which starts at line 02410021
  - the CVAFDSM call which starts at line 02470021 there is no EADSCB option coded so the default EADSCB=NO is taken as the default.

- In order to set up the CVAFDSM calls, the VTOC was opened and read directly. In order to allow a VTOC on an EAV volume to be READ, the DCB coded for that purpose had to be extended by the addition of the DCBE macro which is used to specify EADDCB=OK
  - Line 03870021 shows the addition of ,DCBE=VTOCE1
  - Line 03960021 shows the addition of VTOCE1 DCBD EADSCB=OK
CVDSM10 TITLE 'CVAF CVAFDSM TEST MODULE'

CVDSM10 TITLE 'CVAF CVAFDSM TEST MODULE'

CVDSM10 CSECT 00010021

CVDSM10 AMODE 24 00020021

CVDSM10 RMODE 24 00030021

* 00040021

* 00050021

*********************************************************************** 00060021

*                                                                     *

*    CVDSM1O - MODULE THAT ISSUES THE CVAFDSM MACRO TO VERIFY THE      *
*               PROPER INFORMATION IS CREATED BY THE DFSMS MIGRATION  *
*               ASSISTANCE CODE.                                      *
*                                                                     *
*              THE CVAFDSM MACRO CALL WILL USE THE FOLLOWING:          *
*              ACCESS=MAPDATA,MAP=VOLUME,RTA4BYTE=YES,EXTENTS=ADDR    *
*              EADSCB=NOTOK AS THE DEFAULT FOR A NON EAV VOLUME.      *
*                                                                     *
*              THE CVAFDSM MACRO CALL WILL USE THE FOLLOWING:          *
*              ACCESS=MAPDATA,MAP=VOLUME,RTA4BYTE=YES,EXTENTS=ADDR    *
*              EADSCB=OK CODED FOR AN EAV VOLUME.                     *
*                                                                     *
*              THIS PROGRAM WILL CREATE AN OUTPUT REPORT THAT         *
*              WILL VERIFY THE RETURN CODE AND CVSTAT CODE FOR THE    *
*              CVAFDSM CALL.  THE EXPECTED REPORT OUTPUT IS           *
*              DETAILED BELOW.                                       *
*                                                                     *
*              THIS PROGRAM IS WRITTEN FOR BOTH NON EAV AND EAV      *
*              VOLUMES AND THE RETURN CODES AND CVSTAT CODES         *
*              WILL DIFFER DEPENDING ON THE VOLUME TYPE.             *
*                                                                     *
*   OUTPUT IN OUTDD DATASET SHOULD BE THE FOLLOWING:                 *
*---------------------------------------------------------------------* 00190021

*                                                                     *

*   NON EAV VOLUME                                                   *
*   ---------------------------------------------------------------*

* CVDSM1O START OF OUTPUT MESSAGES VOLUME: VVVVVV                    *
* CVAFDSM CALL: EADSCB=NOTOK IS THE DEFAULT                           *
* CV4EADOK BIT IS NOT SET / EADSCB=NOTOK                              *
* RCOO VERIFIED - THE REQUEST WAS SUCCESSFUL                          *
* CVSTAT CODE: X"00" DEC"000"                                        *
*                                                                     *
* CVDSM1O END OF OUTPUT MESSAGES                                    *
*                                                                     *
* EAV VOLUME                                                         *
*   ---------------------------------------------------------------*

* CVDSM1O START OF OUTPUT MESSAGES VOLUME: VVVVVV                    *
* CVAFDSM CALL: EADSCB=OK KEYWORD CODED                               *
* CV4EADOK BIT SET / EADSCB=OK                                       *
* RCOO VERIFIED - THE REQUEST WAS SUCCESSFUL                          *
* CVSTAT CODE: X"00" DEC"000"                                        *
*                                                                     *
* CVDSM1O END OF OUTPUT MESSAGES                                    *
*                                                                     *
* Appendix C. Code samples DFSMS V1.10  563

Figure C-65  Source of CVDSM10 program (1 of 8)
**CVDSM10 - LOGIC NOTES**

**INITIALIZATION**
- Obtain the necessary information from the DASD volume
- Open an output file and write the necessary output messages

**MAINLINE**
- Invoke INTEXTBL routine to initialize extent table
- Invoke CVAFDSM1 routine - EADSCB keyword not coded (non EAV)
- Report on return code and CVSTAT code returned from call
- Report on return code and CVSTAT code returned from call

**FINALIZATION**
- Write necessary output messages and close the output file
- Exit

**CVDSM10 - JOB INFORMATION**

**NORMAL END OF JOB:**
- RC=00 and OUTDD output as detailed above

**ABNORMAL END OF JOB:**
- ABEND 100 - Error opening VTOC on the DASD volume that is associated with the CVAFDD1 DD statement
- ABEND 101 - Error opening the OUTDD dataset
- ABEND 102 - Error closing the OUTDD dataset

**HOUSEKEEPING**
- Save caller's registers and establish a new register save area

```
STD R14,R12,12(R13)     STANDARD LINKAGE CONVENTION 00930021
BALR R10,0              R10 IS IMPLIED BASE REG 00940021
USING BASE,R10,R11,R12  R11 AND R12 ALSO BASE REGS 00950021
BASE L R11,BASEA01      SET UP ADDRESSING FOR R11 00960021
L R12,BASEA02          SET UP ADDRESSING FOR R12 00970021
B CV0000000 BRANCH AROUND DECLARES 00980021
BASEA01 DC A(BASE+4096) ADDRESSING FOR BASE R11 00990021
BASEA02 DC A(BASE+8192) ADDRESSING FOR BASE R12 01000021
CV000000 DS OH        HOUSEKEEPING... 01010021
ST R13,SAVE+4          SAVE PTR TO CALLER'S SAVE AREA 01020021
LA R14,SAVE           GET ADDRESS OF THE NEW SAVE AREA 01030021
ST R14,(R13)          CHAIN CALLER'S AREA TO OURS 01040021
LR R13,R14            ESTABLISH THE NEW SAVE AREA 01050021
```

Figure C-66 Source of CVDSM1O program (2 of 8)
**Appendix C. Code samples DFSMS V1.10**

Figure C-67  Source of CVDSM1O program (3 of 8)

```
***********************************************************************
* INITIALIZATION                                                      *
INITIAL DS OH  INITIALIZATION SECTION
  BAL R14,IDVLIRTN  INVOKE RTN TO IDENTIFY VOL (CVAFDD1)
  OPEN (OUTFILE,(OUTPUT))  OPEN THE OUTPUT MESSAGE FILE
  TM OUTFILE+48,X'10'  IF OPEN OF OUTPUT FILE NOT OK
  BO OK1
    ABEND 101  ISSUE USER ABEND 101
OK1 DS OH
  L RUCB,UCBADD1
  MVC VOLSER(6),28(RUCB)
  PUT OUTFILE,STRMTMSG  WRITE A RECORD TO THE OUTPUT FILE
*  INITIALIZE BUFFER LIST
***********************************************************************
* MAINLINE                                                            *
MAINLINE DS OH  MAINLINE SECTION
*  PUT OUTFILE,BLNKLINE  WRITE A RECORD TO THE OUTPUT FILE
  BAL R14,INTEXTBL  INVOKE ROUTINE TO INIT EXTENT TABLE
  BAL R14,CVAFDSM1  INVOKE CVAFDSM CALL1 ROUTINE-non-EAV
  BAL R14,CVAFDSM2  INVOKE CVAFDSM CALL2 ROUTINE-EAV
*
***********************************************************************
* FINALIZATION                                                        *
FINAL DS OH  FINALIZATION SECTION
  PUT OUTFILE,BLNKLINE  WRITE A RECORD TO THE OUTPUT FILE
  PUT OUTFILE,ENDMSG  WRITE A RECORD TO THE OUTPUT FILE
  CLOSE (OUTFILE)  CLOSE OUTPUT FILE
  C R15,RCODE00  IF FILE CLOSE IS NOT OK
    BE OK2
  ABEND 102  ISSUE USER ABEND 102
OK2 DS OH
  L R13,4(R13)  RESTORE REGISTER
  LM R14,R12,12(R13)  RESTORE CALLERS REGISTERS
  LA R15,0  SET RC TO 0
  BR R14  RETURN TO CALLER
*
***********************************************************************
* IDVLIRTN                                                            *
*  - OBTAIN THE NECESSARY INFORMATION FROM DASD VOLUME - CVAFDD1 *
IDVLIRTN DS OH  IDENTIFY VOLUME ROUTINE
  ST R14,IDVISAVE  STORE C(R14) INTO SAVE AREA
  RDJFCB (VTODCDB1,(INPUT))  READ JFCB / OPEN VTOC
  MVI JFCB1,X'04'  PUT IN ID FOR FORMAT 4
  MVC JFCB1+1(43),JFCB1  SETUP FOR VTOC OPEN
  OPEN (VTODCDB1,(INPUT)),TYPE=J  OPEN VTOC (OPEN TYPE=J)
    TM VTODCDB1+48,X'10'  IF OPEN OF VTOC NOT OK
    BO OK3
    ABEND 100  ISSUE USER ABEND 100
OK3 DS OH
  SLR RDEB,RDEB  INIT REG1 FOR DEB PTR
  SLR RUCB,RUCB  INIT REG2 FOR UCB PTR
  ICM RDEB,B'0111',VTODCDB1+45  GET DEB ADDRESS
  ST RDEB,DEBADD1  SAVE DEB ADDRESS INTO SAVE AREA
  ICM RUCB,B'0111',33(RDEB)  GET UCB ADDRESS
  ST RUCB,UCBADD1  STORE UCB ADDRESS INTO SAVE AREA
```

Figure C-67  Source of CVDSM1O program (3 of 8)
DFSMS V1.10 and EAV Technical Guide

Figure C-68  Source of CVDSM1O program (4 of 8)

IDVIEXIT DS  OH          EXIT FROM IDVL1RTN  01620021
L  R14,IDV1SAVE       LOAD C(SAVE AREA) INTO R14  01630021
BR R14                EXIT                       01640021
*********************************************************************** 01650021
*                        TSTRCRTN                               * 01660021
*                      - TEST RETURN CODE FROM MACRO CALL      * 01670021
*                      - FORMAT AND PRINT MESSAGES AS NEEDED    * 01680021
*                      - INVOKE CVSTAT MODULE TO CHECK CVSTAT CODE * 01690021
TSTRCRTN DS  OH       CHECK RETURN CODE ROUTINE  01700021
ST R14,TSTRSAVE     STORE C(R14) INTO SAVE AREA  01710021
L  R15,RETCODE       LOAD R15 WITH SAVED RETURN CODE  01720021
C  R15,RCODE16       01730021
BNL PRCERMSG        01740021
B  PROCESS(R15)      01750021
SPACE               01760021
PROCESS EQU * 01770021
B  PRC00MSG         0    RC MESSAGE 01780021
B  PRC04MSG         4    RC MESSAGE 01790021
B  PRC08MSG         8    RC MESSAGE 01800021
B  PRC12MSG        12    RC MESSAGE 01810021
B  PRC16MSG        16    RC MESSAGE 01820021
SPACE               01830021
PRC00MSG  PUT OUTFILE,RC00MSG     WRITE RC00 MESSAGE 01840021
B  PUTMSGE          01850021
PRC04MSG  PUT OUTFILE,RC04MSG     WRITE RC04 MESSAGE 01860021
B  PUTMSGE          01870021
PRC08MSG  PUT OUTFILE,RC08MSG     WRITE RC08 MESSAGE 01880021
B  PUTMSGE          01890021
PRC12MSG  PUT OUTFILE,RC12MSG     WRITE RC12 MESSAGE 01900021
B  PUTMSGE          01910021
PRC16MSG  PUT OUTFILE,RC16MSG     WRITE RC16 MESSAGE 01920021
B  PUTMSGE          01930021
PRCERMSG  PUT OUTFILE,RCERMSG     WRITE RC ERROR MESSAGE 01940021
PUTMSGE DS  OH       01950021
MVC WCSTSTCD(1),CVSTAT  MOVE CVSTAT CODE TO WCSTSTCD  01960021
*   DC 'H'0' ABEND   01970021
BAL R14,CALLCVST    INVOKE CALLCVST TO CALL CVSTCHK MOD  01980021
TSTREXIT DS  OH       EXIT FROM TSTRCRTN  01990021
L  R14,TSTRSAVE     LOAD C(TSTRSAVE) INTO R14  02000021
BR R14                EXIT                       02010021
*********************************************************************** 02020021
*                         CALLCVST                               * 02030021
*                      - ROUTINE TO CALL CVSTCHK FOR BINARY SEARCH OF CVSTAT TABLE   * 02040021
CALLCVST DS  OH       CALL TO CVSTCHK ROUTINE  02050021
ST R14,CALLSAVE     STORE C(R14) INTO SAVE AREA  02060021
LA R1,PARADDR1       LOAD R1 WITH PARMLIST FOR CALL  02070021
L  R15,+V(CVSTCHK)   SET UP CALL TO CVSTCHK  02080021
BALR R14,R15        INVOKE CVSTCHK EXTERNAL SUBPROGRAM  02090021
PUT OUTFILE,WFMTREC1 WRITE A RECORD TO THE OUTPUT FILE  02100021
CALLEXIT DS  OH       EXIT FROM CALLCVST  02110021
L  R14,CALLSAVE     LOAD C(CALLSAVE) INTO R14  02120021
BR R14                EXIT                       02130021
*********************************************************************** 02140021
*                         INTEXTBL                                * 02150021
*                      - ROUTINE TO INITIALIZE THE EXTENT TABLE FOR CVAF MACRO CALL * 02160021

INTEXTBL DS OH INITIALIZE EXTENT TABLE 02170021
ST R14,INEXSAVE STORE C(R14) INTO SAVE AREA 02180021
LA R5,EXTENTS SET UP ADDRESSABILITY TO EXTENT TABLE 02190021
USING DT2EDT02,R5 MAP THE TABLE USING ICVEDT02 02200021
XC EXTENTS(E44),EXTENTS CLEAR EXTENT TABLE 02210021
L R6,C44 LOAD R6 WITH LENGTH - 44 02220021
ST R6,DT2X7LEN STORE EXTENT NBR INTO DT2X7ENT 02230021
MVC DT2X7FLG,DT2X7FLG CLEAR DT2X7FLG 02240021
L R6,C1 LOAD R6 WITH NBR OF EXTENTS - 1 02250021
ST R6,DT2X7ENT STORE EXTENT NBR INTO DT2X7ENT 02260021
MVI DT2X7LEV,X'01' MOVE 1 INTO DT2X7LEV 02270021
XC DT2X7FLG,DT2X7FLG CLEAR DT2X7FLG 02280021
XC DT2X7FLG,DT2X7FLG CLEAR DT2X7FLG 02290021
XC DT2X7CSR,DT2X7CSR CLEAR DT2X7CSR 02300021
XC DT2X7RTA,DT2X7RTA CLEAR DT2X7RTA 02310021
LA R6,DT2ENTRY SET UP ADDRESSABILITY TO EXTENT ENTRY 02320021
USING DT2RTAST,R6 ADDRESS DT2RTAST (RTA START) 02330021
XC DT2RTAST,DT2RTAST CLEAR DT2RTAST 02340021
DROP R5,R6 DROP R5 AND R6 02350021
INEXEXIT DS OH EXIT FROM ROUTINE 02360021
L R14,INEXSAVE LOAD C(SAVE AREA) INTO R14 02370021
BR R14 EXIT 02380021
*********************************************************************** 02390021
* - ISSUE CVAFDSM MACRO CALL WITH EADSCB=NOTOK AS THE DEFAULT 02400021
CVAFDSM1 DS OH CVAFDSM CALL ROUTINE NO EADSCB CODED 02410021
ST R14,CVD1SAVE STORE C(R14) INTO SAVE AREA 02420021
PUT OUTFILE,CALLM1 WRITE A RECORD TO THE OUTPUT FILE 02430021
LA R7,CVPLIST LOAD R7 WITH ADDR OF CVAF PARM LIST 02440021
L RUCB,UCBADD1 LOAD RUCB WITH UCB ADDRESS - CVAFDD1 02450021
* CVAFDSM ACCESS=MAPDATA, CVAFDSM CALL - MAPDATA 02460021
MAP=VOLUME, MAP THE VOLUME 02470021
UCB=(RUCB), PROVIDE THE UCB 02480021
RTA4BYTE=YES, WE WANT RTA'S RETURNED 02490021
EXTENTS=EXTENTS, PROVIDE ADDR OF EXTENT TABLE 02500021
COUNT=NO, NO COUNT FOR THIS CALL 02510021
MF=(E,(R7)) EXECUTE 02520021
* 02530021
ST R15,RETCODE STORE RC INTO RETCODE 02540021
TM CVFL4,CV4EADOK SELECT BASED ON FLAG4 02550021
BNO NOTOK1 EXECUTE 02560021
PUT OUTFILE,OKMSG WRITE OK MSG RECORD 02570021
B OK4 02580021
NOTOK1 PUT OUTFILE,NOTOKMSG WRITE NOTOK MSG RECORD 02590021
OK4 DS OH 02600021
* 02610021
BAL R14,TSTRCRTN INVOKE ROUTINE TO CHECK RC/STAT 02620021
* 02630021
CVDIEXIT DS OH EXIT FROM ROUTINE 02640021
L R14,CVD1SAVE LOAD C(SAVE AREA) INTO R14 02650021
BR R14 EXIT 02660021
*********************************************************************** 02670021
* - ISSUE CVAFDSM MACRO CALL WITH EADSCB=OK CODED 02680021
* 02690021

Figure C-69  Source of CVDSM1O program (5 of 8)
CVAFDSM2 DS OH CVAFDSM CALL ROUTINE EADSCB=NOTOK 02700021
ST R14,CVD2SAVE STORE C(R14) INTO SAVE AREA 02710021
PUT OUTFILE,CALLM2 WRITE A RECORD TO THE OUTPUT FILE 02720021
LA R7,CVPLIST LOAD R7 WITH ADDR OF CVAF PARM LIST 02730021
* RUCB,UCBAADD1 LOAD RUCB WITH UCB ADDRESS - CVAFDD1 02740021
CVAFDSM ACCESS=MAPDATA, CVAFDSM CALL - MAPDATA X02750021
MAP=VOLUME, MAP THE VOLUME X02760021
UCB=(RUCB), PROVIDE THE UCB X02770021
RTA4BYTE=YES, WE WANT RTA'S RETURNED X02780021
EXTENTS=EXTENTS, PROVIDE ADDR OF EXTENT TABLE X02790021
COUNT=NO, NO COUNT FOR THIS CALL X02800021
EADSCB=OK, EADSCB=OK IS CODED FOR THIS CALL X02810021
MF=(E,(R7)) EXECUTE 02820021
* ST R15,RETCODE STORE RC INTO RETCODE 02830021
TM CVFL4,CV4EADOK SELECT BASED ON FLAG4 02840021
BNO NOTOK2 PROVIDE THE UCB 02850021
PUT OUTFILE,OKMSG WRITE OK MSG RECORD 02860021
B OK5 MF=(E,(R7)) 02870021
NOTOK2 PUT OUTFILE,NOTOKMSG WRITE NOTOK MSG RECORD 02880021
OK5 DS OH 02890021
BAL R14,TSTRCRTN INVOKE ROUTINE TO CHECK RC/STAT 02900021
* CVD2EXIT DS OH EXIT FROM ROUTINE 02910021
L R14,CVD2SAVE LOAD C(SAVE AREA) INTO R14 02920021
BR R14 EXIT 02930021
*********************************************************************** 02940021
* WORKING STORAGE 02950021
* DS 0D 02960021
DC CL36'CVDSM1O-WORKING STORAGE BEGINS HERE' 02970021
*********************************************************************** 02980021
* EQUATES 02990021
R0 EQU 0 03000021
R1 EQU 1 03010021
RDEB EQU 1 REG1 FOR DEB ADDRESS 03020021
R2 EQU 2 REG2 FOR UCB ADDRESS 03030021
RUCB EQU 2 REG2 FOR UCB ADDRESS 03040021
R3 EQU 3 03050021
R4 EQU 4 03060021
R5 EQU 5 03070021
R6 EQU 6 03080021
R7 EQU 7 03090021
R8 EQU 8 03100021
R9 EQU 9 03110021
R10 EQU 10 03120021
R11 EQU 11 03130021
R12 EQU 12 03140021
R13 EQU 13 03150021
R14 EQU 14 03160021
R15 EQU 15 03170021
E44 EQU 44 EQUATE FOR DT2X7ENT 03180021
Figure C-70  Source of CVDSM1O program (6 of 8)
Appendix C. Code samples DFSMS V1.10

Figure C-71  Source of CVDSM10 program (7 of 8)

***********************************************************************
*                        SAVE AREAS                                  *
SAVE     DC    18F'0'             MAIN PROGRAM SAVE AREA            
IDVSAVE DC    F'0'               IDENTIFY VOLUME ROUTINE1 SAVE AREA   
CALLSAVE DC    F'0'               CALL CVSTAT ROUTINE SAVE AREA     
TSTRSAVE DC    F'0'               TEST RETURN CODE ROUTINE SAVE AREA  
INEXSAVE DC    F'0'               INIT EXTENT TABLE ROUTINE SAVE AREA  
CVDSAVE DC    F'0'               CVAFDSM CALL1 ROUTINE SAVE AREA     
CVD2SAVE DC    F'0'               CVAFDSM CALL2 ROUTINE SAVE AREA     
***********************************************************************
*                        CONSTANTS                                   *
RCODE00  DC    F'0'               RETURN CODE 0                     
RCODE04  DC    F'4'               RETURN CODE 4                     
RCODE08  DC    F'8'               RETURN CODE 8                     
RCODE12  DC    F'12'              RETURN CODE 12                    
RCODE16  DC    F'16'              RETURN CODE 16                    
C1       DC    F'1'               CONSTANT - 1 TABLE EXTENT ENTRIES   
C44      DC    F'44'              CONSTANT - 44 EXTENT TABLE LENGTH   
DT2EYECA DC    CL8'ICVEDT02'      EYECATCHER FOR ICVEDT02            
***********************************************************************
*                   PROGRAM MESSAGES                                 *
BLNKLINE DC    CL133' '            
STRTMSG  DC    CL133'CVDSM10 START OF OUTPUT MESSAGES                 
VOLTEXT ORG   STRTMSG+33                                                 
DC    C'VOLUME: '                                                     
VOLSER DC    CL6' '                                                     
ENDMSG   DC    CL133'CVDSM10 END OF OUTPUT MESSAGES                     
CALLM1 DC    CL133' CVAFDSM CALL: EADSCB=NOTOK IS THE DEFAULT           
CALLM2 DC    CL133' CVAFDSM CALL: EADSCB=OK KEYWORD CODED               
RC00MSG  DC    CL133' RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL        
RC04MSG  DC    CL133' RC04 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT     
RC08MSG  DC    CL133' RC08 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT     
RC12MSG  DC    CL133' RC12 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT     
RC16MSG  DC    CL133' RC16 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT     
RCERMSG  DC    CL133' ERROR: UNEXPECTED RETURN CODE ENCOUNTERED          
OKMSG    DC    CL133' CV4EADOK BIT SET / EADSCB=OK                      
NOTOKMSG DC    CL133' CV4EADOK BIT IS NOT SET / EADSCB=NOTOK             
***********************************************************************
*                         TABLES                                      *
EXTNTTAB DS    0D                 EXTENT TABLE                      
EXTNTNO  DC    AL1(10)            NUMBER OF EXTENTS                   
EXTNTS DS    1CL5               1 EXTENTS - 5 BYTES LONG                
EXTENTS DS    44X                STORAGE                             
***********************************************************************
*                        WORK AREAS                                  *
DEBADD1  DC    F'0'               DEB ADDRESS SAVE AREA1                
UCBADD1  DC    F'0'               UCB ADDRESS SAVE AREA1                
RECADD1  DC    F'999'             RETURN CODE SAVE AREA                  
WFMREC1 DS    CL133' '            WORK FORMAT RECORD FOR OUTPUT          
CVSTCD DC    XLI'FF'             CVSTAT CODE WORK AREA                  
***********************************************************************
*                PARAMETER LIST FOR EXTERNAL SUBROUTINE               *
DEBADD1 DC    F'0'               DEB ADDRESS SAVE AREA1                
UCBADD1 DC    F'0'               UCB ADDRESS SAVE AREA1                
RECADD1 DC    F'999'             RETURN CODE SAVE AREA                  
WFMREC1 DS    CL133' '            WORK FORMAT RECORD FOR OUTPUT          
CVSTCD DC    XLI'FF'             CVSTAT CODE WORK AREA                  
***********************************************************************
Figure C-71  Source of CVDSM10 program (7 of 8)
C.7.1.1 Execution of sample CVDSM1O program

The CVDSM1O program is a working example. If you want to run the program, you need to assemble link it. It is not necessary to have in depth assembler experience, but familiarity with JCL is required.

The source is stored in one PDS, the JCL in one PDS, and the LOAD modules in another.

Required once in support of all the CVAFxxxx VTOC management programs, set up the CVSTCHK subroutine. Refer to Figure C.13 on page 636.

To create the assembler PDS, use the following steps.

There are three steps to build the program which needs to be done once, after which it can be executed several times.

**Step 1: Create a PDS/PDSE to hold the source members**

In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.
Step 2: Create a PDS/PDSE to hold the LOAD MODULES
In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 0/U. All other attributes can be chosen by you.

Step 3: Store the program source in the PDS
Make sure that your ISPF EDIT menu settings has the screen format defined as DATA, the emulator line length is set to greater than 80, and that the edit session is in NONUM mode.

- Cut and paste the contents of Figure C-64 on page 562 through to Figure C-72 on page 570 one after the other into member CVDSM10. The result should contain 407 lines.

Step 4: Create a PDS/PDSE to hold the JCL members
In this example the PDS is called MHLRES1.EAV.JCL.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

Step 5: Store the JCL source in the PDS
- Cut and paste the contents of Figure C-73 on page 571 into member CVDSM10. The result should contain 29 lines.

```
//MHLRESID JOB (1234567,COMMENT),UAALFO,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*JOBPARM S=* */
//ASMHCL PROC
//ASM EXEC PGM=ASMA90,REGION=0M,
    // PARM='OBJECT,NODECK'
//SYSLIN DD DSN=&OBJ,DISP=(NEW,PASS),UNIT=SYSDA,
    // SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120
//SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB
// DD DISP=SHR,DSN=SYS1.MODGEN
// DD DISP=SHR,DSN=MHLRES1.EAV.ASM
//SYSPRINT DD SYSOUT=* 
//SYSTUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
/*
//LKED EXEC PGM=HEWL,REGION=2048K,COND=(8,LE,ASM),
    // PARM='XREF,LST,LET'
//SYSLIN DD DSN=&OBJ,DISP=(OLD,DELETE)
// DD DDNAME=SYSIN
//SYSPRINT DD SYSOUT=* 
//SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//SYSTUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
// PEND
// EXEC ASMHCL
//ASM.SYSIN DD DISP=SHR,DSN=MHLRES1.EAV.ASM(CVDSM10)
/*
//LKED.SYSLMOD DD DSN=MHLRES1.EAV.LOAD,DISP=SHR
//LKED.SY SIN DD *
SETCODE AC(1)
NAME CVDSM10(R)
Figure C-73 JCL to assemble and link CVDSM10
```
Step 6: Run the job in member CVSEQ8O
Check that the return code from the ASM and LKED steps is 0.

A return code of 8 on the LKED step is an indication that the subroutine required has not been stored in MHLRES1.EAV.LOAD.

Step 7: APF authorize the data set CVSEQ8O has been linked into
- Add the data set to the PARMLIB APF member (PROGxx) and IPL
- Or dynamically APF authorize the data set. Assuming it is MHLRES1.EAV.LOAD and that it is on volume MHLSE1, issue the command
  - `setprog apf,add,dsnname=MHLRES1.EAV.LOAD,volname=MHLSE1`

The CVDSM10 program is now ready for execution. Use the JCL as shown in Figure C-62 on page 561

C.8 VTOC management macros extended for EAV - CVAFFILT

The CVAFFILT macro, with its underlying services is provided to obtain information about the data sets on an indexed or non-indexed VTOC. CVAFFILT itself is not new for EAV support but it has been extended for EAV.

The CVAFFILT macro as implemented in z/OS releases prior to z/OS V1R10, and if not extended by EAV keywords is not permitted to retrieve information from EAV formatted volumes. Use of an un-extended CVAFFILT macro call against a VTOC on an EAV volume will result in a non-zero return code, a non-zero status byte and the data requested will not be returned.

Changes to a program issuing the CVAFFILT macro call are required to allow access to EAV volumes. Existing programs without the EAV enabling operand on the CVAFFILT call will be tracked by the Migration Assistance Tracker but will not fail if run against a non-EAV volume. Such programs will fail if run against an EAV volume.

The example program provided is based on the CVAFFEXP example provided in DFSMS Advanced Services, SC26-7400, expanded to support EAV volumes and recognize Format=8 and Format-9 DSCBS. It issues the CVAFFILT macro in the format required to support EAV, and we document the type of change required to support EAV.

The purpose of this example program is to:
- Demonstrate what happens when CVAFFILT requests are issued to return DSCBs for specific data sets
- Demonstrate what happens when CVAFFILT requests are issued to return DSCBs for a whole volume
- Demonstrate that the Migration Assistance Tracker picks up the CVAFFILT un-announced macro against a non-EAV volume
- Illustrate the code changes required to support EAV format volumes.
In Figure C-74 we show the JCL to examine volumes MLDC65, MLHS1A, and MLDC66.

```
//MHLRES1F JOB (1234567,COMMENT),UAALF0,TIME=10,
  // MSGLEVEL=1,CLASS=A,
  // NOTIFY=&SYSUID
/*JOBPARM S=* */
//CVFLT1E PROC
//RUN      EXEC PGM=CVFLT1E,REGION=0M
//STEPLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//CVAFDD  DD DISP=SHR,UNIT=SYSALLDA, VOL=SER=&VOLSER
//SYSUDUMP DD SYSOUT=* 
//OUTDD    DD SYSOUT=* 
//PEND
//VMLDC65  EXEC CVFLT1E,VOLSER=MLDC65
//VMHLS1A  EXEC CVFLT1E,VOLSER=MLHS1A
//VMLDC66  EXEC CVFLT1E,VOLSER=MLDC66
```

**Figure C-74  Example of JOB to examine three volumes using CVFLT1E**

In Appendix C-75, "CVFLT1E output from sample job - volume MLDC65 (1 of 3)" on page 574, Appendix C-77, "CVFLT1E output from sample job - Volume MHLS1A (2 of 3)" on page 576 and Appendix C-79, "CVFLT1E output from sample job - Volume MHLC66 (3 of 3)" on page 578 we show the result of running the JCL to execute program CVFLT1E on three volumes MLDC65 (EAV) and MHLS1A (non-EAV) and MLDC66 (EAV).
CVFLT1E START OF OUTPUT MESSAGES VOLUME: MLDC65

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN:   MHLRES1.EAV.PDS01
AND FOR DSN:                                     MHLAV.EXTKSDS.DATA
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000001
NUMBER OF FORMAT 9 DSCBS - 0000001

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
NO DSCBS RETURNED FROM CVAFFILT
CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN:   MHLRES1.EAV.PDSE01
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
NO DSCBS RETURNED FROM CVAFFILT
CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN:   MHLRES1.EAV.PDS01
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000

RC04 VERIFIED - CVSTAT 064 RESUME IS NECESSARY
CVAFFILT (INITIAL) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
NUMBER OF FORMAT 1 DSCBS - 0000002
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000004
NUMBER OF FORMAT 9 DSCBS - 0000004
CVAFFILT (RESUME) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000005
NUMBER OF FORMAT 9 DSCBS - 0000005
... several sets of similar RESUME related output sets omitted
CVAFFILT (RESUME) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000005
NUMBER OF FORMAT 9 DSCBS - 0000005
CVAFFILT RESUME OPERATION COMPLETE - ALL DSCBS RETURNED ( 0001186 )

CVFLT1E END OF OUTPUT MESSAGES

Figure C-75  CVFLT1E output from sample job - volume MLDC65 (1 of 3)
This output shows the following results:

- These messages relate to volume MLDC65 (EAV).
  - CVAFFILT CALL: EADSCB=OK KEYWORD CODED
    - The first set of output relates to a call to retrieve data about two data sets
      MHLRES1.EAV.PDS01
      and
      MHLEAV.EXTKSDS.DATA
    Data set MHLRES1.EAV.PDS01 is not on volume MHLC65 so no Format 1 DSCB is returned.
    Data set MHLEAV.EXTKSDS.DATA is on volume MHLC65 and is an EAS data set, so a Format 8 and a Format 9 DSCB is returned.
      - The second set of output relates to a call to retrieve data about the single data set
        MHLRES1.EAV.PDSE01
    Data set MHLRES1.EAV.PDSE01 is not on volume MHLC65 so no Format 1 DSCB is returned.
    - The third set of output relates to a call to retrieve data about the single data set
      MHLRES1.EAV.PDS01
    Data set MHLRES1.EAV.PDS01 is not on volume MHLC65 so no Format 1 DSCB is returned.
    - The fourth set of output relates to a call to retrieve data about the volume
      MHLC65
    The initial CVAFFILT call has found that there are more than 11 DSCBs on the volume. As a consequence a number of RESUME calls have been made retrieving the DSCBs until all have been retrieved.
    As the INITIAL or RESUME calls complete a list of the DSCB types found is listed.
    non-EAS data sets have a Format-1 DSCB and can have one or more Format-3 DSCBs. EAS data set have a Format-8 DSCB and a Format-9 DSCB and can have one or more Format-3 DSCBs and these are reflected on the volume.
    There are 1186 DSCBs of type 1 or type 8 identified on the volume, plus several type 3 and type 8.
    In Figure C-76 and Figure C-77 we show part of an ISPF listing which shows that there were 1186 data sets identified on the volume.

```
<table>
<thead>
<tr>
<th>Command - Enter &quot;/&quot; to select action</th>
<th>Message</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>MHLEAV.EDSBIG02.DATA</td>
<td>MLDC65</td>
<td></td>
</tr>
<tr>
<td>MHLEAV.EDSBIG02.INDEX</td>
<td>MLDC65</td>
<td></td>
</tr>
<tr>
<td>MHLEAV.EXTKSDS.DATA</td>
<td>MLDC65</td>
<td></td>
</tr>
<tr>
<td>MHLEAV.EXTKSDS.IX</td>
<td>MLDC65</td>
<td></td>
</tr>
<tr>
<td>MHLEAV.PE.D175843.DATA</td>
<td>MLDC65</td>
<td></td>
</tr>
<tr>
<td>MHLEAV.PE.D175843.INDEX</td>
<td>MLDC65</td>
<td></td>
</tr>
<tr>
<td>MHLEAV.PE.D175849.DATA</td>
<td>MLDC65</td>
<td></td>
</tr>
</tbody>
</table>
```

*Figure C-76  ISPF 3.4 listing extract from Volume MLDC65*
CVFLT1E START OF OUTPUT MESSAGES VOLUME: MHLS1A

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN: MHLRES1.EAV.PDS01
AND FOR DSN: MHLEAV.EXTKSDS.DATA
NUMBER OF FORMAT 1 DSCBS - 0000001
NUMBER OF FORMAT 3 DSCBS - 0000001
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN: MHLRES1.EAV.PDSE01
NUMBER OF FORMAT 1 DSCBS - 0000001
NUMBER OF FORMAT 3 DSCBS - 000010
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
NO DSCBS RETURNED FROM CVAFFILT
CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN: MHLRES1.EAV.PDS01
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000

RC04 VERIFIED - CVSTAT 064 RESUME IS NECESSARY
CVAFFILT (INITIAL) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
NUMBER OF FORMAT 1 DSCBS - 0000009
NUMBER OF FORMAT 3 DSCBS - 0000001
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000
CVAFFILT (RESUME) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
NUMBER OF FORMAT 1 DSCBS - 0000001
NUMBER OF FORMAT 3 DSCBS - 000010
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000
... several sets of similar RESUME related output sets omitted

CVAFFILT (RESUME) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
NUMBER OF FORMAT 1 DSCBS - 0000001
NUMBER OF FORMAT 3 DSCBS - 0000010
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000
CVAFFILT (RESUME) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
NUMBER OF FORMAT 1 DSCBS - 0000008
NUMBER OF FORMAT 3 DSCBS - 0000002
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000
CVAFFILT RESUME OPERATION COMPLETE - ALL DSCBS RETURNED (0000019)

CVFLT1E END OF OUTPUT MESSAGES

Figure C-77 CVFLT1E output from sample job - Volume MHLS1A (2 of 3)
This output shows the following results:

- These messages relate to volume MHLS1A (non-EAV).
  - CVAFFILT CALL: EADSCB=OK KEYWORD CODED
    - The first set of output relates to a call to retrieve data about two data sets
      MHLRES1.EAV.PDS01
      and
      MHLEAV.EXTKSDS.DATA
      Data set MHLRES1.EAV.PDS01 is on volume MHLS1A so one Format 1 DSCB is returned.
      Data set MHLEAV.EXTKSDS.DATA is not on volume MHLS1A so no DSCB is returned.
    - The second set of output relates to a call to retrieve data about the single data set
      MHLRES1.EAV.PDSE01
      Data set MHLRES1.EAV.PDSE01 is on volume MHLS1A so a Format 1 DSCB is returned. It is a PDSE in 123 extents so three Format 3 DSCBs are also returned.
    - The third set of output relates to a call to retrieve data about the single data set
      MHLRES1.EAV.PDS01
      Data set MHLRES1.EAV.PDS01 is not on volume MHLS1A so no Format 1 DSCB is returned.
    - The fourth set of output relates to a call to retrieve data about the volume MHLS1A
      The initial CVAFFILT call has found that there are more than 11 DSCBs on the volume. As a consequence a RESUME call has been made retrieving the DSCBs until all have been retrieved.
      As the INITIAL or RESUME calls complete a list of the DSCB types found is listed.
      non-EAS data sets have a Format-1 DSCB and can have one or more Format-3 DSCBs.
      Volume MHLS1A is not an EAV volume so no Format 8 or Format 9 DSCBs are returned.
      There are 19 DSCBs of type 1 identified on the volume, plus several type 3.
      In Figure C-78 and Figure C-79 we show part of an ISPF listing which shows that there were 19 data sets identified on the volume.

<table>
<thead>
<tr>
<th>Command - Enter &quot;/&quot; to select action</th>
<th>Message</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>MHLRES1.BCAT.LIST.DCB</td>
<td></td>
<td>MHLS1A</td>
</tr>
<tr>
<td>MHLRES1.EAV.PDSE01</td>
<td></td>
<td>MHLS1A</td>
</tr>
<tr>
<td>MHLRES1.EAV.PDS01</td>
<td></td>
<td>MHLS1A</td>
</tr>
<tr>
<td>MHLRES1.HCD.TERM</td>
<td></td>
<td>MHLS1A</td>
</tr>
<tr>
<td>MHLRES1.SPFTEMP8.CNTL</td>
<td></td>
<td>MHLS1A</td>
</tr>
</tbody>
</table>

Figure C-78  ISPF 3.4 listing extract from Volume MHLS1A
CVFLT1E START OF OUTPUT MESSAGES VOLUME: MLDC66

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
NO DSCBS RETURNED FROM CVAFFILT
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000
CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN: MHLRES1.EAV.PDS01
AND FOR DSN: MHLEAV.EXTKSDS.DATA
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
NO DSCBS RETURNED FROM CVAFFILT
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000
CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN: MHLRES1.EAV.PDSE01
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
NO DSCBS RETURNED FROM CVAFFILT
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000
CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN: MHLRES1.EAV.PDS01
NUMBER OF FORMAT 1 DSCBS - 0000000
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000

RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL
NUMBER OF FORMAT 1 DSCBS - 0000009
NUMBER OF FORMAT 3 DSCBS - 0000000
NUMBER OF FORMAT 8 DSCBS - 0000000
NUMBER OF FORMAT 9 DSCBS - 0000000

CVFLT1E END OF OUTPUT MESSAGES

Figure C-79 CVFLT1E output from sample job - Volume MHLC66 (3 of 3)
This output shows the following results:

- These messages relate to volume MHLC66 (EAV).
  - CVAFFILT CALL: EADSCB=OK KEYWORD CODED
    - The first set of output relates to a call to retrieve data about two data sets
      MHLRES1.EAV.PDS01
      and
      MHLEAV.EXTKSDS.DATA
      Data set MHLRES1.EAV.PDS01 is not on volume MHLC66 so no DSCB is returned.
      Data set MHLEAV.EXTKSDS.DATA is not on volume MHLC66 so no DSCB is returned.
    - The second set of output relates to a call to retrieve data about the single data set
      MHLRES1.EAV.PDSE01
      Data set MHLRES1.EAV.PDSE01 is not on volume MHLC66 so no DSCB is returned.
    - The third set of output relates to a call to retrieve data about the single data set
      MHLRES1.EAV.PDS01
      Data set MHLRES1.EAV.PDS01 is not on volume MHLC66 so no Format 1 DSCB is returned.
    - The fourth set of output relates to a call to retrieve data about the volume.
      MHLC66 has only 9 data sets on it so no RESUME processing was necessary.

In Figure C-80 we show an ISPF listing which shows that there were 9 data sets identified
on the volume.

![Table]

<table>
<thead>
<tr>
<th>Command</th>
<th>Message</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS1.ANTMAIN.FCWK0001.DATA</td>
<td>MLD66</td>
<td></td>
</tr>
<tr>
<td>SYS1.ANTMAIN.FCWK0002.DATA</td>
<td>MLD66</td>
<td></td>
</tr>
<tr>
<td>SYS1.ANTMAIN.FCWK0003.DATA</td>
<td>MLD66</td>
<td></td>
</tr>
<tr>
<td>SYS1.ANTMAIN.FCWK0004.DATA</td>
<td>MLD66</td>
<td></td>
</tr>
<tr>
<td>SYS1.TEST.MLDC66</td>
<td>MLD66</td>
<td></td>
</tr>
<tr>
<td>SYS1.TEST.SC63</td>
<td>MLD66</td>
<td></td>
</tr>
<tr>
<td>SYS1.TEST.SC63.XXX</td>
<td>MLD66</td>
<td></td>
</tr>
<tr>
<td>SYS1.VTOCIX.MLDC66</td>
<td>MLD66</td>
<td></td>
</tr>
<tr>
<td>SYS1.VVDS.VMLDC66</td>
<td>MLD66</td>
<td></td>
</tr>
</tbody>
</table>

Figure C-80  ISPF 3.4 listing from Volume MHLC66

A version of the program CVFLT1E called CVFLT1O was run. It is almost identical to
CVFLT1E except that the CVAFFILT macros are issue with EADSCB=NOTOK. In CVFLT1E
the CVAFFILT macros are issued with EADSCB=OK. The example simulates the effect of
running an existing program that uses CVAFFILT with EADSCB not specified because the
default on z/OS V1R10 is the same as EADSCB=NOTOK. Releases of z/OS prior to V1R10
do not support the EADSCB keyword on the CVAF macros, but the earlier code has the same
effect as EADSCB=NOTOK.
C.8.1 EAV Migration Assistance Tracker considerations: CVAFFILT

The EAV Migration Assistance Tracker will detect use of the CVAFFILT service that has not been updated for EAV compatibility.

Any appearance on the tracker list must be investigated and resolved because use of an un-extended OBTAIN against an EAV volume in due course will fail.

For information about the EAV Migration Tracker, refer to Appendix C of DFSMS Advanced Services, SC26-7400 and in this document to “Migration Assistance Tracker” on page 641.

In Figure C-81 we show the output from the Migration Assistance Tracker that relates to the run of CVFLT1O run against volume MHLC65. It shows that CVFLT1O needs attention before it is used against an EAV format volume.

----TRACKING INFORMATION----- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM
SMS-E:1 CVAFFILT STAT086 4560601 MHLRES1F CVFLT1O 692 42 2

Figure C-81   CVFLT1O Migration Assistance Tracker entry

Source code listing
In Figure C-82 on page 581 through to Figure C-98 on page 597 we show assembler source for a program to issue the CVAFFILT macro.

In this source, the following coding relates to EAV support:

- In routine CVAFRD1 which starts at line 05810014
  - the CVAFFILT call which starts at line 05860014 shows the use of EADSCB=OK
- In routine CVAFRD2 which starts at line 05960014
  - the CVAFFILT call which starts at line 06030014 shows the use of EADSCB=OK
- In routine CVAFRDA which starts at line 06130014
  - the CVAFFILT call which starts at line 06150014 shows the use of EADSCB=OK
- In the Release Work Areas routine which starts at line 06710014
  - the CVAFFILT call which starts at line 06730014 shows the use of EADSCB=OK
- In routine CVAFRL which starts at line 06870014
  - the CVAFFILT call which starts at line 06890014 shows the use of EADSCB=OK

In order to set up the CVAFFILT calls, the VTOC was opened and read directly. In order to allow a VTOC on an EAV volume to be READ, the DCB coded for that purpose had to be extended by the addition of the DCBE macro which is used to specify EADDCB=OK

- Line 08260014 shows the addition of ,DCBE=VTOCDCBE
- Line 08340014 shows the addition of VTOCDCBE DCBD EADSCB=OK
Figure C-82  Source of CVFLT1E program (1 of 15)
*RC04 VERIFIED - CVSTAT 064 RESUME IS NECESSARY
*CVAFFILT (INITIAL) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
*NUMBER OF FORMAT 1 DSCBS - 0000002
*NUMBER OF FORMAT 3 DSCBS - 0000000
*NUMBER OF FORMAT 8 DSCBS - 0000004
*NUMBER OF FORMAT 9 DSCBS - 0000004
*CVAFFILT (RESUME) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
*NUMBER OF FORMAT 1 DSCBS - 0000000
*NUMBER OF FORMAT 3 DSCBS - 0000000
*NUMBER OF FORMAT 8 DSCBS - 0000005
*NUMBER OF FORMAT 9 DSCBS - 0000005
*CVAFFILT (RESUME) RETURNED THE FOLLOWING DSCBS FOR THE VOLUME:
*NUMBER OF FORMAT 1 DSCBS - 0000000
*NUMBER OF FORMAT 3 DSCBS - 0000000
*NUMBER OF FORMAT 8 DSCBS - 0000005
*NUMBER OF FORMAT 9 DSCBS - 0000005
*CVAFFILT RESUME OPERATION COMPLETE - ALL DSCBS RETURNED ( 0001187 )
* * *
CVFLT1E END OF OUTPUT MESSAGES
* *
*-----------------------------------------------*  
* NOTE: THE NUMBER OF DSCBS RETURNED WILL VARY IF THE DATASET DOES *
* NOT EXIST ON THE VOLUME OF IF THE DATASET EXTENT IS NOT AS *
* LISTED ABOVE OR IF THERE IS NO VTOC INDEX ON THE VOLUME. *
* * *
* NOTE: WHEN CREATING A PDSE ON A SMS MANAGED VOLUME A VVDS *
* WILL ALSO BE CREATED AND THE FMT1 COUNT WILL BE *
* INCREASED BY ONE. *
* * *
*********************************************************************** 
*********************************************************************** 
* * *
CVFLT1E - LOGIC NOTES *
* *
** THIS EXAMPLE WILL PERFORM THE FOLLOWING: **
* *
* INITIALIZATION *
* - OBTAIN THE NECESSARY INFORMATION FROM THE DASD VOLUME *
* - OPEN AN OUTPUT FILE AND WRITE THE NECESSARY OUTPUT MESSAGES *
* - INITIALIZE THE NECESSARY BUFFER LIST FOR CVAFFILT *
* * *
* MAINLINE *
* - INITIALIZE A FCL TO READ FOR TWO SPECIFIC DATASETS *
* - ISSUE CVAFFILT READ TO READ THE DSCBS FOR THE TWO DATASETS *
* - CHECK THE RETURN CODE AND CVSTAT CODE FROM CVAFFILT *
* - COUNT THE NUMBER OF FMT1/3/8/9 DSCBS FOR THE REQUEST *
* - FORMAT THE DSCB COUNTS AND WRITE TO THE OUTPUT DATASET *
* - ISSUE CVAFFILT RLSE TO RELEASE THE WORK AREAS USED *
* * *

Figure C-83  Source of CVFLT1E program (2 of 17)
Figure C-84  Source of CVFLT1E program (3 of 17)
INITIAL DS OH INITIALIZATION SECTION 01600014
BAL R14,IVOLRTN INVOKE RTN TO IDENTIFY THE VOLUME(S) 01610014
BAL R14,OPENRTN INVOKE OPEN OUTPUT DATASET RTN 01620014
L R1,UCBADD 01630014
MVC VOLSER(6),28(R1) 01640014
MVC POETLINE(133),STRMTSGM move start MSG TO LINE 01650014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 01660014
MVC POETLINE(133),BLNKLNE move blank line TO LINE 01670014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 01680014
BAL R14,BUFLRTN INVOKE RTN TO INIT BUFFER LIST (H/E) 01690014

* * *

MAINLINE DS OH MAINLINE SECTION 01700014
MVI DSNBR,X'02' SET DSNBR FLAG TO TWO 01710014
BAL R14,FCL1RTN INVOKE RTN TO INIT FCL (DS01/DS02) 01720014
BAL R14,CVAFLRTN INVOKE CVAFFILT READ RTN 01730014
BAL R14,TSTRCRTN INVOKE TEST RC RTN 01740014
L R15,RETCODE LOAD R15 WITH RETURN CODE 01750014
CH R15,RCODE00 IS RC = 0 01760014
BNE MAIN0010 NO - DO NOT FORMAT/PRINT LINES 01770014
MVC DS(44),DS01 ELSE MOVE DSN FOR DS01 TO MSG LINE 01780014
MVC PDETLINE(133),DSCBMSGA MOVE MSG TO LINE 01790014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 01800014
MVC DS2(44),DS02 MOVE DSN FOR DS02 TO MSG LINE 01810014
MVC PDETLINE(133),DSCBMSGB MOVE MSG TO LINE 01820014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 01830014
BAL R14,FMTOPRTN INVOKE FORMAT OUTPUT MSG RTN 01840014
B MAIN0020 BRANCH TO MAIN0020 01850014

MAIN0010 DS OH CVAFFILT - CVSTAT LOGICAL ERROR 01860014
MVC DS3(44),DS01 MOVE DSN FOR DS01 TO MSG LINE 01870014
MVC PDETLINE(133),DSCBMSGC MOVE MSG TO LINE 01880014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 01890014
MVC DS3(44),DS02 MOVE DSN FOR DS02 TO MSG LINE 01900014
MVC PDETLINE(133),DSCBMSGD MOVE MSG TO LINE 01910014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 01920014
BAL R14,FMTOPRTN INVOKE FORMAT OUTPUT MSG RTN 01930014
B MAIN0020 BRANCH TO MAIN0020 01940014

MAIN0020 DS OH 01950014
MVC PDETLINE(133),DSCBMSGE MOVE MSG TO LINE 01960014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 01970014
BAL R14,IVOLRTN INVOKE RTN TO IDENTIFY THE VOLUME(S) 01980014
BAL R14,OPENRTN INVOKE OPEN OUTPUT DATASET RTN 01990014
MVI DSNBR,X'01' SET DSNBR FLAG TO ONE 02000014
BAL R14,FCL2RTN INVOKE RTN TO INIT FCL (DS03) 02010014
BAL R14,CVAFLRTN INVOKE CVAFFILT READ RTN 02020014
BAL R14,TSTRCRTN INVOKE TEST RC RTN 02030014
L R15,RETCODE LOAD R15 WITH RETURN CODE 02040014
CH R15,RCODE00 IS RC = 0 02050014
BNE MAIN0030 NO - DO NOT FORMAT/PRINT LINES 02060014
MVC DS(44),DS03 ELSE MOVE DSN FOR DS03 TO MSG LINE 02070014
MVC PDETLINE(133),DSCBMSGF MOVE MSG TO LINE 02080014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 02090014
BAL R14,FMTOPRTN INVOKE FORMAT OUTPUT MSG RTN 02100014
B MAIN0040 BRANCH TO MAIN0040 02110014

MAIN0030 DS OH CVAFFILT - CVSTAT LOGICAL ERROR 02120014
MVC DS3(44),DS03 MOVE DSN FOR DS03 TO MSG LINE 02130014

Figure C-85 Source of CVFLT1E program (4 of 17)
MVC PDETLINE(133), DSCBMSGC MOVE MSG TO LINE 02140014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 02150014

MAIN0040 DS OH 02160014
MVC PDETLINE(133), BLNKLNE MOVE BLANK LINE TO LINE 02170014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 02180014
BAL R14, CVAFRL INVOKE CVAFFILT RELEASE RTN 02190014
BAL R14, FCLR3RTN INVOKE RTN TO INIT FCL (DSO1) 02200014
BAL R14, CVAFRD1 INVOKE CVAFFILT READ RTN 02210014
BAL R14, TSTRCRTN INVOKE TEST RC RTN 02220014
L R15, RETCODE LOAD R15 WITH RETURN CODE 02230014
CH R15, RCODE00 IS RC = 0 02240014
BNE MAIN0050 NO - DO NOT FORMAT/PRINT LINES 02250014
MVC DS(44), DS01 ELSE MOVE DSN FOR DS01 TO MSG LINE 02260014
MVC PDETLINE(133), DSCBMSGA MOVE MSG TO LINE 02270014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 02280014
BAL R14, FMTOPRTN INVOKE FORMAT OUTPUT MSG RTN 02290014
B MAIN0060 BRANCH TO MAIN0060 02300014

MAIN0050 DS OH CVAFFILT - CVSTAT LOGICAL ERROR 02310014
MVC DS3(44), DS01 MOVE DSN FOR DS01 TO MSG LINE 02320014
MVC PDETLINE(133), DSCBMSGA MOVE MSG TO LINE 02330014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 02340014

MAIN0060 DS OH 02350014
MVC PDETLINE(133), BLNKLNE MOVE BLANK LINE TO LINE 02360014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 02370014
BAL R14, CVAFRL INVOKE CVAFFILT RELEASE RTN 02380014
BAL R14, FCLR4RTN INVOKE RTN TO INIT FCL (ENTIRE VOL) 02390014
BAL R14, CVAFRD4 INVOKE CVAFFILT READ RTN (FOR RESUME) 02400014
LA R9, CVPLDEFA ESTABLISH ADDRESSABILITY 02410014
USING CVPLMAP, R9 TO THE CVPL (FOR CVSTAT) 02420014
*********************************************************************** 02430014
* FINALIZATION * 02440014
FINAL DS OH FINALIZATION SECTION 02450014
MVC PDETLINE(133), BLNKLNE MOVE BLANK LINE TO LINE 02460014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 02470014
MVC PDETLINE(133), ENDMSEG MOVE END MSG TO LINE 02480014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 02490014
BAL R14, CLOSERTN INVOKE CLOSE OUTPUT DATASET RTN 02500014
L R13, 4(R13) RESTORE REGISTER 02510014
LM R14, R12, 12(R13) RESTORE CALLERS REGISTERS 02520014
LA R15, 0 SET RC TO 0 02530014
BR R14 RETURN TO CALLER 02540014
*********************************************************************** 02550014
* OPENRTN * 02560014
* - ROUTINE TO OPEN OUTPUT FILE USED BY THIS MODULE * 02570014
OPENRTN DS OH OPEN FILES ROUTINE 02580014
ST R14, OPENSAVE STORE C(R14) INTO SAVE AREA 02590014
OPEN (OUTFILE, (OUTPUT)) OPEN THE OUTDD OUTPUT FILE FOR MSGS 02600014
TM OUTFILE+(DCBOFLGS-IHADCB), DCBOFOPEN IS FILE OPEN? 02610014
BO OPENEXIT FILE OPEN OK - EXIT OPEN RTN 02620014
LA R1, EABN101 OUTPUT FILE NOT OPEN-USER ABEND 101 02630014
BAL R14, ABENDRTN INVOKE ABEND ROUTINE 02640014
OPENEXIT DS OH EXIT FROM OPEN FILES ROUTINE 02650014
L R14, OPENSAVE LOAD C(OPENSAVE) INTO R14 02660014
BR R14 EXIT 02670014

Figure C-86 Source of CVFLT1E program (5 of 17)
**CLOSERTN**

* - ROUTINE TO CLOSE OUTPUT FILE USED BY THIS MODULE *

CLOSERTN DS OH CLOSE FILES ROUTINE

ST R14,CLOSSAVE STORE C(R14) INTO SAVE AREA
CLOSE (OUTFILE) CLOSE OUTPUT FILE
LTR R15,R15 CHECK IF CLOSED OK
BZ CLOSEXIT IF OK BRANCH TO CLOSEXIT
LA R1,EABN102 ELSE SETUP FOR USER ABEND 102
BAL R14,ABENDRTN INVOKE ABEND ROUTINE

CLOSEXIT DS OH EXIT FROM CLOSE ROUTINE
L R14,CLOSSAVE LOAD C(CLOSSAVE) INTO R14
BR R14 EXIT

**ABENDRTN**

* - FORCE AN ABEND ROUTINE *

ABENDRTN DS OH ABEND ROUTINE

ST R14,ABENSAVE STORE C(R14) INTO SAVE AREA
ABEND (R1),DUMP ISSUE USER ABEND 102
BAL R14,ABENDRTN INVOKE ABEND ROUTINE

**IDVOLRTN**

* - OBTAIN THE NECESSARY INFORMATION FROM THE DASD VOLUME *

IDVOLRTN DS OH IDENTIFY VOLUME ROUTINE

ST R14,IDVLSAVE STORE C(R14) INTO SAVE AREA
RDJFCB (VTODCB,(INPUT)) READ JFCB / OPEN VTOC
MVI JFCB1,X'04' PUT IN ID FOR FORMAT 4
MVC JFCB1+1(43),JFCB1 SETUP FOR VTOC OPEN
OPEN (VTOCDCB,(INPUT)),TYPE=J OPEN VTOC (OPEN TYPE=J)
TM VTOCDCB+(DCBOFLGS-IHADCB),DCBOFOPN
BO IDVOL010 BRANCH TO IDVOL010 - GOOD OPEN
LA R1,EABN100 ELSE SETUP FOR USER ABEND 100
BAL R14,ABENDRTN INVOKE ABEND ROUTINE

IDVOL010 DS OH GOOD OPEN - OBTAIN VOLUME INFORMATION

SLR RDEB,RDEB INIT REG1 FOR DEB PTR
SLR RUCB,RUCB INIT REG2 FOR UCB PTR
ICM RDEB,B'0111',VTOCDCB+(DCBDEBA-IHADCB) GET DEB ADDRESS
ST RDEB,DEBADD SAVE DEB ADDRESS INTO R1
ICM RUCB,B'0111',(DEBBASND-DEBBASIC)+(DEBUCBA-DEBDASD)(RDEB)
ST RUCB,UCBADD SAVE UCB ADDRESS INTO R2

IDVLEXIT DS OH EXIT FROM IDVOLRTN
L R14,IDVLSAVE LOAD C(IDVLSAVE) INTO R14
BR R14 EXIT

**TSTRCRTN**

* - TEST RETURN CODE FROM CVAFFILT *

TSTRCRTN DS OH CHECK RETURN CODE ROUTINE

ST R14,TSTRSAVE STORE C(R14) INTO SAVE AREA
L R15,RETCODE REG1 FOR DEB PTR
ICM RDEB,B'0111',VTOCDCB+(DCBDEBA-IHADCB) GET DEB ADDRESS
ST RDEB,DEBADD SAVE DEB ADDRESS INTO R1
ICM RUCB,B'0111',((DEBBASND-DEBBASIC)+(DEBUCBA-DEBDASD))(RDEB)
ST RUCB,UCBADD SAVE UCB ADDRESS INTO R2

TSTRCRTN DS OH EXIT FROM TSTRCRTN
L R15,RETCODE LOAD R15 WITH RC SAVED FROM LAST CALL
CH R15,RCODE00 IF RETURN CODE = 00
CH R15,RCODE04 IF RETURN CODE = 04
BE TSTRC00 BRANCH TO PROCESS RC00
BE TSTRC04 BRANCH TO PROCESS RC04
TSTRCER DS OH ELSE PRINT GENERAL ERROR MESSAGE 03260014
  MVC POETLINE(133),RCERMGMV ERROR MSG TO PRINT LINE 03270014
  PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 03280014
  B TSTREXIT BRANCH TO EXIT RTN 03290014
TSTRCOO DS OH PROCESS RETURN CODE = 00 03300014
  MVC POETLINE(133),ROCOMMSG MOVE RCOO MSG TO PRINT LINE 03310014
  PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 03320014
  BAL R14,FMTCTRTN BRANCH TO FMT COUNT RTN 03330014
  BAL R14,FMTORPTN INVOKE FORMAT OUTPUT MSG ROUTINE 03331020
  B TSTREXIT BRANCH TO EXIT RTN 03340014
TSTRCO4 DS OH PROCESS RETURN CODE = 04 03350014
  CLI CVSTAT,STATO64 DO WE NEED RESUME? 03360014
  BNE TSTR0010 NO - PRINT LOGICAL ERROR MSG 03370014
  BAL R14,CVAFRS INVOKE CVAFRS RESUME ROUTINE 03380014
  B TSTREXIT BRANCH TO EXIT RTN 03390014
TSTR0010 DS OH PRINT RCO4-OTHER LOGICAL ERROR/CVSTAT 03400014
  MVC POETLINE(133),RCO4MSG MOVE RCO4 MSG TO PRINT LINE 03410014
  PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 03420014
  BAL R14,FMTCTRTN BRANCH TO FMT COUNT RTN 03430014
TSTREXIT DS OH EXIT FROM TSTRCRTN 03433004
  L R14,TSTRSAVE LOAD C(TSTRSAVE) INTO R14 03440014
  BR R14 EXIT 03450014
*********************************************************************** 03460014
* FMTCTRTN *                                                            03470014
* - COUNT THE NUMBER OF FMT1 AND FMT3                                   03480014
*                   AND FMT8 AND FMT9 DSCBS FOR THE REQUEST             03490014
FMTCTRTN DS OH COUNT DSCBS RETURNED ROUTINE 03500014
  ST R14,FMTCSAVE STORE C(R14) INTO SAVE AREA 03510014
  SLR R6,R6 ZERO OUT R6 03520014
  SLR R7,R7 ZERO OUT R7 03530014
  SLR R8,R8 ZERO OUT R8 03540014
  SLR R9,R9 ZERO OUT R9 - BORROWING R9 FROM CVPLDEFA ADDR. 03550014
  CLI DSNBR,X'01' IF DSNBR FLAG = 1 03560014
  BE FMTC0010 BRANCH TO LOAD ADDR OF FCL 03570014
  LA R4,FCLDEF2 ELSE LOAD R4 WITH ADDR OF FCL2 03580014
  B FMTC0020 BRANCH TO SET USING 03590014
FMTC0010 DS OH COUNT DSCBS RETURNED ROUTINE 03600014
  LA R4,FCLDEF LOAD R4 WITH ADDR OF FCL 03610014
FMTC0020 DS OH CONTINUE - SET USING 03620014
  USING FCLMAP,R4 ESTABLISH ADDRESSABILITY TO FCL 03630014
  SLR R5,R5 ZERO OUT R5 03640014
  ICM R5,B'0011',FCLDSCHB DETERMINE IF ANY DSCBS RETURNED 03650014
  BZ FMTC0060 NO - GO AND PRINT APPROPRIATE MSG 03660014
  LA R4,DSCBDEF LOAD R4 WITH ADDR OF DSCB MAP 03670014
  DC H'O' ABEND 03680014
  USING DSCBMAP,R4 ESTABLISH ADDRESSABILITY TO DSCB 03690014
FMTC0030 DS OH COUNT DSCBS BY TYPE RETURNED 03700014
  CLI DS1FMID,T,X'F1' IF FORMAT1 03710014
  BE FMTC0040 BRANCH TO FMTC0040 03720014
  CLI DS1FMID,T,X'F8' IF FORMAT1 03730014
  BE FMTC0033 BRANCH TO FMTC0033 03740014
  CLI DS1FMID,T,X'F3' IF FORMAT3 03750014
  BE FMTC0032 BRANCH TO FMTC0032 03760014
  CLI DS1FMID,T,X'F9' IF FORMAT8 03770014
  BE FMTC0035 BRANCH TO FMTC0035 03780014

Figure C-88  Source of CVFLT1E program (7 of 17)
FMTC0032 DS OH FMTC032 ADD 1 TO FORMAT3 COUNT 03790014
LA R7,(R7) ADD 1 TO FORMAT3 COUNT 03800014
B FMTC0050 BRANCH TO FMTC0050 03810014
FMTC0033 DS OH FMTC033 ADD 1 TO FORMAT8 COUNT 03820014
LA R8,(R8) ADD 1 TO FORMAT8 COUNT 03830014
B FMTC0050 BRANCH TO FMTC0050 03840014
FMTC0035 DS OH FMTC035 ADD 1 TO FORMAT9 COUNT 03850014
LA R9,(R9) ADD 1 TO FORMAT9 COUNT 03860014
B FMTC0050 BRANCH TO FMTC0050 03870014
FMTC0040 DS OH FMTC0040 FORMAT1 INCREMENT 03880014
LA R6,(R6) ADD 1 TO FORMAT1 COUNT 03890014
FMTC0050 DS OH FMTC0050 PROCESS THROUGH DSCBS 03900014
LA R4,DSCBSIZ(R4) ADD DSCBSIZ TO R4 03910014
BCT R5,FMT0030 SUBTRACT 1 FROM DSCB COUNT AND CONT 03920014
ST R6,RETF1 STORE #FMT1S INTO RETF1 03930014
ST R7,RETF3 STORE #FMT3S INTO RETF3 03940014
ST R8,RETF8 STORE #FMT8S INTO RETF8 03950014
ST R9,RETF9 STORE #FMT9S INTO RETF9 03960014
L R9,RETF18 GET ACCUMULATED COUNT 03970014
AR R9,R6 ADD ON FMT1 03980014
AR R9,R8 ADD ON FMT8 03990014
ST R9,RETF18 STORE 04000014
DROP R4 DROP R4 USING 04010014
B FMTC0060 BRANCH AROUND FMTC0060 04020014
FMTC0060 DS OH PRINT MSG - NO DSCBS RETURNED 04030014
MVC PDETLINE(133),NODSCBMOV MSG TO PRINT LINE 04040014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 04050014
SLR R6,R6 ZERO OUT R6 04060014
ST R6,RETF1 LOAD R6 WITH NBR OF FMT1S RETURNED 04070014
CVD R6,WF1 CONVERT TO DEC FOR PRINTING 04080014
UNPK WFMTREC+29(7),WF1 UNPACK TO FORMAT LINE 04090014
OI WFMTREC+35,X'F0' SET APPROPRIATE BITS 04100014
MVC PDETLINE(133),WFMTREC MOVE RECORD TO OUTPUT LINE 04110014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 04120014

***********************************************************************
* FMTOPRTN *
* - FORMAT THE DSCB COUNTS AND WRITE TO THE OUTPUT DATASET *
***********************************************************************

FMTC0060 DS OH PRINT MSG - NO DSCBS RETURNED 04030014
MVC PDETLINE(133),NODSCBMOV MSG TO PRINT LINE 04040014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 04050014
SLR R6,R6 ZERO OUT R6 04060014
ST R6,RETF1 LOAD R6 WITH NBR OF FMT1S RETURNED 04070014
CVD R6,WF1 CONVERT TO DEC FOR PRINTING 04080014
UNPK WFMTREC+29(7),WF1 UNPACK TO FORMAT LINE 04090014
OI WFMTREC+35,X'F0' SET APPROPRIATE BITS 04100014
MVC PDETLINE(133),WFMTREC MOVE RECORD TO OUTPUT LINE 04110014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 04120014

***********************************************************************
* FMTOPRTN *
* - FORMAT THE DSCB COUNTS AND WRITE TO THE OUTPUT DATASET *
***********************************************************************

FMTC0060 DS OH PRINT MSG - NO DSCBS RETURNED 04030014
MVC PDETLINE(133),NODSCBMOV MSG TO PRINT LINE 04040014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 04050014
SLR R6,R6 ZERO OUT R6 04060014
ST R6,RETF1 LOAD R6 WITH NBR OF FMT1S RETURNED 04070014
CVD R6,WF1 CONVERT TO DEC FOR PRINTING 04080014
UNPK WFMTREC+29(7),WF1 UNPACK TO FORMAT LINE 04090014
OI WFMTREC+35,X'F0' SET APPROPRIATE BITS 04100014
MVC PDETLINE(133),WFMTREC MOVE RECORD TO OUTPUT LINE 04110014
PUT OUTFILE,PDETLINE WRITE A RECORD TO THE OUTPUT FILE 04120014

Figure C-89  Source of CVFLT1E program (8 of 17)
MVC MSG(29),DSCBMSG3 MOVE MSG TO FORMAT LINE 04300014
L R7,RETF3 LOAD R7 WITH NBR OF FMT3S RETURNED 04310014
CVD R7,WF3 CONVERT TO DEC FOR PRINTING 04320014
UNPK WMTRREC+29(7),WF3 UNPACK TO FORMAT LINE 04330014
OI WMTRREC+35,X'F0' SET APPROPRIATE BITS 04340014
MVC POETLINE(133),WMTRREC MOVE RECORD TO OUTPUT LINE 04350014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 04360014
MVC MSG(29),DSCBMSG8 MOVE MSG TO FORMAT LINE 04370014
L R8,RETF8 LOAD R8 WITH NBR OF FMT8S RETURNED 04380014
CVD R8,WF8 CONVERT TO DEC FOR PRINTING 04390014
UNPK WMTRREC+29(7),WF8 UNPACK TO FORMAT LINE 04400014
OI WMTRREC+35,X'F0' SET APPROPRIATE BITS 04410014
MVC POETLINE(133),WMTRREC MOVE RECORD TO OUTPUT LINE 04420014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 04430014
MVC MSG(29),DSCBMSG9 MOVE MSG TO FORMAT LINE 04440014
L R9,RETF9 LOAD R9 WITH NBR OF FMT9S RETURNED 04450014
CVD R9,WF9 CONVERT TO DEC FOR PRINTING 04460014
UNPK WMTRREC+29(7),WF9 UNPACK TO FORMAT LINE 04470014
OI WMTRREC+35,X'F0' SET APPROPRIATE BITS 04480014
MVC POETLINE(133),WMTRREC MOVE RECORD TO OUTPUT LINE 04490014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 04500014
FMTOEXIT DS 0H EXIT FROM FMTOPRTN 04510014
LA R9,CVPLDEFA ESTABLISH ADDRESSABILITY RELOAD R9 04520014
L R14,FMTOSAVE LOAD C(FMTOSAVE) INTO R14 04530014
BR R14 EXIT 04540014
*********************************************************************** 04550014
* BUFLRTN *                                                             04560014
* - INITIALIZE BUFFER LIST HEADER (BFLH) *                              04570001
BUFLRTN DS 0H BUFFER LIST INITIALIZATION ROUTINE 04580014
ST R14,BUFLSAVE STORE C(R14) INTO SAVE AREA 04590014
XC BFLHDEF(BFLSIZE),BFLHDEF CLEAR BUFR LIST AREA 04600014
LA R1,BFLHDEF R1 - BUFFER LIST HEADER 04610014
USING BFLMAP,R1 ESTABLISH ADDRESSABILITY 04620014
MVI BFLHNOE,BUFBNBR SET NUMBER OF BUFFER ELEMENTS 04630014
OI BFLHLF,,BFLHDFCB IDENTIFY AS DSCB BUFR ELEMENT LIST 04640014
LA R2,BFLHDEF+BFLHLN R2 - FIRST BUFFER LIST ELEMENT 04650014
USING BFLE,R2 ESTABLISH ADDRESSABILITY 04660014
LA R3,DSCBDEF R3 - FIRST DSCB BUFFER 04670014
LA R4,BUFBNBR R4 = NUMBER OF ELEMENTS AND BUFRS 04680014
BFL0010 OI BFLELF,BFLECHR REQUEST CCHRHR ON RETURN 04690014
MVI BFLELTH,DSCBSIZ SET BUFR LNGTH TO FULL DSCB SIZE 04700014
ST R3,BFLEBUF ADDR(DSCB BUFFER) 04710014
LA R2,BFLELN(R2) R2 - NEXT BUFFER LIST ELEMENT 04720014
LA R3,DSCBSIZ(R3) R3 - NEXT DSCB BUFFER 04730014
BCT R4,BFL0010 LOOP THROUGH ALL ELEMENTS 04740014
DROP R1,R2 DROP TEMP USINGS 04750014
BUFLEXIT DS 0H EXIT FROM BUFLRTN 04760014
L R14,BUFLSAVE LOAD C(BUFLSAVE) INTO R14 04770014
BR R14 EXIT 04780014

Figure C-90  Source of CVFLT1E program (9 of 17)
* FCL1RTN *

* - INITIALIZE FILTER CRITERIA LIST (FCL) HEADER AND ELEMENT *

* - INITIALIZE A FCL TO READ FOR TWO SPECIFIC SEQ DATASETS *

FCL1RTN DS OH FCL INITIALIZATION ROUTINE
ST R14,FCLSAVE STORE C(R14) INTO SAVE AREA
XC FCLDEF2(FCLSIZE2),FCLDEF2 CLEAR FCL AREA
LA R1,FCLDEF2 R1 - FCL HEADER
USING FCLMAP,R1 ESTABLISH ADDRESSABILITY
LA R2,FCLHDEND R2 - FIRST FCL ELEMENT
USING FCLDSN,R2 ESTABLISH ADDRESSABILITY
MVC FCLID,CFLID SET THE EYECATCHER FCL
MVC FCLCOUNT,=H'2' SET NUMBER OF FCL ELEMENTS
MVI FCL1FLAG,X'80' SET FLAG FOR FULLY QUAL DSN
LH R3,DS01 GET DSN NAME LENGTH
STC R3,FCLDSNLG STORE LENGTH
LA R3,DS01+2 DSN=

* SEQ DS - 5 EXTENTS *

* 1 FORMAT1 AND 1 FORMAT3 *

ST R3,FCLDSNA SET ADDR OF DSN
LA R2,FCLDSNEL(R2) LOAD R2 WITH ADDR OF 2ND FCL ELEMENT
LH R3,DS02 GET DSN NAME LENGTH
STC R3,FCLDSNLG STORE LENGTH
LA R3,DS02+2 DSN=

* SEQ DS - 5 EXTENTS *

* 1 FORMAT1 AND 1 FORMAT3 *

ST R3,FCLDSNA SET ADDR OF DSN
DROP R1,R2 DROP TEMP USING
FC1EXIT DS OH EXIT FROM FCL1RTN
L R14,FCLSAVE LOAD C(FCLSAVE) INTO R14
BR R14 EXIT

FCL2RTN DS OH FCL INITIALIZATION ROUTINE
ST R14,FCLSAVE STORE C(R14) INTO SAVE AREA
XC FCLDEF(FCLSIZE),FCLDEF CLEAR FCL AREA
LA R1,FCLDEF R1 - FCL HEADER
USING FCLMAP,R1 ESTABLISH ADDRESSABILITY
LA R2,FCLHDEND R2 - FIRST (ONLY) FCL ELEMENT
USING FCLDSN,R2 ESTABLISH ADDRESSABILITY
MVC FCLID,CFLID SET THE EYECATCHER FCL
MVC FCLCOUNT,=H'1' SET NUMBER OF FCL ELEMENTS
MVI FCL1FLAG,X'80' SET FLAG FOR FULLY QUAL DSN
LH R3,DS03 GET DSN NAME LENGTH
STC R3,FCLDSNLG STORE LENGTH
LA R3,DS03+2 DSN=

* PDSE DS - 122 EXTENTS *

* 1 FORMAT1 AND 10 FORMAT3S *

ST R3,FCLDSNA SET ADDR OF DSN
DROP R1,R2 DROP TEMP USING
FC2EXIT DS OH EXIT FROM FCL2RTN
L R14,FCLSAVE LOAD C(FCLSAVE) INTO R14
BR R14 EXIT

Figure C-91  Source of CVFLT1E program (10 of 17)
Figure C-92  Source of CVFLT1E program (11 of 17)
CVAFFILT ACCESS=READ,UCB=(R3),FCL=FCLDEF,BUFLIST=BFLHDEF, EADSCB=OK, MF=(E,(R2)), ST R15,RETCODE STORE RC FOR LATER INTERROGATION CVRIEXIT DS OH EXIT FROM CVAFRD1 L R14,CVR1SAVE LOAD C(CVR1SAVE) INTO R14 BR R14 EXIT

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

* CVAFRD2 *
* - INVOKE THE CFAFFILT MACRO AND READ THE DSCBS (2 DSNS) *
CVAFFILT ACCESS=READ,UCB=(R3),FCL=FCLDEF,BUFLIST=BFLHDEF, EADSCB=OK, MF=(E,(R2)), ST R15,RETCODE STORE RC FOR LATER INTERROGATION CVRIEXIT DS OH EXIT FROM CVAFRD1 L R14,CVR1SAVE LOAD C(CVR1SAVE) INTO R14 BR R14 EXIT

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

* CVAFRDA *
* - INVOKE THE CFAFFILT MACRO AND READ ALL THE DSCBS *
CVAFFILT ACCESS=READ,UCB=UCBADD,FCL=FCLDEF,BUFLIST=BFLHDEF, EADSCB=OK, FLTAREA=KEEP,IOAREA=KEEP, MF=(E,CVPLDEFA) ST R15,RETCODE STORE RC FOR LATER INTERROGATION CVRAEXIT DS OH EXIT FROM CVAFRD1 L R14,CVRASAVE LOAD C(CVRASAVE) INTO R14 BR R14 EXIT

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

* CVAFRS *
* - INVOKE THE CFAFFILT MACRO USING RESUME *
CVAFFILT ACCESS=READ,UCB=UCBADD,FCL=FCLDEF,BUFLIST=BFLHDEF, EADSCB=OK, FLTAREA=KEEP,IOAREA=KEEP, MF=(E,CVPLDEFA) ST R15,RETCODE STORE RC FOR LATER INTERROGATION CVRAEXIT DS OH EXIT FROM CVAFRD1 L R14,CVRASAVE LOAD C(CVRASAVE) INTO R14 BR R14 EXIT

***********************************************************************
CVRS0020 DS OH PROCESS FOR RC04 - CHECK FOR STAT064 06370014
CLI CVSTAT,STAT064 IS THE CVSTAT CODE 064(RESUME NEEDED) 06380014
BNE CVRS0060 NO - BRANCH TO CVRS0060 06390014
CLI RESFLG,X'01' IS RESUME FLAG ON 06400014
BE CVRS0030 YES - BRANCH TO CVRS0030 06410014
MVC POETLINE(133),ST64MSG1 MOVE TO LINE (RESUME NEEDED) 06420014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 06430014
MVC POETLINE(133),ST64MSG2 MOVE TO LINE (INITIAL) 06440014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 06450014
BAL R14,FMTCTRNTN INVOKE FORMAT COUNT ROUTINE 06460014
BAL R14,FMTOPRTN INVOKE FORMAT OUTPUT MSG ROUTINE 06470014
MVI RESFLG,X'01' RESUME NEEDED - SET FLAG ON 06480014
B CVRS0040 BRANCH TO CVRS0040 06490014
CVRS0030 DS OH RESUME PROCESSING 06500014
MVC POETLINE(133),ST64MSG3 MOVE TO LINE (RESUME) 06510014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 06520014
BAL R14,FMTCTRNTN INVOKE FORMAT COUNT ROUTINE 06530014
BAL R14,FMTOPRTN INVOKE FORMAT OUTPUT MSG ROUTINE 06540014
CVRS0040 DS OH RESUME PROCESSING 06550014
CVAFFILT ACCESS=RESUME,MF=(E,CVPLDEFA), EADSCB=OK 06560014
X 06570014
B CVRS0000 BRANCH TO CHECK RETURN CODE AGAIN 06580014
CVRS0050 DS OH RC IS 0 NO LONGER NEED RESUME 06590014
MVC POETLINE(133),ST64MSG3 MOVE TO LINE (RESUME) 06600014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 06610014
BAL R14,FMTCTRNTN INVOKE FORMAT COUNT ROUTINE 06620014
BAL R14,FMTOPRTN INVOKE FORMAT OUTPUT MSG ROUTINE 06630014
L R6,RETF18 LOAD R6 WITH NBR OF FMT1S & FMT8S RETURNED 06640014
CVD R6,WF18 CONVERT TO DEC FOR PRINTING 06650014
UNPK ST64MSG4+59(7),WF18 UNPACK TO FORMAT LINE 06660014
OI ST64MSG4+65,X'F0' SET APPROPRIATE BITS 06670014
MVC POETLINE(133),ST64MSG4 MOVE TO LINE (COMPLETE) 06680014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 06690014
* 06700014
* RELEASE WORK AREAS 06710014
* 06720014
CVAFFILT ACCESS=RLSE,FLTAREA=NOKEEP,IOAREA=NOKEEP, EADSCB=OK, 06730014
MF=(E,CVPLDEFA) 06740014
X 06750014
B CVRSEXIT BRANCH TO EXIT ROUTINE 06760014
CVRS0060 DS OH RC IS 4 BUT CVSTAT IS NOT 064 06770014
MVC POETLINE(133),RC04MSG MOVE TO LINE 06780014
DC H'0' ABEND 06790014
PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 06800014
CVRSEXIT DS OH EXIT FROM CVAFRS 06810014
L R14,CVRSSAVE LOAD R14 INTO CVRSSAVE 06820014
BR R14 EXIT 06830014
*********************************************************************** 06840014
* CVAFRL * 06850014
* - INVOKE THE CVAFFILT MACRO AND RELEASE WORK AREAS * 06860014
CVAFLR DS OH CVAFFILT - RLSE ROUTINE 06870014
ST R14,CVRSSAVE STORE C(R14) INTO SAVE AREA 06880014

Figure C-94  Source of CVFLT1E program (13 of 17)
CVAFFILT ACCESS=RLSE,FCL=0,BUFLIST=0,FLTAREA=NOKEEP, X06890014
   EADSCB=OK, X06900014
   MF=(E,CVPLDEF) 06910014
   CH R15,RCODE00 IF RC = 0 THEN 06920014
   BE CVRLEXIT BRANCH TO EXIT 06930014
   MVC POETLINE(133),RLSEMSG ELSE MOVE MSG TO LINE 06940014
   PUT OUTFILE,POETLINE WRITE A RECORD TO THE OUTPUT FILE 06950014
   CVRLEXIT DS OH EXIT FROM CVAFRL 06960014
   L R14,CVRLSAVE LOAD C(CVRLSAVE) INTO R14 06970014
   BR R14 EXIT 06980014
*********************************************************************** 06990014
* WORKING STORAGE * 07000014
DS OD 07010014
   DC CL36'CVFLT1E-WORKING STORAGE BEGINS HERE' 07020014
*********************************************************************** 07030014
* EQUATES * 07040014
   EABN100 EQU 100 USER ABEND CODE 100 - VTOC OPEN ERROR 07050014
   EABN101 EQU 101 USER ABEND CODE 101 - OUTDD OPEN ERROR 07060014
   EABN102 EQU 102 USER ABEND CODE 102 - OUTDD CLOSE ERROR 07070014
   BUFNBR EQU 11 11 BUFFERS TO BE USED 07080014
   R0 EQU 0 07090014
   R1 EQU 1 07100014
   RDEB EQU 1 REG1 FOR DEB ADDRESS 07110014
   R2 EQU 2 07120014
   RUCB EQU 2 REG2 FOR UCB ADDRESS 07130014
   R3 EQU 3 07140014
   R4 EQU 4 07150014
   R5 EQU 5 07160014
   R6 EQU 6 07170014
   R7 EQU 7 07180014
   R8 EQU 8 07190014
   R9 EQU 9 07200014
   R11 EQU 11 07210014
   R12 EQU 12 07220014
   R13 EQU 13 07230014
   R14 EQU 14 07240014
   R15 EQU 15 07250014
   * 07260014
*********************************************************************** 07270014
* SAVE AREAS * 07280014
   SAVE DC 18F'0' MAIN PROGRAM SAVE AREA 07290014
   OPENSAVE DC F'0' OPEN FILES ROUTINE SAVE AREA 07300014
   CLOSSAVE DC F'0' CLOSE FILES ROUTINE SAVE AREA 07310014
   ABENSAVE DC F'0' ABEND ROUTINE SAVE AREA 07320014
   IDLVSSAVE DC F'0' IDENTIFY VOLUME ROUTINE SAVE AREA 07330014
   BUFLSAVE DC F'0' INITIALIZE BUFFER ROUTINE SAVE AREA 07340014
   FCL1SAVE DC F'0' INITIALIZE FCL1 ROUTINE SAVE AREA 07350014
   FCL2SAVE DC F'0' INITIALIZE FCL2 ROUTINE SAVE AREA 07360014
   FCL3SAVE DC F'0' INITIALIZE FCL3 ROUTINE SAVE AREA 07370014
   FCL4SAVE DC F'0' INITIALIZE FCL4 ROUTINE SAVE AREA 07380014
   TSTRTSAVE DC F'0' TEST RETURN CODE ROUTINE SAVE AREA 07390014

Figure C-95  Source of CVFLT1E program (14 of 17)
CVRISAVE DC F'0' CVAFFILT READ 1 DSN ROUTINE SAVE AREA 07400014
CVR2SAVE DC F'0' CVAFFILT READ 2 DSN ROUTINE SAVE AREA 07410014
CVRASAVE DC F'0' CVAFFILT READ ALL ROUTINE SAVE AREA 07420014
CVRLSAVE DC F'0' CVAFFILT RLSE ROUTINE SAVE AREA 07430014
CVRSSAVE DC F'0' CVAFFILT RESUME ROUTINE SAVE AREA 07440014
FMTCSAVE DC F'0' FORMAT DSCB COUNT ROUTINE SAVE AREA 07450014
FMTOSAVE DC F'0' FORMAT OUTPUT ROUTINE SAVE AREA 07460014
*********************************************************************** 07470014
* CONSTANTS * 07480014
RETCODE  DC F'999' 07490014
RCODE00  DC H'0' RETURN CODE 0 - HALFWORD 07500014
RCODE04  DC H'4' RETURN CODE 4 - HALFWORD 07510014
DSNNBR   DC X'FF' INDICATE NBR OF DSNS TO PROCESS 07520014
RESFLG   DC X'00' RESUME FLAG - OFF 07530014
BLNKLNE  DC CL133' ' 07540014
STRTMSG  DC CL133' CVFLT1E START OF OUTPUT MESSAGES' 07550014
VOLTEXT  ORG STRTMSG+35 07560014
DC C'VOLUME: ' 07570014
VOLSER   DC CL6' ' 07580014
ORG 07590014
ENDMSG   DC CL133' CVFLT1E END OF OUTPUT MESSAGES' 07600014
ST64MSG1 DC CL133' RC04 VERIFIED - CVSTAT 064 RESUME IS NECESSARY' 07610014
ST64MSG2 DS OCL133 07620014
DC CL49' CVAFFILT (INITIAL) RETURNED THE FOLLOWING DSCBS' 07630014
DC CLB4'FOR THE VOLUME:' 07640014
ST64MSG3 DS OCL133 07650014
DC CL49' CVAFFILT (RESUME) RETURNED THE FOLLOWING DSCBS' 07660014
DC CLB4'FOR THE VOLUME:' 07670014
ST64MSG4 DS OCL133 07680014
DC CL48' CVAFFILT RESUME OPERATION COMPLETE - ALL DSCBS' 07690014
DC CLB5'RETURNED (         )' 07700014
DSCBMSGA DS OCL133 07710014
DC CL48' CVAFFILT RETURNED THE FOLLOWING DSCBS FOR DSN:' 07720014
DS DC CL44' ' 07730014
DC CL41' ' 07740014
DSCBMSGB DS OCL133 07750014
DC CL48' AND FOR DSN:' 07760014
DS2 DC CL44' ' 07770014
DC CL41' ' 07780014
DSCBMSGC DS OCL133 07790014
DC CL48' CVAFFILT LOGICAL ERROR STATUS RETURNED - DSN:' 07800014
DS3 DC CL44' ' 07810014
DC CL41' ' 07820014
NODSCBM DC CL133' NO DSCBS RETURNED FROM CVAFFILT' 07830014
RCO0MSG DC CL133' RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL' 07840014
RCO4MSG DC CL133' RC04 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT' 07850014
RCERMSG DC CL133' RC08, RC12, OR RC16 RETURNED FROM CVAFFILT' 07860014
RLSEM SG DC CL133' NON ZERO RETURN CODE BACK FROM RLSE' 07870014
DSCBMSG1 DC CL29' NUMBER OF FORMAT 1 DSCBS -' 07880014
DSCBMSG2 DC CL29' NUMBER OF FORMAT 3 DSCBS -' 07890014
DSCBMSG3 DC CL29' NUMBER OF FORMAT 8 DSCBS -' 07900014
DSCBMSG4 DC CL29' NUMBER OF FORMAT 9 DSCBS -' 07910014
DS01 DC H'17',CL44'MHLRES1.EAV.PDS01' 07920014
DS02 DC H'19',CL44'MHLEAV.EXTKSDS.DATA' 07930014
DS03 DC H'18',CL44'MHLRES1.EAV.PDSE01' 07940014
CFCLID DC CL4'FCL' 07950014

Figure C-96  Source of CVFLT1E program (15 of 17)
Figure C-97  Source of CVFLT1E program (16 of 17)
**C.8.1.1 Execution of sample CVFLT1E program**

The CVFLT1E program is a working example. If you want to run the program, you need to assemble and link it. It is not necessary to have in depth assembler experience, but familiarity with JCL is required.

The source is stored in one PDS, the JCL in one PDS, and the LOAD modules in another.

To create the assembler PDS, use the following steps.

There are three steps to build the program which needs to be done once, after which it can be executed several times.

**Step 1: Create a PDS/PDSE to hold the source members**

In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

**Step 2: Create a PDS/PDSE to hold the LOAD MODULES**

In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 0/U. All other attributes can be chosen by you.

**Step 3: Store the program source in the PDS**

Make sure that your ISPF EDIT menu settings has the screen format defined as DATA, the emulator line length is set to greater than 80, and that the edit session is in NONUM mode.

- Cut and paste the contents of Figure C-82 on page 581 through to Figure C-98 on page 597 one after the other into member CVFLT1E. The result should contain 872 lines.

**Step 4: Create a PDS/PDSE to hold the JCL members**

In this example the PDS is called MHLRES1.EAV.JCL.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.
Step 5: Store the JCL source in the PDS

- Cut and paste the contents of Figure C-99 into member CVFLT1E. The result should contain 30 lines.

```ml
//MHLRES1F JOB (1234567,COMMENT),UAALFO,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*JOBPARM S=* 
//ASMHCL PROC
//ASM EXEC PGM=ASMA90,REGION=0M,
// PARM='OBJECT,NODECK'
//SYSLIN DD DSN=&OBJ,DISP=(NEW,PASS),UNIT=SYSDA,
// SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120
//SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB
// DD DISP=SHR,DSN=SYS1.MODGEN
// DD DISP=SHR,DSN=MHLRES1.EAV.ASM
//SYSPRINT DD SYSOUT=* 
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5)) */
//LKED EXEC PGM=HEWL,REGION=2048K,COND=(8,LE,ASM), 
// PARM='XREF,LIST,LET'
//SYSLIN DD DSN=&OBJ,DISP=(OLD,DELETE)
// DD DDNAME=SYSIN
//SYSPRINT DD SYSOUT=* 
//SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
// PEND 
// EXEC ASMHCL
//ASM.SYSIN DD DISP=SHR,DSN=MHLRES1.EAV.ASM(CVFLT1E)
/* 
//LKED.SYSLMOD DD DSN=MHLRES1.EAV.LOAD,DISP=SHR
//LKED.SYSIN DD * 
SETCODE AC(1) 
NAME CVFLT1E(R) 
```

Figure C-99  JCL to assemble and link CVFLT1E

Step 7: Run the job in member CVFLT1E

Check that the return code from the ASM and LKED steps is 0.

A return code of 8 on the LKED step is an indication that the subroutine required has not been stored in MHLRES1.EAV.LOAD.

Step 6: APF authoress the data set CVFLT1E has been linked into

- Add the data set to the PARMLIB APF member (PROGxx) and IPL
- Or dynamically APF authoress the data set. Assuming it is MHLRES1.EAV.LOAD and that it is on volume MHLSE1, issue the command
  - setprog apf,add,dsname=MHLRES1.EAV.LOAD, volume=MHLSE1

The CVFLT1E program is now ready for execution. Use the JCL as shown in Figure C-74 on page 573.
C.9 VTOC management macros extended for EAV - CVAFSEQ

The CVAFSEQ macro, with its underlying services is provided to obtain information about the data sets on an indexed or non indexed VTOC. CVAFSEQ itself is not new for EAV support but it has been extended for EAV.

The CVAFSEQ macro as implemented in z/OS releases prior to z/OS V1R10, and if not extended by EAV keywords is not permitted to retrieve information from EAV formatted volumes. Use of an un-extended CVAFSEQ macro call against a VTOC on an EAV volume will result in a non zero return code, a non-zero status byte and the data requested will not be returned.

Changes to a program issuing the CCAFSEQ macro call are required to allow access to EAV volumes. Existing programs without the EAV enabling operand on the CVAFSEQ call will be tracked by the Migration Assistance Tracker but will not fail if run against a non-EAV volume. Such programs will fail if run against an EAV volume.

We have provided an example of a program that issues the CVAFSEQ macro in the format used prior to z/OS 1.10, and as extended to support EAV to illustrate the type of change required to support EAV. The program uses a table of 20 entries, so does not list all the data sets on a volume (unless there are fewer than 20).

The purpose of this example program is to:

- Demonstrate what happens when an enhanced macro call is issued against a non-EAV volume
- Demonstrate what happens when an enhanced macro call is issued against an EAV volume
- Demonstrate what happens when an un-enhanced macro call is issued against a non-EAV volume
- Demonstrate what happens when an un-enhanced macro call is issued against an EAV volume
- Demonstrate that the Migration Assistance Tracker picks up the CVAFSEQ un-enhanced macro against a non-EAV volume
- Illustrate the code changes required to support EAV format volumes.

In Figure C-100 we show the JCL to examine volumes MLDC65 and SBOX1T.

```
//MHLRES1Q JOB (1234567,COMMENT),UAALFO,TIME=10, // MSGLEVEL=1,CLASS=A, // NOTIFY=&SYSUID /*JOBPARM S=* //CVSEQ80 PROC //RUN EXEC PGM=CVSEQ80,REGION=0M //STEPLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD //CVAFDD DD DISP=SHR,UNIT=3390,VOL=SER=MLDC65 //SYSUDUMP DD SYSOUT=* //OUTDD DD SYSOUT=* //PEND // EXEC CVSEQ80 // EXEC CVSEQ80 //CVAFDD DD DISP=SHR,UNIT=3390,VOL=SER=SBOX1T
```

Figure C-100 Example of JOB to examine two volumes using CVSEQ80
In Figure C-101 we show the result of running the JCL to execute program CVSEQ8O on volume MLDC65 (EAV).

<table>
<thead>
<tr>
<th>CVSEQ8O START OF OUTPUT MESSAGES VOLUME: MLDC65</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVAFSEQ CALL: EADSCB KEYWORD NOT CODED</td>
</tr>
<tr>
<td>CV4EADOK BIT IS NOT SET / EADSCB=NOTOK</td>
</tr>
<tr>
<td>RC04 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT</td>
</tr>
<tr>
<td>CVSTAT CODE: X&quot;52&quot;    DEC&quot;082&quot;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CVAFSEQ CALL: EADSCB=NOTOK CODED</td>
</tr>
<tr>
<td>CV4EADOK BIT IS NOT SET / EADSCB=NOTOK</td>
</tr>
<tr>
<td>RC04 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT</td>
</tr>
<tr>
<td>CVSTAT CODE: X&quot;52&quot;    DEC&quot;082&quot;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CVAFSEQ CALL: EADSCB=OK CODED</td>
</tr>
<tr>
<td>CV4EADOK BIT SET / EADSCB=OK</td>
</tr>
<tr>
<td>RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL</td>
</tr>
<tr>
<td>CVSTAT CODE: X&quot;00&quot;    DEC&quot;000&quot;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>DATASETS ON THE VOLUME LISTED IN PHYSICAL SEQUENTIAL ORDER:</td>
</tr>
<tr>
<td>DSN: SYS1.VTOCIX.MLDC65</td>
</tr>
<tr>
<td>DSN: SYS1.VVDS.VMLDC65</td>
</tr>
<tr>
<td>DSN: MHLEAV.EXTKSDS.DATA</td>
</tr>
<tr>
<td>DSN: MHLEAV.EXTKSDS.IX</td>
</tr>
<tr>
<td>DSN: MHLEAV.EDSBIG02.DATA</td>
</tr>
<tr>
<td>DSN: MHLEAV.EDSBIG02.INDEX</td>
</tr>
<tr>
<td>END OF DATA REACHED - ALL DATASETS PROCESSED</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>CVSEQ8O END OF OUTPUT MESSAGES</td>
</tr>
</tbody>
</table>

*Figure C-101  CVSEQ8O output from sample job - volume MLDC65 (1 of 2)*

This output shows the following results:

- These messages relate to volume MLDC65 (EAV).
  - CVAFSEQ CALL: EADSCB=NOTOK IS THE DEFAULT
    - Result is FAILURE with CVSTAT X"52" (DEC(82))
  - CVAFSEQ CALL: EADSCB=NOTOK IS CODED (same as the default)
    - Result is FAILURE with CVSTAT X"52" (DEC(82))
  - CVAFSEQ CALL: EADSCB=OK KEYWORD CODED
    - Result is SUCCESS with CVSTAT X"00" (DEC(00))
    - Data sets on the volume listed. The table of 20 DSCBs is used to hold all the DSCBs returned, but only the Format-1 and Format-8 entries are listed so the list might show less than 20 entries.
In Figure C-102 we show the result of running the JCL to execute program CVSEQ8O on volume SBOX1T (non-EAV).

<table>
<thead>
<tr>
<th>CVSEQ8O START OF OUTPUT MESSAGES VOLUME: SBOX1T</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVAFSEQ CALL: EADSCB KEYWORD NOT CODED</td>
</tr>
<tr>
<td>CV4EADOK BIT IS NOT SET / EADSCB=NOTOK</td>
</tr>
<tr>
<td>RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL</td>
</tr>
<tr>
<td>CVSTAT CODE: X&quot;00&quot;    DEC&quot;000&quot;</td>
</tr>
</tbody>
</table>

| CVAFSEQ CALL: EADSCB=NOTOK CODED              |
| CV4EADOK BIT IS NOT SET / EADSCB=NOTOK        |
| RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL    |
| CVSTAT CODE: X"00"    DEC"000"                |

| CVAFSEQ CALL: EADSCB=OK CODED                 |
| CV4EADOK BIT SET / EADSCB=OK                  |
| RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL    |
| CVSTAT CODE: X"00"    DEC"000"                |

<table>
<thead>
<tr>
<th>DATASETS ON THE VOLUME LISTED IN PHYSICAL SEQUENTIAL ORDER:</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSN: SYS1.VTOCIX.SBOX1T</td>
</tr>
<tr>
<td>DSN: PAOLOR9.COPY.G0006V00</td>
</tr>
<tr>
<td>DSN: PAOLOR9.COPY.G0011V00</td>
</tr>
<tr>
<td>DSN: DBBAU.ARCHLOG1.A0000101</td>
</tr>
<tr>
<td>DSN: DBBAU.ARCHLOG2.B0000102</td>
</tr>
<tr>
<td>DSN: DONNAS.IOCP.P1</td>
</tr>
<tr>
<td>DSN: HSMACT.H3.MIGLOG.D06251.T182054</td>
</tr>
<tr>
<td>DSN: PAOLOR9.COPY.G0012V00</td>
</tr>
<tr>
<td>DSN: PAOLOR9.ROY33</td>
</tr>
<tr>
<td>DSN: HERING.PDF1681.BACKUP</td>
</tr>
<tr>
<td>DSN: DBBFU.DBF1.ARCLG1.A0000017</td>
</tr>
<tr>
<td>DSN: DBBFU.ARCHLOG1.D07051.T1214245.A0000020</td>
</tr>
<tr>
<td>DSN: KMT1.PMR69638.VIDALIA</td>
</tr>
<tr>
<td>DSN: FISCHER.SC64.SPFTEMP1.CNTL</td>
</tr>
<tr>
<td>DSN: CASSIER.RMFZOS18.ADMGDF</td>
</tr>
<tr>
<td>DSN: PAOLOR2.NORMENO0.UNLOAD.ORIG.BIN</td>
</tr>
<tr>
<td>DSN: KJCCRAIG.IOFD01.PMR44844</td>
</tr>
<tr>
<td>DSN: ROWLEY.SC63.SPFLOG1.LIST</td>
</tr>
<tr>
<td>DSN: DBBFU.DBF1.ARCLG1.A0000014</td>
</tr>
<tr>
<td>END OF DATA REACHED - ALL DATASETS PROCESSED</td>
</tr>
</tbody>
</table>

| CVSEQ8O END OF OUTPUT MESSAGES                       |

*Figure C-102  CVSEQ8O output from sample job - Volume SBOX1T (2 of 2)*
These messages relate to volume SBOX1T (non-EAV).

- CVAFSEQ CALL: EADSCB=NOTOK IS THE DEFAULT
  - Result is SUCCESS with CVSTAT X"00" (DEC(00))
- CVAFSEQ CALL: EADSCB=NOTOK IS CODED (same as the default)
  - Result is SUCCESS with CVSTAT X"00" (DEC(00))
- CVAFSEQ CALL: EADSCB=OK KEYWORD CODED
  - Result is SUCCESS with CVSTAT X"00" (DEC(00))
  - Data sets on the volume are listed. The table of 20 DSCBs is used to hold all the DSCBs returned, but only the Format-1 and Format-8 entries are listed so the list might show less than 20 entries.

C.9.1 EAV Migration Assistance Tracker considerations: CVAFSEQ

The EAV Migration Assistance Tracker will detect use of the CVSEQ service that has not been updated for EAV compatibility.

Any appearance on the tracker list must be investigated and resolved because use of an un-extended OBTAIN against an EAV volume in due course will fail.

For information about the EAV Migration Tracker, refer to Appendix C of DFSMS Advanced Services, SC26-7400, and in this document to “Migration Assistance Tracker” on page 641.

In Figure C-103 we show the output from the Migration Assistance Tracker that relates to the run of CVSEQ8O run against volume SBOX1T. It shows that CVSEQ8O needs attention before it is used against an EAV format volume.

<table>
<thead>
<tr>
<th>Source code listing</th>
</tr>
</thead>
<tbody>
<tr>
<td>In Figure C-104 on page 603 through to Figure C-117 on page 616 we show assembler source for a program to issue the CVAFSEQ macro.</td>
</tr>
<tr>
<td>In this source, the following coding relates to EAV support:</td>
</tr>
<tr>
<td>▶ In routine CVAFSQ3, which starts at line 05640007:</td>
</tr>
<tr>
<td>▶ In routine CVAFSQ1, which starts at line 04950007:</td>
</tr>
<tr>
<td>▶ In routine CVAFSQ2, which starts at line 05290007:</td>
</tr>
<tr>
<td>▶ In order to set up the CVAFSEQ calls, the VTOC was opened and read directly. In order to allow a VTOC on an EAV volume to be READ, the DCB coded for that purpose had to be extended by the addition of the DCBE macro which is used to specify EADDDB=OK:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Figure C-103 CVSEQ8O Migration Assistance Tracker entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS-E:1 CVAFSEQ STAT082</td>
</tr>
<tr>
<td>45201 MHLRES1Q CVSEQ8O 480 42 1</td>
</tr>
</tbody>
</table>
CVSEQ80 TITLE 'CVAF CVAFSEQ TEST MODULE'
CVSEQ80 CSECT
CVSEQ80 AMODE 31
CVSEQ80 RMODE 24
***********************************************************************
* CVSEQ80 - MODULE THAT ISSUES THE CVAFSEQ MACRO AND PROCESS A      *
* A VOLUME TO RETURN DSCBS, 20 AT A TIME, IN PHYSICAL                *
* SEQUENTIAL ORDER USING A STARTING CHHR OF ZERO.                    *
* THE CVAFSEQ MACRO CALL WILL BE ISSUED THREE TIMES                  *
* USING THE FOLLOWING EADSCB KEYWORD SETTINGS:                       *
* - EADSCB KEYWORD NOT CODED (DEFAULTS TO EADSCB=NOTOK)              *
* - EADSCB=NOTOK CODED                                               *
* - EADSCB=OK CODED                                                  *
* CV4EADOK BIT SETTING WILL BE DETERMINED FOR EACH CALL.*            *
* FOR A NON EAV VOLUME:                                              *
* WILL EXPECT ALL DSCBS ON THE VOLUME RETURNED IN THE                *
* ORDER THE DATASETS WERE CREATED.                                   *
* FOR A EAV VOLUME:                                                  *
* WILL EXPECT RCO4 / STAT082 RETURNED FROM CVAFSEQ CALL              *
* WHEN THE EADSCB KEYWORD IS NOT CODED OR EADSCB=NOTOK IS CODED.     *
* WILL EXPECT ALL DSCBS ON THE VOLUME RETURNED IN THE                *
* ORDER THE DATASETS WERE CREATED WHEN EADSCB=OK IS CODED.          *
* THIS PROGRAM WILL CREATE AN OUTPUT REPORT THAT                    *
* SHOULD BE SIMILAR TO THE EXAMPLES LISTED BELOW.                   *
*---------------------------------------------------------------------*
* EXAMPLE 1: NON EAV VOLUME WITH OS VTOC                            *
* --------------------------------------                            *
* CVAFSEQ CALL: EADSCB KEYWORD NOT CODED                             *
* CV4EADOK BIT IS NOT SET / EADSCB=NOTOK                             *
* RCO0 VERIFIED - THE REQUEST WAS SUCCESSFUL                         *
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED                           *
* CVAFSEQ CALL: EADSCB=NOTOK CODED                                    *
* CV4EADOK BIT IS NOT SET / EADSCB=NOTOK                             *
* RCO0 VERIFIED - THE REQUEST WAS SUCCESSFUL                         *
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED                           *
* CVAFSEQ CALL: EADSCB=OK CODED                                      *
* CV4EADOK BIT SET / EADSCB=OK                                       *
* RCO0 VERIFIED - THE REQUEST WAS SUCCESSFUL                         *
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED                           *
* DATASETS ON THE VOLUME LISTED IN PHYSICAL SEQUENTIAL ORDER:        *
* DSN: CVS5MSC2.SEQ01                                                 *
* DSN: CVS5MSC2.PDS01                                                 *
* DSN: CVS5MSC2.VSAM01.DATA 00570007
* DSN: SYS1.VD5.V1P9503 00580007
* DSN: CVS5MSC2.PDSE01 00590007
* DSN: CVS5MSC2.SEQ02 00600007
* DSN: CVS5MSC2.VSAM02.DATA 00610007
* DSN: CVS5MSC2.PDSE02 00620007
* END OF DATA REACHED - ALL DATASETS PROCESSED 00630007
* 00640007
* CVSEQ8O END OF OUTPUT MESSAGES 00650007
* 00660007
* EXAMPLE 2: EAV VOLUME WITH OS VTOC 00670007
* 00680007
* 00690007
* CVSEQ8O START OF OUTPUT MESSAGES 00700007
* 00710007
* CVAFSEQ CALL: EADSCB KEYWORD NOT CODED 00720007
* CV4EADOK BIT IS NOT SET / EADSCB=NOTOK 00730007
* RC04 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT 00740007
* X"52" DEC"082" 52 - CVSTAT CODE VERIFIED 00750007
* 00760007
* CVAFSEQ CALL: EADSCB=NOTOK CODED 00770007
* CV4EADOK BIT IS NOT SET / EADSCB=NOTOK 00780007
* RC04 VERIFIED - LOGICAL ERROR STATUS IN CVSTAT 00790007
* X"52" DEC"082" 52 - CVSTAT CODE VERIFIED 00800007
* 00810007
* CVAFSEQ CALL: EADSCB=OK CODED 00820007
* CV4EADOK BIT SET / EADSCB=OK 00830007
* RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL 00840007
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED 00850007
* 00860007
* DATASETS ON THE VOLUME LISTED IN PHYSICAL SEQUENTIAL ORDER: 00870007
* DSN: CVS5MSC2.SEQ01 00880007
* DSN: CVS5MSC2.PDS01 00890007
* DSN: CVS5MSC2.VSAM01.DATA 00900007
* DSN: SYS1.VD5.V1P9503 00910007
* DSN: CVS5MSC2.PDSE01 00920007
* DSN: CVS5MSC2.SEQ02 00930007
* DSN: CVS5MSC2.VSAM02.DATA 00940007
* DSN: CVS5MSC2.PDSE02 00950007
* END OF DATA REACHED - ALL DATASETS PROCESSED 00960007
* 00970007
* CVSEQ8O END OF OUTPUT MESSAGES 00980007
* 00990007
* EXAMPLE 3: NON EAV VOLUME WITH IX VTOC 01000007
* 01010007
* 01020007
* CVSEQ8O START OF OUTPUT MESSAGES 01030007
* 01040007
* CVAFSEQ CALL: EADSCB KEYWORD NOT CODED 01050007
* CV4EADOK BIT IS NOT SET / EADSCB=NOTOK 01060007
* RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL 01070007
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED 01080007
* 01090007
* CVAFSEQ CALL: EADSCB=NOTOK CODED 01100007
* CV4EADOK BIT IS NOT SET / EADSCB=NOTOK 01110007
* RC00 VERIFIED - THE REQUEST WAS SUCCESSFUL 01120007
* X"00" DEC"000" 00 - CVSTAT CODE VERIFIED 01130007

Figure C-105  Source of CVSEQ8O program (2 of 15)
Figure C-106   Source of CVSEQ8O program (3 of 15)
NOTE: THIS MODULE REQUIRES THE USE OF THE CVSTCHK MODULE WHICH IS CALLED TO INTERPRET THE CVSTAT CODE RETURNED.

INITIALIZATION
- OBTAIN THE NECESSARY INFORMATION FROM THE DASD VOLUME
- OPEN THE OUTPUT FILE AND WRITE NECESSARY OUTPUT MESSAGES
- SET STARTING CCHHR TO ZERO
- SET END OF DATA SWITCH TO NO END OF DATA

MAINLINE
- LOAD THE TABLE WITH DSCB ADDRESSES TO USE FOR CVAFSEQ CALL
- INITIALIZE BUFFER LIST
- INVOKE CVAFSQ1 ROUTINE - EADSCB KEYWORD NOT CODED
  - REPORT ON RETURN CODE AND CVSTAT CODE RETURNED FROM CALL
- INITIALIZE BUFFER LIST
- INVOKE CVAFSQ2 ROUTINE - EADSCB=NOTOK CODED
  - REPORT ON RETURN CODE AND CVSTAT CODE RETURNED FROM CALL
- INITIALIZE BUFFER LIST
- INVOKE CVAFSQ3 ROUTINE - EADSCB=OK CODED
  - REPORT ON RETURN CODE AND CVSTAT CODE RETURNED FROM CALL
  - DO WHILE MORE TABLE ENTRIES TO PROCESS
    - INVOKE PRTBRTN ROUTINE TO PROCESS DSCBS RETURNED FROM CVAFSEQ
      - PRINT OUT ALL DSNAMES ON THE VOLUME IN SEQUENTIAL ORDER
- FINALIZATION
  - WRITE NECESSARY MESSAGES AND CLOSE THE OUTPUT FILE AND EXIT

CVSEQ8O - JOB INFORMATION
- RC=00 AND OUTDD OUTPUT AS DETAILED ABOVE
- ABNORMAL END OF JOB:
  - ABEND 100 - ERROR OPENING VTOC ON THE DASD VOLUME THAT IS ASSOCIATED WITH THE CVAFDD DD STATEMENT
  - ABEND 101 - ERROR OPENING THE OUTDD DATASET
  - ABEND 102 - ERROR CLOSING THE OUTDD DATASET

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

* HOUSEKEEPING
- SAVE CALLER'S REGISTERS AND ESTABLISH A NEW REGISTER SAVE AREA
- STANDARD LINKAGE CONVENTION
- DCL R11 AS IMPLIED BASE REG
- USING BASE,R11,R12
- R12 IS ALSO BASE REG
- SET UP ADDRESSING FOR R12
- BRANCH AROUND DECLARES
- ADDRESSING FOR R12

Figure C-107   Source of CVSEQ8O program (4 of 15)
Appendix C. Code samples DFSMS V1.10

Figure C-108   Source of CVSEQ80 program (5 of 15)

```
CV000000 DS OH CONTINUE... 02230007
ST R13,SAVE+4 SAVE PTR TO CALLER'S SAVE AREA 02240007
LA R14,SAVE GET ADDRESS OF THE NEW SAVE AREA 02250007
ST R14,(R13) CHAIN CALLER'S AREA TO OURS 02260007
LR R13,R14 ESTABLISH THE NEW SAVE AREA 02270007
*
02280007
*********************************************************************** 02290007
* INITIALIZATION 02300007
*
INITIAL DS OH INITIALIZATION SECTION 02330007
BAL R14,IDVOLRTN INVOKE RTN TO IDENTIFY THE VOLUME(S) 02340007
OPEN (OUTFILE,(OUTPUT)) OPEN THE OUTPUT MESSAGE FILE 02350007
TM OUTFILE+48,X'10' IF OPEN OF OUTPUT FILE NOT OK 02360007
BO OK1 ISSUE USER ABEND 101 02380007
OK1 DS OH 02390007
L R1,UCBADD 02400007
MVC VOLSER(6),28(R1) 02410007
PUT OUTFILE,STRMTMSG WRITE A RECORD TO THE OUTPUT FILE 02420007
PUT OUTFILE,BLNKLINE WRITE A RECORD TO THE OUTPUT FILE 02430007
MVC CCHHRS,CCHHR0 INIT CCHHR START TO ZERO 02440007
MVI SWEOD,NOEOD SET SWITCH TO NO END OF DATA 02450007
*
02460007
*********************************************************************** 02470007
* MAINLINE 02480007
*
MAINLINE DS OH MAINLINE SECTION 02510007
BAL R14,LDTABRTN INVOKE LDTABRTN TO LOAD TABLE 02520007
BAL R14,INITBRTN INVOKE INITBRTN TO INIT BUFF LIST 02530007
BAL R14,CVAFSQ1 INVOKE CVAFSQ1 TO ISSUE CVAFSEQ 02540007
PUT OUTFILE,BLNKLINE WRITE A RECORD TO THE OUTPUT FILE 02550007
BAL R14,INITBRTN INVOKE INITBRTN TO INIT BUFF LIST 02560007
BAL R14,CVAFSQ2 INVOKE CVAFSQ2 TO ISSUE CVAFSEQ 02570007
PUT OUTFILE,BLNKLINE WRITE THE REC TO OUTPUT FILE 02580007
BAL R14,INITBRTN INVOKE INITBRTN TO INIT BUFF LIST 02590007
BAL R14,CVAFSQ3 INVOKE CVAFSQ3 TO ISSUE CVAFSEQ 02600007
PUT OUTFILE,BLNKLINE WRITE THE REC TO OUTPUT FILE 02610007
PUT OUTFILE,MSG0 WRITE THE REC TO OUTPUT FILE 02620007
*
02630007
CLI SWEOD,EOD DOWHILE TABLE DATA TO PROCESS 02640007
BE TABDONE 02650007
BAL R14,PRTBRTN INVOKE PRTBRTN TO PROCESS TBL 02660007
*
02670007
TABDONE DS OH 02680007
PUT OUTFILE,EODMSG WRITE THE REC TO OUTPUT FILE 02690007
*
02700007
*********************************************************************** 02710007
* FINALIZATION 02720007
*
FINAL DS OH FINALIZATION SECTION 02750007
PUT OUTFILE,BLNKLINE WRITE A RECORD TO THE OUTPUT FILE 02760007
PUT OUTFILE,ENDMSG WRITE A RECORD TO THE OUTPUT FILE 02770007
```

Figure C-108 Source of CVSEQ80 program (5 of 15)
CLOSE (OUTFILE) CLOSE OUTPUT FILE
C R15,RCODE00 IF FILE CLOSE IS NOT OK
BE OK2
ABEND 102 ISSUE USER ABEND 102
OK2 DS OH
L R13,4(R13) RESTORE REGISTER
LM R14,R12,12(R13) RESTORE CALLERS REGISTERS
LA R15,0 SET RC TO 0
BR R14 RETURN TO CALLER

***********************************************************************
*                          IDVOLRTN                                   *
*     - OBTAIN THE NECESSARY INFORMATION FROM THE DASD VOLUME         *
IDVOLRTN DS OH IDENTIFY VOLUME ROUTINE
ST R14,IDVLSAVE STORE C(R14) INTO SAVE AREA
RDJFCB (VTOCDCB,(INPUT)) READ JFCB / OPEN VTOC
MVI JFCB1,X'04' PUT IN ID FOR FORMAT 4
MVC JFCB1+1(43),JFCB1 SETUP FOR VTOC OPEN
OPEN (VTOCDCB,(INPUT)),TYPE=J OPEN VTOC (OPEN TYPE=J)
TM VTOCDCB+48,X'10' IF OPEN OF VTOC NOT OK
BO OK3
ABEND 100 ISSUE USER ABEND 100
OK3 DS OH
SLR R3,R3 INIT R3 FOR DEB_PTR
SLR R4,R4 INIT R4 FOR UCB_PTR
ICM R3,B'0111',VTOCDCB+45 GET DEB ADDRESS
ICM R4,B'0111',33(R3) GET UCB ADDRESS
ST R3,DEBADD SAVE DEB ADDRESS
ST R4,UCBADD STORE UCB ADDRESS
IDVLEXIT DS OH EXIT FROM IDVOLRTN
L R14,IDVLSAVE LOAD C(IDVLSAVE) INTO R14
BR R14 EXIT

***********************************************************************
*                          LDTABRTN                                   *
*  - LOAD 20 ENTRY TABLE WITH DSCB ADDRESSES TO USE                   *
LDTABRTN DS OH LOAD TABLE ROUTINE
ST R14,LDTBSAVE STORE C(R14) INTO SAVE AREA
LA R4,DSCB01 LOAD R4 WITH ADDRESS OF DSCB01
ST R4,TOSCB01 STORE ADDRESS OF DSCB01 INTO TABLE
LA R4,DSCB02 LOAD R4 WITH ADDRESS OF DSCB02
ST R4,TOSCB02 STORE ADDRESS OF DSCB02 INTO TABLE
LA R4,DSCB03 LOAD R4 WITH ADDRESS OF DSCB03
ST R4,TOSCB03 STORE ADDRESS OF DSCB03 INTO TABLE
LA R4,DSCB04 LOAD R4 WITH ADDRESS OF DSCB04
ST R4,TOSCB04 STORE ADDRESS OF DSCB04 INTO TABLE
LA R4,DSCB05 LOAD R4 WITH ADDRESS OF DSCB05
ST R4,TOSCB05 STORE ADDRESS OF DSCB05 INTO TABLE
LA R4,DSCB06 LOAD R4 WITH ADDRESS OF DSCB06
ST R4,TOSCB06 STORE ADDRESS OF DSCB06 INTO TABLE
LA R4,DSCB07 LOAD R4 WITH ADDRESS OF DSCB07
ST R4,TOSCB07 STORE ADDRESS OF DSCB07 INTO TABLE
LA R4,DSCB08 LOAD R4 WITH ADDRESS OF DSCB08
ST R4,TOSCB08 STORE ADDRESS OF DSCB08 INTO TABLE

Figure C-109 Source of CVSEQ80 program (6 of 15)
Figure C-110  Source of CVSEQ80 program (7 of 15)
BCT R4,INIT0010 BRANCH TO INIT0010 IF C(R4) GT ZERO 03880007
DROP R2,R7,R8 DROP R2,R7,R8 03890007
INITEXIT DS OH EXIT FROM INITBRTN 03900007
L R14,INITSAVE LOAD C(INITSAVE) INTO R14 03910007
BR R14 EXIT 03920007
*
*********************************************************************** 03940007
*  PRTBRTN  * 03950007
*  - PROCESS TABLE WHICH CONTAINS ADDRESS OF DSCB FOR EACH ENTRY  * 03960007
*  RETURNED FROM CVAFSEQ CALL.  * 03970007
*  TABLE IS CURRENTLY 20 ENTRIES.  * 03980007
*  * 03990007
PRTBRTN DS OH PROCESS TABLE ENTRIES 04000007
ST R14,PRTBSAVE STORE C(R14) INTO SAVE AREA 04010007
L R4,COUNT LOAD COUNT IN R4 04020007
LA R2,TABLE LOAD ADDRESS OF TABLE INTO R2 04030007
USING TBLMAP,R2 ESTABLISH ADDRESSABILITY TO TABLE 04040007
PRTB0000 DS OH PROCESS ENTRIES 04050007
L R3,DSCBA ADDRESSABILITY TO DSCBA 04060007
CLC 0(1,R3),FMT4 IS IT A FMT4? 04070007
BNE PRTB0010 NO, THEN CONTINUE TO PROCESS DSN 04080007
B PRTB0060 YES, BRANCH TO POINT TO NEXT ENTRY 04090007
PRTB0010 DS OH FMT5 CHECK 04100007
CLC 0(1,R3),FMT5 IS IT A FMT5? 04110007
BNE PRTB0020 NO, THEN CONTINUE TO PROCESS DSN 04120007
B PRTB0060 YES, BRANCH TO POINT TO NEXT ENTRY 04130007
PRTB0020 DS OH FMT7 CHECK 04140007
CLC 0(1,R3),FMT7 IS IT A FMT7? 04150007
BNE PRTB0024 NO, THEN CONTINUE TO PROCESS DSN 04160007
B PRTB0060 YES, BRANCH TO POINT TO NEXT ENTRY 04170007
PRTB0024 DS OH FMT9 CHECK 04180007
CLC 0(1,R3),FMT9 IS IT A FMT9? 04190007
BNE PRTB0030 NO, THEN CONTINUE TO PROCESS DSN 04200007
B PRTB0060 YES, BRANCH TO POINT TO NEXT ENTRY 04210007
PRTB0030 DS OH DETERMINE IF END OF DATA WAS REACHED 04220007
CLC 0(1,R3),NODSN IS THERE '00' IN FIRST BYTE 04230007
BNE PRTB0040 NO, THEN CONTINUE TO PROCESS DSN 04240007
B PRTB0060 YES, BRANCH TO POINT TO NEXT ENTRY 04250007
PRTB0040 DS OH PROCESS DSN - FORMAT 04260007
MVC DSNMSG(44),0(R3) MOVE DSN TO PRINT LINE 04270007
PRTB0050 DS OH PROCESS / FORMAT CCHR 04280007
PUT OUTFILE,MSG1 WRITE A RECORD TO THE OUTPUT FILE 04290007
PRTB0060 DS OH PREPARE TO PROCESS NEXT TABLE ENTRY 04300007
LA R2,TBLING(R2) POINT TO NEXT TABLE ENTRY 04310007
BCT R4,PRTB0000 BRANCH TO PRTB0000 IF C(R4) GT ZERO 04320007
MVI SWEOD,EOD YES, SET SWITCH TO END OF DATA 04330007
PRTBEXIT DS OH EXIT FROM PRTBRTN 04340007
L R14,PRTBSAVE LOAD C(PRTBSAVE) INTO R14 04350007
BR R14 EXIT 04360007
*
*********************************************************************** 04380007
*  TSTRCRTN  * 04390007
*  - TEST RETURN CODE FROM CVAFSEQ  * 04400007
*  - FORMAT AND PRINT MESSAGES AS NEEDED  * 04410007
*  - INVOKE CVSTAT MODULE TO CHECK CVSTAT CODE  * 04420007
*  * 04430007

Figure C-111  Source of CVSEQ80 program (9 of 15)
Appendix C. Code samples DFSMS V1.10

Figure C-112 Source of CVSEQ8O program (10 of 15)
* CVAFSEQ ACCESS=GT,
  UCB=(R3),
  BRANCH=(YES,PGM),
  BUFLIST=BUFLISTH,
  MF=(E,(R2))
* 
  ST R15,RETCODE STORE RC INTO RETCODE
* 
* REPORT ON CV4EADOK BIT SETTING
* 
  TM CVFL4,CV4EADOK SELECT BASED ON FLAG4
  BNO NOTOK1
  PUT OUTFILE,OKMSG WRITE OK MSG RECORD
  B OK4
  NOTOK1 PUT OUTFILE,NOTOKMSG WRITE NOTOK MSG RECORD
  OK4 DS OH
* 
  BAL R14,TSTRCRTN INVOKE ROUTINE TO CHECK RC/STAT
* 
  CVS1EXIT DS OH EXIT FROM CVAFSQ1
  L R14,CVS1SAVE LOAD C(SAVE AREA) INTO R14
  BR R14 EXIT
* 
*********************************************************************** 05240007
*                           CVAFSEQ2                                   * 05250007
*     - INVOKE THE CVAFSEQ MACRO AND READ THE DSCBS                   * 05260007
*     EADSCB=NOTOK IS CODED                                           * 05270007
*     - INVOKE THE CVAFSEQ MACRO AND READ THE DSCBS                   * 05280007
* CVAFSEQ2 DS OH CVAFSEQ - ROUTINE2
  ST R14,CVS2SAVE STORE C(R14) INTO SAVE AREA
  LA R2,CVPLIST LOAD R2 WITH ADDR OF CVPLIST
  LA R3,CVPLIST2 LOAD R3 WITH ADDR OF CVPLIST
  PUT OUTFILE,CALLMR12 WRITE A REC TO OUTPUT FILE
* 
  CVAFSEQ ACCESS=GT,
  DEB=(R3),
  BRANCH=(YES,PGM),
  BUFLIST=BUFLISTH,
  EADSCB=NOTOK,
  MF=(E,(R2))
* 
  ST R15,RETCODE STORE RC INTO RETCODE
* 
* REPORT ON CV4EADOK BIT SETTING
* 
  TM CVFL4,CV4EADOK SELECT BASED ON FLAG4
  BNO NOTOK2
  PUT OUTFILE,OKMSG WRITE OK MSG RECORD
  B OK5
  NOTOK2 PUT OUTFILE,NOTOKMSG WRITE NOTOK MSG RECORD
  OK5 DS OH

Figure C-113 Source of CVSEQ80 program (11 of 15)
BAL R14, TSTRCRTN
INVOKING ROUTINE TO CHECK RC/STAT

BAL R14, TSTRCRTN
INVOKING ROUTINE TO CHECK RC/STAT

INVOKE ROUTINE TO CHECK RC/STAT

EXIT FROM CVAFSQ2

LOAD C(SAVE AREA) INTO R14

EXIT

EXIT FROM CVAFSQ2

LOAD C(SAVE AREA) INTO R14

EXIT

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

- INVOKE THE CVAFSEQ MACRO AND READ THE DSCBS
- EADSCB=OK IS CODED

CVAFSEQ ACCESS=GT,
DEB=(R3),
BRANCH=(YES, PGM),
BUFLIST=BUFLISTH,
EADSCB=OK,
MF=(E, (R2))

STORE RC INTO RETCODE

REPORT ON CV4EADOK BIT SETTING

SELECT BASED ON FLAG4

WRITE OK MSG RECORD

WRITE NOTOK MSG RECORD

WRITE A REC TO OUTPUT FILE

WRITE A REC TO OUTPUT FILE

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

WORKING STORAGE

DS OD

DC CL36'CVSEQ8O-WORKING STORAGE BEGINS HERE'

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

EQUATES

R0 EQU 0
R1 EQU 1
R2 EQU 2
R3 EQU 3
R4 EQU 4
R5 EQU 5

Figure C-114  Source of CVSEQ8O program (12 of 15)
Figure C-115  Source of CVSEQ8O program (13 of 15)
Figure C-116  Source of CVSEQ80 program (14 of 15)
Figure C-117   Source of CVSEQ8O program (15 of 15)
C.9.1.1 Execution of sample CVSEQ8O program

The CVSEQ8O program is a working example. If you want to run the program, you need to assemble and link it. It is not necessary to have in depth assembler experience, but familiarity with JCL is required.

The source is stored in one PDS, the JCL in one PDS, and the LOAD modules in another.

Required once in support of all the CVAFxxxx VTOC management programs, set up the CVSTCHK subroutine. Refer to C.13, “Supporting subroutine for sample programs” on page 636.

To create the assembler PDS, use the following steps.

There are three steps to build the program which needs to be done once, after which it can be executed several times.

Step 1: Create a PDS/PDSE to hold the source members
In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

Step 2: Create a PDS/PDSE to hold the LOAD MODULES
In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 0/U. All other attributes can be chosen by you.

Step 3: Store the program source in the PDS
Make sure that your ISPF EDIT menu settings has the screen format defined as DATA, the emulator line length is set to greater than 80, and that the edit session is in NONUM mode.

▶ Cut and paste the contents of Figure C-64 on page 562 through to Figure C-72 on page 570 one after the other into member CVDSM1O. The result should contain 407 lines.

Step 4: Create a PDS/PDSE to hold the JCL members
In this example the PDS is called MHLRES1.EAV.JCL.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.
Step 5: Store the JCL source in the PDS

- Cut and paste the contents of Figure C-118 into member CVSEQ8O. The result should contain 29 lines.

```
//MHLRES1Q JOB (1234567,COMMENT),UAALFO,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*/JOBPARM S=* 
//ASMHCL PROC
//ASM EXEC PGM=ASMA90,REGION=0M,
//       PARM='OBJECT,NODECK'
//SYSLIN DD DSN=&OBJ,DISP=(NEW,PASS),UNIT=SYSDA,
//         SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120
//SYSLIB DD DISP=SHR,DSN=SYS1.MACLIB
//      DD DISP=SHR,DSN=SYS1.MODGEN
//      DD DISP=SHR,DSN=MHLRES1.EAV.ASM
//SYSPRINT DD SYSOUT=* 
//SYSTUT DD DSN=&SYSTUT,UNIT=SYSDA,SPACE=(CYL,(5,5))
/*
//LKED EXEC PGM=HEWL,REGION=2048K,COND=(8,LE,ASM),
//       PARM='XREF,LIST,LET'
//SYSLIN DD DSN=&OBJ,DISP=(OLD,DELETE)
//      DD DDNAME=SYSIN
//SYSPRINT DD SYSOUT=* 
//SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//SYSTUT DD DSN=&SYSTUT,UNIT=SYSDA,SPACE=(CYL,(5,5))
// PEND
// EXEC ASMHCL
//ASMSYSIN DD DISP=SHR,DSN=MHLRES1.EAV.ASM(CVSEQ8O)
/*
//LKED.SYSIMOD DD DSN=MHLRES1.EAV.LOAD,DISP=SHR
//LKED.SYSIN DD *
SETCODE AC(1)
NAME CVSEQ8O(R)
```

*Figure C-118  JCL to assemble and link CVSEQ8O*

Step 6: Run the job in member CVSEQ8O

Check that the return code from the ASM and LKED steps is 0.

A return code of 8 on the LKED step is an indication that the subroutine required has not been stored in MHLRES1.EAV.LOAD.

Step 7: APF authorize the data set CVSEQ8O has been linked into

- Add the data set to the PARMLIB APF member (PROGxx) and IPL
- Or dynamically APF authorize the data set. Assuming it is MHLRES1.EAV.LOAD and that it is on volume MHLSE1, issue the command
  - `setprog apf,add,dsname=MHLRES1.EAV.LOAD,vol=MHLSE1`

The CVSEQ8O program is now ready for execution. Use the JCL as shown in Figure C-100 on page 599.
C.10 DCB OPEN of VTOC extension for EAV

The VTOC on a DASD volume is a data set, and it can be opened and read provided certain special set up is completed.

The information returned for an EAV volume is different from that for a non-EAV volume, so the control blocks have to be updated to support detail about EAV volumes.

C.10.1 EAV Migration Assistance Tracker considerations: DCB OPEN of VTOC

The EAV Migration Assistance Tracker will detect OPEN of the VTOC that has not been updated for EAV compatibility.

Any appearance on the tracker list must be investigated and resolved because use of OPEN against an EAV volume in due course will ABEND the task.

For information about the EAV Migration Tracker, refer to Appendix C of DFSMS Advanced Services, SC26-7400 and in this document to “Migration Assistance Tracker” on page 641.

In Figure C-119 we show information taken from the tracker entries that relate to program LSPACEO which has issued the LSPACE macro. The cause such an entry and an example of resolution is discussed in this section.

Figure C-119  Example of tracker entry for OPEN of VTOC

The warning DCB OPEN VTOC 113-48 corresponds with the abend code that will be issued with message IEC142I if an attempt is made to open an EAV volume VTOC without updating the macros used.

C.10.2 OPEN/DCB/READ macro use to read the VTOC

The READ macro supported by OPEN and DCB can be coded to return data from the VTOC. In order to illustrate the type of changes required to support EAV volumes, a very simple scenario is illustrated:

- Progam to OPEN and READ the VTOC of a volume in pre-z/OS V1R10 form
- Progam to OPEN and READ the VTOC of a volume in z/OS V1R10 form

C.10.2.1 Program OPNVTCO: Use of DCB not enhanced

Program OPNVTCO issues the OPEN macro against a volume that is defined through the JCL VTOC DD statement. The data set on the DD statement is insignificant and it is not used as provided, but the volume whose VTOC is to be read must be defined.

In Figure C-120 on page 620 through to Figure C-121 on page 621 we show source code to issue the OPEN and READ macros.

The information is extracted and formatted for printing.
OPEN (PRINTDCB,(OUTPUT))
RDJFCB VTOCDCB
MVC JFCBDSNM(44),=44X'04'
OPEN (VTOCDCB,(INPUT)),TYPE=J
EXTRACT ATIOT,FIELDS=(TIOT)
L R1,ATIOT TCBTIOT - POINT TO TIOT
LA R15,24(R1) TIOENTRY
USING TIOENTRY,R15
TIOTLOOP CLI TIOENTRY,X'00' END OF TIOT?
BE ABEND YES - IMPOSSIBLE
CLC TIOEDDMM,VTOCDD DOES DDNAME MATCH?
BE TIOTEXIT YES - RETURN
SR R1,R1
IC R1,TIOELNGH GET LENGTH OF ENTRY
LA R15,0(R1,R15) POINT TO NEXT ENTRY
B TIOTLOOP
ABEND DC H'0'
TIOTEXIT L R1,TIOEFSRT-1 GET UCB ADDRESS
LA R1,0(R1) CLEAR HI ORDER BYTE
ST R1,UCBAD SAVE UCB ADDRESS
MVC VOLSER,28(R1)
PUT PRINTDCB,PRINTL0
READEM READ VTOCDEC,DF,VTOCDCB,WORKAREA
CHECK VTOCDEC
MVC DSNAM,WORKAREA
CLC DSNAM(4),=X'00000000'
BE READEM IGNORE
* MVI DSNAM,C' '
* MVI DSNAM+1(43),DSNAM
* MVI DSNAM(13),=C'FORMAT-0 DSCB'
NOTF0 EQU *
CLC DSNAM(4),=X'03030303'
BNE NOTF3
MVI DSNAM,C' '
MVC DSNAM+1(43),DSNAM
MVC DSNAM(13),=C'FORMAT-3 DSCB'
NOTF3 EQU *
CLC DSNAM(4),=X'04040404'
BNE NOTF4
MVI DSNAM,C' '
MVC DSNAM+1(43),DSNAM
MVC DSNAM(13),=C'FORMAT-4 DSCB'
NOTF4 EQU *
CLC DSNAM(4),=X'05050505'
BNE NOTF5
MVI DSNAM,C' '
MVC DSNAM+1(43),DSNAM
MVC DSNAM(13),=C'FORMAT-5 DSCB'

Figure C-120  Source of OPNVTCO program (1 of 2)
Figure C-121  Source of OPNVTCO program (2 of 2)
In Figure C-122 we show the JCL to execute program OPNVTCO. The volume whose SPACE is requested is defined through the VTOC DD statement. The data set named is not significant but the VOL=SER=volser is.

```
//MHLRES1V JOB (1234567,COMMENT),UAALFO,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*JOBPARM S=* 
//OPNVTCO PROC
//ASM      EXEC PGM=OPNVTCO,REGION=0M
//STELIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD
//VTOC DD DISP=OLD,VOL=SER=MLD20C,UNIT=SYSALLDA,
//      DSNNAME='FORMAT4.DSCB'
//SYSUDUMP DD SYSOUT=*
//PRINT DD SYSOUT=* 
// PEND
// EXEC OPNVTCO
//
```

*Figure C-122  JCL to execute program OPNVTCO*

In Figure C-123 we show part of the output from running program OPNVTCO. The Format 3, 4, and 5 DSCB entries are pseudo names, the actual key is in binary form.

```
DATA SETS ON VOLUME MLD20C DSCB FID (FORMAT-0 IGNORED)
FORMAT-4 DSCB 4
FORMAT-5 DSCB 5
SYS1.VTOCIX.MLD20C 1
MHLRES1.SMF42TG.XMITOBIN 1
SYS1.VVDS.VMLD20C 1
MHLRES1.EXTREQ.DATA 1
MHLRES1.SMF8538C.TEXT 1
MHLRES4.SC63.ISPF42.ISPPROF 1
MHLRES1.VSBOX01.BACKUP 1
FORMAT-3 DSCB 3
MHLRES1.HCD.MSGLOG 1
MHLRES1.SMF4216K.TEXT 1
MHLRES4.SC64.SPFLG1.LIST 1
MHLRES4.JCL.CNTL 1
FORMAT-3 DSCB 3
MHLRES3.DUMP.MLDC65 1
MHLRES1.SUPERC.LIST 1
MHLRES1.SMF85TO.LOAD 1 
```

*Figure C-123  Output from OPNVTCO program*
In Figure C-124 we show the result of attempting to run program OPNVTCO against volume MLDC65, which is an EAV volume.

C.10.2.2 Program OPNVTCOE: Use of DCB enhanced

Program OPNVTCOE issues the OPEN macro against a volume that is defined through the JCL VTOC DD statement. The data set on the DD statement is insignificant and it is not used as provided, but the volume whose VTOC is to be read must be defined.

In Figure C-125 on page 624 through to Figure C-127 on page 626 we show source code to issue the OPEN and READ macros.

The essential change required to OPNVTCO to make OPNVTCOE is to add the DCBE macro and up update the DCB macro to refer to it.

The DCBE macro is required to allow specification of EADSCB=OK to signify that the program has been updated to support EAV.

In this simple example, the other code update is to add code to interpret the Format-9 DSCB internal name.

The information is extracted and formatted for printing.
Figure C-125  Source of OPNVTCE program (1 of 3)
NOTF5 EQU * 00520007
  CLC DNAME(1),='X'09' 00530007
  BNE NOTF9 00540007
  MVI DNAME,C' ' 00550007
  MVC DNAME+1(43),DNAME 00560007
  MVC DNAME(13),='C'FORMAT-9 DSCB' 00570007

  NOTF9 EQU * 00580007
  MVC FIDVOL,WORKAREA+44 00590007
  PUT PRINTDCB,PRINTL1 00600007
  B READEM 00610007

  * 00620007
  DC H'O' 00630007
  RETURN DS OH 00640007
  SEGEND 00650007

  EYEEYE DC CLB'WORKWORK' 00660007
  WORKAREA DS OD 00670007
  GET WORK AREA (140 BYTES)
  DS CL140 00680007
  00690007
  ATIOT DS F 00700007
  ADDRESS OF THE TIOT
  00710007
  UCBAD DS F 00720007
  ADDRESS OF THE UCB WE WANT
  00730007
  * 00740007
  DWORD DS D 00750007
  00760007
  PWORD DC CL11' ' 00770007
  00780007
  VTOCDD DC CLB'VTOC' 00790007
  00800007
  * 00810007
  PRINTDCB DCB DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133 00820007
  00830007
  VTOCDB DCB DDNAME=VTOC,DSORG=PS,MACRF=(R),EXLST=EXVTC,EODAD=RETURN,X00840007
  00850007
  KEYLEN=44,BKSIZE=96,RECFM=F,DCBE=VTOCDCBE 00860007
  00870007
  VTOCDCBE DCBE EADSCB=OK 00880007
  00890007
  EXVTC DS OF 00900007
  DC AL1(EXLLASTE+EXLRJFCB) 00910007
  DC AL3(JFCBAREA) 00920007
  00930007
  JFCBAREA DS OF,176C 00940007
  00950007
  JFCBDNMM EQU JFCBAREA 00960007
  00970007
  * 00980007
  PRINTL0 DC CL133' ' 00990007
  ORG PRINTL0+1 00990007
  DC CL1' ' 00990007
  DC 'DATA SETS ON VOLUME'
  DC CL1' ' 00990007
  DC VOLSER DC CL6' ' 00990007
  DC CL19' ' 00990007
  DC 'DSCB FID'
  DC CL1' ' 00990007
  DC CL26' (FORMAT-0 IGNORED)' 00990007
  ORG 00990007
  PRINTL1 DC CL133' ' 00990007
  ORG PRINTL1+1 00990007
  DC C' ' 00990007
  DSNAME DC CL44' ' 00990007
  DC C' ' 00990007
  FIDVOL DC CL1' ' 00990007
  ORG 00990007
  DSECT 00990007
  IHAEXLST , DCB EXIT LIST MAPPING 00990007

Figure C-126  Source of OPNVTC program (2 of 3)
In Figure C-128 we show the JCL to execute program OPNVTCE against two volumes:

- MLD20C which had been picked up by the tracker when listed using OPNVTCO and now is not when using OPNVTCE
- MLDC65 which had abended when listed using OPNVTCO but does not now when listed using OPNVTCE.

```plaintext
```

Figure C-128  JCL to execute program OPNVTCE against volumes MLD20C and MLDC65
In Figure C-129 we show part of the output from OPNVTCE running against volume MLD20C. The output is the same as from OPNVTCO showing that the enhanced code in OPNVTCE still works against non-EAV volumes.

<table>
<thead>
<tr>
<th>DATA SETS ON VOLUME MLD20C</th>
<th>DSCB FID (FORMAT-0 IGNORED)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMAT-4 DSCB</td>
<td>4</td>
</tr>
<tr>
<td>FORMAT-5 DSCB</td>
<td>5</td>
</tr>
<tr>
<td>SYS1.VTOCIX.MLD2OC</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES1.SMF42TG.XMITOBIN</td>
<td>1</td>
</tr>
<tr>
<td>SYS1.VVDS.VMLD20C</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES1.EXTPREF.DATA</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES1.EXTEQ.DATA</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES1.SMFBSMBC.TEXT</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES4.ST63.ISPP4D.ISPPROF</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES1.VSBOX01.BACKUP</td>
<td>1</td>
</tr>
<tr>
<td>FORMAT-3 DSCB</td>
<td>3</td>
</tr>
<tr>
<td>MHLRES1.HCD.MSGLOG</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES1.SMF4216K.TEXT</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES4.SC64.SPPLOG1.LIST</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES4.JCL.CNTL</td>
<td>1</td>
</tr>
<tr>
<td>FORMAT-3 DSCB</td>
<td>3</td>
</tr>
<tr>
<td>MHLRES3.DUMP.MLD65</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES1.SUPERC.LIST</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES1.SMF85STO.LOAD</td>
<td>1</td>
</tr>
<tr>
<td>MHLRES1.SMF4216D.TEXT</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure C-129  Output from execution of program OPNVTCE against volume MLD20C

In Figure C-130 we show part of the output from OPNVTCE running against volume MLDC65. Volume MLDC65 does have a cylinder managed area (is an EAV volume) and has a data set allocated there which shows up, because it is a VSAM data set, as having data set components represented by a Format-8 DSCB instead of a Format-1 DSCB and has an associated Format-9 DSCB.

<table>
<thead>
<tr>
<th>DATA SETS ON VOLUME MLDC65</th>
<th>DSCB FID (FORMAT-0 IGNORED)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMAT-4 DSCB</td>
<td>4</td>
</tr>
<tr>
<td>FORMAT-5 DSCB</td>
<td>5</td>
</tr>
<tr>
<td>SYS1.VTOCIX.MLDC65</td>
<td>1</td>
</tr>
<tr>
<td>SYS1.VVDS.VMLDC65</td>
<td>1</td>
</tr>
<tr>
<td>MHLAV.EXTKSDS.DATA</td>
<td>8</td>
</tr>
<tr>
<td>FORMAT-9 DSCB</td>
<td>9</td>
</tr>
<tr>
<td>MHLAV.EXTKSDS.IX</td>
<td>8</td>
</tr>
<tr>
<td>FORMAT-9 DSCB</td>
<td>9</td>
</tr>
</tbody>
</table>

Figure C-130  Output from execution of program OPNVTCE against volume MLDC65

C.10.2.3 Execution of sample programs to DCB OPEN the VTOC

The OPNVTCO and OPNVTCE programs are simple working examples. If you want to run either of these programs, you need to assemble link it. It is not necessary to have in depth assembler experience, but familiarity with JCL is required.

The source and support macros are stored in one PDS, the JCL in one PDS, and the LOAD modules in another.

Required once in support of all the VTOC management programs is set up of the macros. Refer to Figure C.12 on page 630.

To create the assembler PDS for OPNVTCO and/or OPNVTCE, use the following steps.

There are three steps to build the program which needs to be done once, after which it can be executed several times.
Step 1: Create a PDS/PDSE to hold the source members
In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.

Step 2: Create a PDS/PDSE to hold the LOAD MODULES
In this example the PDS is called MHLRES1.EAV.ASM.

The LRECL/RECFM must be 0/U. All other attributes can be chosen by you.

Step 3: Store the program source in the PDS
Make sure that your ISPF EDIT menu settings has the screen format defined as DATA, the emulator line length is set to greater than 80, and that the edit session is in NONUM mode.

► Choose whether to build OPNVTC1 or OPNVTC0. This example assumes that OPNVTC1 is being built.

► Cut and paste the contents of Figure C-125 on page 624 through to Figure C-127 on page 626 one after the other into member OPNVTC1. The result should contain 115 lines.

In order to assemble OPNVTC1 the supporting macros are needed. Listing of the macros are provided. They need to be stored as members in MHLRES1.EAV.ASM. The same macros are used for several programs and subroutines. Refer to C.12, “Supporting macros for sample programs” on page 630 for information about storing the macros if this has not already been done.

Step 4: Create a PDS/PDSE to hold the JCL members
In this example the PDS is called MHLRES1.EAV.JCL.

The LRECL/RECFM must be 80/FB. All other attributes can be chosen by you.
Step 5: Store the JCL source in the PDS
Cut and paste the contents of Figure C-131 into member OPNVTCE. The result should contain 29 lines.

```
//MHLRES1V JOB (1234567,COMMENT),UAALFO,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*JOBPARM S=* 
//ASMHCL PROC 
//ASM EXEC PGM=ASMA90,REGION=OM, 
// PARM='OBJECT,NODECK' 
//SYSLIN DD DSN=&OBJ,DISP=(NEW,PASS),UNIT=SYSDA, 
// SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120 
//SYSLIB DD DISP=SHR,DSN=SYS1.MODGEN 
// DD DISP=SHR,DSN=MHLRES1.EAV.ASM 
//SYSPRINT DD SYSOUT=* 
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5)) 
/*
//LKED EXEC PGM=HEWL,REGION=2048K,COND=(8,LE,ASM), 
// PARM='XREF,LIST,LET' 
//SYSLIN DD DSN=&OBJ,DISP=(OLD,DELETE) 
// DD DDNAME=SYSIN 
//SYSPRINT DD SYSOUT=* 
//SYSLIB DD DISP=SHR,DSN=MHLRES1.EAV.LOAD 
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5)) 
 // PEND 
 // EXEC ASMHCL 
 //ASM.SYSIN DD DISP=SHR,DSN=MHLRES1.EAV.ASM(OPNVTCE) 
/* 
//LKED.SYSLMOD DD DSN=MHLRES1.EAV.LOAD,DISP=SHR 
//LKED.SYSIN DD * 
NAME OPNVTCE(R) 
```

Figure C-131  JCL to assemble and link OPNVTCE

Step 6: Run the job in member OPNVTCE
Check that the return code from the ASM and LKED steps is 0.

A return code of 8 on the ASM step is an indication that the macros required have not been stored in MHLRES1.EAV.ASM.

The OPNVTCE program is now ready for execution.

An exactly similar process can be followed to set up program OPNVTCO.
C.11 DCB OPEN of EAS eligible data set extension for EAV

Any data set can be opened and read using EXCP mode provided certain special set up is completed. This is a specialized area and no examples are provided.

Under normal use, only VSAM data sets are EAS eligible, and because they are VSAM they are normally not accessed using a DCB. However, the interface for EXCP mode uses the DCB control block so if VSAM data sets are being accessed by EXCP then new provisions apply to allow for the data set access to continue if the data set is moved to an EAV volume.

The information returned for an EAV volume is different from that for a non-EAV volume, so the control blocks have to be updated to support detail about EAV volumes.

C.11.1 EAV Migration Assistance Tracker Consideration: DCB OPEN of an EAS Eligible data set

The EAV Migration Assistance Tracker will detect OPEN for EXCP (MACRF=E) of an EAS eligible data set if the related DCB has not been updated for EAV compatibility.

Any appearance on the tracker list must be investigated and resolved because use of OPEN for EXCP against a data set on an EAV volume in due course will ABEND the task.

For information about the EAV Migration Tracker, refer to Appendix C of *DFSMS Advanced Services*, SC26-7400 and in this document to “Migration Assistance Tracker” on page 641.

In Figure C-132 we show information taken from appendix C of *DFSMS Advanced Services*, SC26-7400. The cause such an entry must be investigated and resolved because an abend will occur after the data set involved moves to an EAV volume.

```
-----TRACKING INFORMATION-----  -VALUE-- JOBNAME  PROGNAME+OFF-- ASID  NUM
SMS-E:1  DCB OPEN VSAM 113-44  01 OPENJBN  OPENPGM xxx  28  4
```

*Figure C-132  Example of tracker entry for OPEN for EXCP of EAS eligible data set*

The warning DCB OPEN VTOC 113-44 corresponds with the abend code that will be issued with message IEC142I if an attempt is made to open for EXCP mode a data set on an EAV volume.

C.12 Supporting macros for sample programs

The sample programs use common macros to avoid repeating them in each listing. Before assembling the programs, the macros must be stored.

**Store the macro source in a PDS**

The macros can be stored in a separate PDS, or in the same PDS as the source. In the examples below they are stored in the same PDS as prepared for the source.
Cut and paste the contents of Figure C-133 into member SEGSTART. The result should contain 26 lines.

```
MACRO 00010000
&NAME  SEGSTART  00020000
&NAME  STM  14,12,12(13)  00030000
  SAVE HIS REGS IN HIS SAVE ARE  00040000
R0   EQU   0  00050000
R1   EQU   1  00060000
R2   EQU   2  00070000
R3   EQU   3  00080000
R4   EQU   4  00090000
R5   EQU   5  00100000
R6   EQU   6  00110000
R7   EQU   7  00120000
R8   EQU   8  00130000
R9   EQU   9  00140000
R10  EQU  10  00150000
R11  EQU  11  00160000
R12  EQU  12  00170000
R13  EQU  13  00180000
R14  EQU  14  00190000
R15  EQU  15  00200000
BALR 12,0  00210000
  SET UP ADDRESSABILITY
USING *,12  00220000
  USE REG 12 AS BASE REG
ST  13,SAVEREGS+4  00230000
  SAVE @ OF HIS SAVEAREA IN MINE
LA  03,SAVEREGS  00240000
  LOAD @ OF MY SAVE AREA IN REG 3
ST  03,8(13)  00250000
  SAVE @ OF MY SAVE AREA IN HIS
LR  13,03  00260000
  LOAD @ OF MY SAVE AREA IN REG 13
MEND
```

*Figure C-133 Macro source SEGSTART*

Cut and paste the contents of Figure C-134 into member SEGEND. The result should contain 8 lines.

```
MACRO 00010000
&NAME  SEGEND  00020000
&NAME  L  13,SAVEREGS+4  00030000
  LOAD REG13 WITH @ OF HIS SAVE
LM 14,12,12(13)  00040000
  RESTORE REGS FROM HIS SAVEAREA
XR R15,R15  00050000
BR 14  00060000
  RETURN TO CALLING RTN VIA REG 14
SAVEREGS DC 18F'0'  00070000
  SET UP SAVE AREA
MEND
```

*Figure C-134 Macro source SEGEND*

Cut and paste the contents of Figure C-135 on page 632 through Figure C-137 on page 634 one after the other into member ZCSSTART. The result should contain 169 lines.
Figure C-135  Macro source ZCSSTART (1 of 3)
Figure C-136  Macro source ZCSSTART (2 of 3)
| .N3   | AIF (BASE NE 4).N4            | 01130000 |
| &R4L  | SETC '&RB'(1,8)               | 01140000 |
|      | AGO .EQUS                    | 01150000 |
| .N4   | AIF (BASE NE 5).N5            | 01160000 |
| &R5L  | SETC '&RB'(1,8)               | 01170000 |
|      | AGO .EQUS                    | 01180000 |
| .N5   | AIF (BASE NE 6).N6            | 01190000 |
| &R6L  | SETC '&RB'(1,8)               | 01200000 |
|      | AGO .EQUS                    | 01210000 |
| .N6   | AIF (BASE NE 7).N7            | 01220000 |
| &R7L  | SETC '&RB'(1,8)               | 01230000 |
|      | AGO .EQUS                    | 01240000 |
| .N7   | AIF (BASE NE 8).N8            | 01250000 |
| &R8L  | SETC '&RB'(1,8)               | 01260000 |
|      | AGO .EQUS                    | 01270000 |
| .N8   | AIF (BASE NE 9).N9            | 01280000 |
| &R9L  | SETC '&RB'(1,8)               | 01290000 |
|      | AGO .EQUS                    | 01300000 |
| .N9   | AIF (BASE NE 10).N10          | 01310000 |
| &R10L | SETC '&RB'(1,8)               | 01320000 |
|      | AGO .EQUS                    | 01330000 |
| .N10  | AIF (BASE NE 11).N11          | 01340000 |
| &R11L | SETC '&RB'(1,8)               | 01350000 |
|      | AGO .EQUS                    | 01360000 |
| .N11  | AIF (BASE NE 13).N13          | 01370000 |
| &R13L | SETC '&RB'(1,8)               | 01380000 |
|      | AGO .EQUS                    | 01390000 |
| .N13  | AIF (BASE NE 14).N14          | 01400000 |
| &R14L | SETC '&RB'(1,8)               | 01410000 |
|      | AGO .EQUS                    | 01420000 |
| .N14  | AIF (BASE NE 15).N15          | 01430000 |
| &R15L | SETC '&RB'(1,8)               | 01440000 |
|      | AGO .EQUS                    | 01450000 |
| .N15  | AIF (BASE NE 16).N0           | 01460000 |
| &R0L  | SETC '&RB'(1,8)               | 01470000 |
|      | AGO .EQUS                    | 01480000 |
| .NOEQU| ANOP                        | 01490000 |
| &R1L  | SETC '&RB'(1,8)               | 01500000 |
| .EQUS | ANOP                        | 01510000 |
| &R0L  | EQU 0                       | 01520000 |
| &R1L  | EQU 1                       | 01530000 |
| &R2L  | EQU 2                       | 01540000 |
| &R3L  | EQU 3                       | 01550000 |
| &R4L  | EQU 4                       | 01560000 |
| &R5L  | EQU 5                       | 01570000 |
| &R6L  | EQU 6                       | 01580000 |
| &R7L  | EQU 7                       | 01590000 |
| &R8L  | EQU 8                       | 01600000 |
| &R9L  | EQU 9                       | 01610000 |
| &R10L | EQU 10                      | 01620000 |
| &R11L | EQU 11                      | 01630000 |
| &R12L | EQU 12                      | 01640000 |
| &R13L | EQU 13                      | 01650000 |
| &R14L | EQU 14                      | 01660000 |
| &R15L | EQU 15                      | 01670000 |
| .NOEQU| ANOP                        | 01680000 |
|      | MEND                          | 01690000 |

Figure C-137  Macro source ZCSSTART (3 of 3)
Cut and paste the contents of Figure C-138 and Figure C-139 one after the other into member ZCSEND. The result should contain 84 lines.

```
MACRO
  &NAME ZCSEND &RENT=,&RC=0,&SP=,&SAVE=0,&BSM=YES,&SAREA=
  LLC &RETC,&SAVO,&SAVI
  GBL &ZCSSECT,&SUBP,&ARNAME
  GBL &RETN,&NSAR
  AIF ('&SYSECT' EQ '&ZCSSECT').CHKR
  &ZCSSECT CSECT
  .CHKR AIF ('&NAME' EQ '').NONAME
  &NAME DS OH
  .NONAME AIF ('&RENT' EQ '').CHKRETC
  &RETC SETB ('&RENT' EQ 'NO')
  .CHKRETC ANOP
  &RETC SETC '&RC'
  AIF ('&RC'(1,1) NE '(').RCCON
  &RETC SETC '&RC'(2,K '&RC-2)
  .RCCON ANOP
  AIF ('&SAREA' EQ '').USENSAR
  &NSAR SETB ('&SAREA' EQ 'NO')
  .USENSAR AIF (&NSAR).RETR
  AIF (&RETN).NORENT
  &SAV0 SETC '&SAVE(1)'
  &SAV1 SETC '&SAVE(2)'
  AIF (K '&SAV0 EQ 0).ROOK
  AIF ('&SAVO'(1,1) NE 'R').ROOK
  &SAVO SETC '&SAVO'(2,K '&SAVO-1)
  .ROOK AIF (K '&SAV1 EQ 0).ROIK
  AIF ('&SAV1'(1,1) NE 'R').ROIK
  &SAV1 SETC '&SAV1'(2,K '&SAV1-1)
  .ROIK ANOP
  AIF ('&SAV0' NE '1') AND ('&SAV1' NE '1').NS1
  LR 14,1
  .NS1 ANOP
  LR 1,13
  L 13,4(13)
  AIF ('&SAVO' NE '0') AND ('&SAV1' NE '0').NOTO
  ST 0,20(13)
  .NOTO AIF ('&SAV0' NE '1') AND ('&SAV1' NE '1').NOT1
  ST 14,24(13)
  .NOT1 ANOP
  AIF ('&RC'(1,1) NE '(').NS15
  ST &RETC,16(13)
  .NS15 ANOP
  AIF ('&SUBP' NE '').NO
  &SUBP SETC '0'
  .NO AIF ('T '&SP EQ '0').FR1
  &SUBP SETC '&SP'
  .FR1 FREEMAIN R,LV=ZCSWORKL,A=(1),SP=&SUBP
  LM 14,12,12(13) RESTORE REGISTERS
  AIF ('&RC'(1,1) EQ '(').SETI
  LA 15,&RETC SET UP R15
  AGO .SETI
  .*
```

*Figure C-138  Macro source ZCSEND (1 of 2)*

Appendix C. Code samples DFSMS V1.10  635
C.13 Supporting subroutine for sample programs

The sample programs use common macros to avoid repeating them in each listing. Before assembling the programs, the macros must be stored.

C.13.1 Subroutine CVSTCHK

In this section we discuss the subroutine CVSTCHK.

Store the subroutine source in a PDS
The source should stored in same PDS as the source for the calling program.

Cut and paste the contents of Figure C-140 on page 637 through Figure C-143 on page 640 to into member CVSTCHK. The result should contain 179 lines.
**Figure C-140   Source of subroutine CVSTCHK (1 of 4)**

```
**********************************************************************  00010010
*                     C  V  S  T  C  H  K                            *  00020010
*                                                                    *  00030010
* PROGRAM: THIS IS THE CVSTAT CODE TABLE SEARCH SUBPROGRAM           *  00040010
* MODIFIED TO JUST INTERPRET CVSTAT AND RETURN DEC AND HEX           *  00050010
* ORIGINAL AUTHOR: HUGH ETZELL                                       *  00060010
*                                                                    *  00070010
**********************************************************************  00080010
EJECT                                                              00090010
**********************************************************************  00100010
* CVSTCHK                                                            *  00110010
*                                                                    *  00120010
* INPUT                                                              *  00130010
* PARAMETERS: A(WCVSTCD,WFMTREC)                                     *  00140010
*                                                                    *  00150010
* WCVSTCD  - CVSTAT CODE PASSED AS INPUT (IN HEX)                    *  00160010
* WFMTREC  - WORK AREA TO FORMAT RECORD                              *  00170010
*                                                                    *  00180010
* THE FOLLOWING FORMAT(S) FOR THIS RECORD WILL BE USED:              *  00190010
*                                                                    *  00200010
* CVSTAT CODE: X'FF' DEC'999'                                        *  00210010
*                                                                    *  00220010
**********************************************************************  00230010

CVSTCHK  CSECT                    CVSTCHK CONTROL SECTION               00240010
CVSTCHK  AMODE 24                                                       00250010
CVSTCHK  RMODE 24                 CVSTCHK CONTROL SECTION               00260010
PRINT NOGEN              INHIBIT MACRO EXPANSION               00270010
STM   R14,R12,12(R13)    STANDARD LINKAGE CONVENTION           00280010
BALR  R10,0              R10 IS IMPLIED BASE REG               00290010
USING BASE,R10,R11,R12   R11 AND R12 ALSO BASE REG            00300010
BASE     L     R11,BASEAD1        SET UP ADDRESSING FOR R11             00310010
L     R12,BASEAD2        SET UP ADDRESSING FOR R12             00320010
B     CVST0000           BRANCH AROUND DECLARES                00330010
BASEAD1  DC    A(BASE+4096)       ADDRESSING FOR BASE R11               00340010
BASEAD2  DC    A(BASE+8192)       ADDRESSING FOR BASE R12               00350010
DC    CL8''      OBTAIN SYSTEM DATE                            00360010
DC    CL8''      OBTAIN SYSTEM TIME                            00370010
CVST0000 DS    0H                 HOUSEKEEPING...                       00380010
LR    R2,R13             LOAD ADDR OF CALLING PGM SAVE AREA    00390010
*                                 INTO R2                               00400010
LA    R13,CVSTSAVE         LOAD ADDR OF SAVE AREA INTO R13        00410010
ST    R13,(R2)            STORE ADDR OF THIS RTN'S SAVE AREA       00420010
*                                 INTO THE THIRD WORD OF THE CALLING   00430010
*                                 PGM'S SAVE AREA                        00440010
*                                 AREA INTO THE SECOND WORD OF THIS      00450010
*                                 RTN'S SAVE AREA                       00460010
LM    R2,R3,0(R1)         LOAD R2=*WCVSTCD), R3=*WFMTREC)            00470010
MVC   WCVCODE(1),0(R2)   MOVE 1 BYTE FROM WCVSTCD TO WCVCODE      00480010
*                                 NOW IN CVSTCHK SUBPGM                  00490010
WTO  '   NOW IN CVSTCHK SUBPGM    ',ROUTCODE=(2,11)            00500010
00510010
00520010
00530010

```
* BRANCH TO TOO HIGH IN TABLE 00540010
BAL R14,CONVHRTN CALL CONVERT TO HEX RTN 00550010
MVC WFORDEC(3),CQDEC MOVE ??? TO DECIMAL PLACE HOLDER 00560010
LH R1,WCVCODEH 00570010
CVD R1,DWORD 00580010
OI DWORD+7,X'OF' 00590010
UNPK PWORD(11),DWORD+2(6) 00600010
MVC WFORDEC(3),PWORD+8 00610010
MVC WFORDESC(46),CBLANKS CLEAR OUT DESCRIPTION-JUST IN CASE 00620010
MVC O(133,R3),WFORMAT MOVE FORMATED LINE TO WFMREC 00630010
LM R14,R12,12(R13) PREPARE TO EXIT 00660010
BR R14 EXIT 00670010
********************************************************************** 00680010
* CONVHRTN - CONVERT TO HEX ROUTINE 00690010
* WTO ' NOW IN CONV ROUTINE ',ROUTCDE=(2,11) 00700010
CONVHRTN DS OH CONVERT TO HEX ROUTINE 00710010
ST R14,CONVSAVE STORE C(R14) INTO SAVE AREA 00720010
* DOWHILE LENGTH TO CONVERT (GT) ZERO 00730010
CONV0010 DS OH DOWHILE LENGTH TO CONVERT (GT) ZERO 00740010
L R5,CONE LOAD R5 WITH ONE - LENGTH 00750010
LA R6,THEX1 LOAD R6 WITH ADDR OF HEX TABLE 1 00760010
LA R7,TCHAR1 LOAD R7 WITH ADDR OF CHAR TABLE 1 00770010
L R8,CONE LOAD R8 WITH 1 - LOOP INCREMENT 00780010
L R9,C15 LOAD R9 WITH 15 - LOOP LIMIT 00790010
SR R2,R2 INIT R2 WITH ZERO - LOOP COUNTER 00800010
MVC WBYTE(1),WCVCODE MOVE BYTE TO BE CONVERTED TO WBYTE 00810010
NI WBYTE,EF0 (AND IMED) SET LAST HALF BYTE TO 0 00820010
CONV0020 DS OH DOWHILE 1ST HALF BYTE NOT CONVERTED 00830010
CLC WBYTE(1),0(R6) IF WBYTE EQ HEX1 TBL ENTRY 00840010
BE CONV0030 BRANCH TO CONVERT 1ST HALF BYTE 00850010
LA R6,1(R6) INCREMENT TO NEXT HEX TBL ENTRY 00860010
BXLE R2,R8,CONV0020 R2=R2+R8, IF (LE) R9 CONTINUE 00870010
CONV0030 DS OH CONVERT 1ST HALF BYTE 00880010
AR R7,R2 R7 NOW HAS RELATED CHAR TBL ADDR 00890010
MVC WFORHEX(1),0(R7) MOVE CONVERTED HALF BYTE TO WFORHEX 00900010
LA R6,THEX2 LOAD R6 WITH ADDR OF HEX TABLE 2 00910010
LA R7,TCHAR2 LOAD R7 WITH ADDR OF CHAR TABLE 2 00920010
L R8,CONE LOAD R8 WITH 1 - LOOP INCREMENT 00930010
L R9,C15 LOAD R9 WITH 15 - LOOP LIMIT 00940010
SR R2,R2 INIT R2 WITH ZERO - LOOP COUNTER 00950010
MVC WBYTE(1),WCVCODE MOVE BYTE TO BE CONVERTED TO WBYTE 00960010
NI WBYTE,EOF (AND IMED) SET LAST HALF BYTE TO 0 00970010
CONV0040 DS OH DOWHILE 2ND HALF BYTE NOT CONVERTED 00980010
CLC WBYTE(1),0(R6) IF WBYTE EQ HEX2 TBL ENTRY 00990010
BE CONV0050 BRANCH TO CONVERT 2ND HALF BYTE 01000010
LA R6,1(R6) INCREMENT TO NEXT HEX TBL ENTRY 01010010
BXLE R2,R8,CONV0040 R2=R2+R8, IF (LE) R9 CONTINUE 01020010
CONV0050 DS OH CONVERT 2ND HALF BYTE 01030010
AR R7,R2 R7 NOW HAS RELATED CHAR TBL ADDR 01040010
MVC WFORHEX+1(1),0(R7) MOVE CONVERTED HALF BYTE TO WFORHEX 01050010
BCT R5,CONV0010 R5=R5-1, NE ZERO BRANCH TO CONV0010 01060010
Figure C-141 Source of subroutine CVSTCHK (2 of 4)
CONVEXIT DS 0H  EXIT FROM CONVHRTN ROUTINE  01070010
L R14,CONVSAVE  LOAD C(CONVSAVE) INTO R14  01080010
BR R14  EXIT  01090010
EJECT  01100010
******************************************************************************
* WORKING STORAGE *  01110010
DS OD  01120010
DC CL36'CVSTCHK -WORKING STORAGE BEGINS HERE'  01130010
******************************************************************************
* EQUATES *  01140010
EFO EQU X'F0'  SET LAST HALF BYTE TO ZERO (NI)  01150010
E0F EQU X'0F'  SET 1ST HALF BYTE TO ZERO (NI)  01160010
R0 EQU 0  REGISTER EQUATE  01170010
R1 EQU 1  REGISTER EQUATE  01180010
R2 EQU 2  REGISTER EQUATE  01190010
R3 EQU 3  REGISTER EQUATE  01200010
R4 EQU 4  REGISTER EQUATE  01210010
R5 EQU 5  REGISTER EQUATE  01220010
R6 EQU 6  REGISTER EQUATE  01230010
R7 EQU 7  REGISTER EQUATE  01240010
R8 EQU 8  REGISTER EQUATE  01250010
R9 EQU 9  REGISTER EQUATE  01260010
R10 EQU 10  REGISTER EQUATE  01270010
R11 EQU 11  REGISTER EQUATE  01280010
R12 EQU 12  REGISTER EQUATE  01290010
R13 EQU 13  REGISTER EQUATE  01300010
R14 EQU 14  REGISTER EQUATE  01310010
R15 EQU 15  REGISTER EQUATE  01320010
EJECT  01330010
******************************************************************************
* SAVE AREAS *  01340010
CVSTSAVE DS 18F  CVSTCHK ROUTINE SAVE AREA  01350010
CONVSHERE DS F'0'  CONVTHEX ROUTINE SAVE AREA  01360010
******************************************************************************
* CONSTANTS *  01370010
C15 DC F'15'  FULL WORD - FIFTEEN  01380010
CONE DC F'1'  FULL WORD - ONE  01390010
CBLANKS DC CL46' '  01400010
CQDEC DC CL3'???'  01410010
******************************************************************************
* WORK AREAS *  01420010
DWORD DS D  01430010
PWORD DC CL11 ' '  01440010
HALIGN DS OH  01450010
WCVCODEH DC X'00'  01460010
WCVCODE DC XL1'FF'  WORK AREA USED FOR CVSTAT CODE  01470010
WBYTE DC CL1 ' '  CONVERTED BYTE WORK AREA  01480010
WFORMAT DS OCL133  OUTPUT FORMAT LINE  01490010
DC CL2 ' '  01500010
DC C'CVSTAT CODE: '  01510010
DC CL2'X''  01520010
WFORHEX DC CL2 ' '  PLACE HOLDER FOR HEX CODE  01530010
DC CL1'''  01540010
DC CL4' '  01550010
DC CL4'DEC'''  01560010
Appendix C. Code samples DFSMS V1.10  639
Store the JCL source in the PDS

- Cut and paste the contents of Figure C-144 into member CVSTCHK. The result should contain 29 lines.
Run the job in member CVSTCHK
Check that the return code from the ASM and LKED steps is 0.

The CVSTCHK subroutine now ready to be linked in with any program load modules that require it.

C.14 Migration Assistance Tracker

To prepare for EAV it is important that programs accessing VTOC’s recognize the changed addressing beyond the current limit of 65520 cylinders. The EAV Migration Assistance (MA) Tracker is an important tool that can help you find programs that you might need to change to support the Extended Address Volume (EAV) format. For users that used the Console ID Tracking facility to identify 1-byte console Ids, they will be familiar with the MA extension for EAV tracking.

The Migration Assistance Tracker is described in DFSMS Advanced Services, SC26-7400. It will help you prepare for introduction of EAV volumes because it can:

- Identify select systems services by job and program name, where the invoking programs might require analysis for changes to use new services. The program calls are identified as informational instances for possible migration actions. They are not considered errors, because the services return valid information.

- Identify possible instances of improper use of returned information in programs, like parsing 28-bit cylinder numbers in output as 16-bit cylinder numbers. These instances are identified as warnings.

- Identify instances of programs that will either fail or run with an informational message if they run on an EAV. These are identified as programs in error. The Migration Assistance Tracker flags programs with the functions shown below, when the target volume of the operation is non-EAV, and the function invoked did not specify the EADSCB=OK keyword (new permission keyword, the caller has to issue):
  - DADSM
  - OBTAIN
  - CVAFDIR
  - CVAFSEQ
  - CVAFDSM
  - CVAFFILT
  - CVAFFSM
  - OPEN of VTOC
  - DCB OPEN of an EAS eligible data set

This enables the systems programmer to identify programs in error by job and program name, without failing the programs. It also allows you to exclude programs, that are not yet ready for evaluation. Programs identified in this phase of migration assistance tracking will continue to fail if the system service is issued for an EAV, if you do not specify the EADSCB=OK keyword for them.
DFSMS instances tracked by the EAV migration assistance tracker

In this section we discuss EAV tracking of DFSMS instances.

**LSPACE (SVC 78)**

An LSPACE request with the DATA=, MSG=, or EXPMSG= keywords was issued. Additional data from track-managed space is available with the EXPDATA= and XEXPMSG= keywords. When this instance occurs for any volume type, it will be recorded in the tracker as an informational message. These five keywords are mutually exclusive.

LSPACE processing (IGC0007H) will set the following tracking information:

TRPL\_TRACK\_INFO =

'SMS-I:3 LSPACE reqtype '  
Where:

reqtype = DATA= or MSG= or EXPMSG=

TRPL\_TRACK\_DATA =

Byte 0-1

Set to zero, not used.

Byte 2 Set to the LSPACE parameter flag byte

BIT 2 ON INDICATES THAT LSPACE WITH THE DATA= KEYWORD WAS SPECIFIED

BIT 3 ON INDICATES THAT LSPACE WITH THE MSG= KEYWORD WAS SPECIFIED

BIT 4 ON INDICATES THAT LSPACE WITH THE EXPMSG=KEYWORD WAS SPECIFIED

Byte 3

Set to 03

DFSMS Tracking category 3: EAV Migration. Informational Message. New function is available. Additional data from track-managed space is available with the EXPDATA= and XEXPMSG= keywords.

TRPL\_VIOLATORS\_ADDR =

SET TO THE RESUME PSW OF THE RB THAT ISSUED SVC 78. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

In Figure C-145 we show a sample display of data from the tracking assistance for LSPACE macro instances.

```
15.00.00 SYSTEM1 d opdata,tracking
15.00.00 SYSTEM1 CNZ1001I 15.00.00 TRACKING DISPLAY 631
STATUS=ON NUM=3 MAX=1000 MEM=7T EXCL=0 REJECT=0
-----TRACKING INFORMATION----- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM
SMS-I:3 LSPACE MSG= 5003 ALLOCAS IEFW21SD 4CE5C 11 1
SMS-I:3 LSPACE DATA= 2003 VTDS0IS1 VTDS0IS2 118 28 2
SMS-I:3 LSPACE EXPMSG= 8803 VTDS0IS1 VTDS0IS2 118 28 2
SMS-I:3 LSPACE MSG= 9003 *MASTER* IEE70110 52F6 01 46
```

*Figure C-145  Tracking data from LSPACE macro exceptions*
In Figure C-146 we show a sample exclusion list for the LSPACE function.

<table>
<thead>
<tr>
<th>Jobname</th>
<th>Pgmname</th>
<th>Comments (ignored)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS-I:3 LSPACE*</td>
<td><em>MASTER</em></td>
<td>IEE70110</td>
</tr>
<tr>
<td>SMS-I:3 LSPACE*</td>
<td>ALLOCAS</td>
<td>IEFW21SD</td>
</tr>
<tr>
<td>SMS-I:3 LSPACE*</td>
<td>*</td>
<td>VTDSOIS2</td>
</tr>
<tr>
<td>SMS-I:3 LSPACE*</td>
<td>VTDSOIS1</td>
<td>*</td>
</tr>
</tbody>
</table>

**DEVTYPE (SVC 24)**

A DEVTYPE request with DEVTAB or UCBLIST without INFOLIST, returns the number of cylinders on the volume. This is in a two-byte field at offset 8, which is too small if the volume has more than 65 520 cylinders. Consider using INFO=DASD, which returns the number of cylinders in a four-byte field.

When INFO=DASD is specified, additional fields are now provided, which include these (see mapping macro IHADVA):

- **DVAIXVLD BIT**
  - DVACYLMG, DVAEADSCB, DVAWIRSZ valid

- **DVACYLMG BIT**
  - Cylinder-managed space exists on this volume and begins at DVALCYL
  - in multicylinder units of DVAMCU.

- **DVAEADSCB BIT**
  - Extended attribute DSCBs, Format 8 and 9 DSCBs, are allowed on this volume. Valid when DVAIXVLD is set.

- **DVAMCU 8-BIT UNSIGNED INTEGER**
  - Minimum allocation size in cylinders for cylinder-managed space. Each extent in this space must be a multiple of this value. Also referred to as the multicylinder unit (MCU). This is the smallest unit of disk space in cylinders that can be allocated in cylinder-managed space.
  - Valid when DVACYLMG is set.

  This field is zero on releases before z/OS 1.10 or if the status is not yet known. In these two cases DVAIXVLD is not set.
DVALCYL 16-BIT UNSIGNED INTEGER
* First cylinder address divided by 4095 where space is managed in multicylinder units. Cyl-managed space begins at this address.
* Valid when DVACLYMG is set. This field is zero on releases before z/OS 1.10 or if the status is not yet known. In these two cases DVAIXVLD is not set.

DVAVIRSZ 16-BIT UNSIGNED INTEGER
* Block size of the index data set.
* Valid when DAVIXVLD is set on.
* When valid and zero the volume has no working VTOC index. This field is zero on releases before z/OS 1.10 or if the status is not yet known. In these cases DVAIXVLD is not set.

When this instance occurs for any volume type, it will be recorded in the tracker as an informational message.

DEVTYPE processing (IGC0002D) will set the following tracking information:

TRPL_TRACK_INFO =
   'SMS-I:3 DEVTYPE '

TRPL_TRACK_DATA =
Byte 0-2
   Set to zero, not used.
Byte 3
   Set to 03.
   DFSMS Tracking category 3: EAV Migration. Informational Message. New function is available. Additional data from DEVTYPE INFO=DASD invocation is available. See mapping macro IHADVA.

TRPL_VIOLATORS_ADDR =
   SET TO THE RESUME PSW OF THE RB THAT ISSUED SVC 24. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS

In Figure C-147 we show a sample display of data from the tracking assistance for DEVTYPE macro instances.

```
15.00.00 SYSTEM1 d opdata,tracking
15.00.00 SYSTEM1 CNZI001I 15.00.00 TRACKING DISPLAY 631
STATUS=ON NUM=1 MAX=1000 MEM=7T EXCL=0 REJECT=0
-----TRACKING INFORMATION----- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM
SMS-I:3 DEVTYPE 03 DEVTPJOB DEVTPROG 4CE5C 11 1
```

Figure C-147  Tracking data from DEVTYPE macro exceptions
In Figure C-148 we show a sample exclusion list for the DEVTYPE function.

<table>
<thead>
<tr>
<th>Jobname</th>
<th>Pgname</th>
<th>Tracking Information Mask</th>
<th>Mask</th>
<th>Mask</th>
<th>Comments (ignored)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS<em>DEVTYPE</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ALL DEVTYPE</td>
</tr>
</tbody>
</table>

**Figure C-148   DEVTYPE tracking assistant exclusion list sample**

**IDCAMS LISTDATA PINNED**

An IDCAMS LISTDATA PINNED request was processed. The track addresses for the PINNED tracks can contain 28-bit cylinder numbers.

When this instance occurs for any volume type, it will be recorded in the tracker as a warning message.

IDCAMS LISTDATA PINNED processing (IDCSS05) will set the following tracking information:

**TRPL_TRACK_INFO =**

`'SMS-W:2 IDCAMS LISTDATA PINN'`

**TRPL_TRACK_DATA =**

- Byte 0-2
  - Set to zero, not used.
- Byte 3
  - Set to 02.

DFSMS Tracking category 2: EAV Migration. Warning Message. An IDCAMS LISTDATA PINNED request was processed. The track addresses for the PINNED tracks can contain 28-bit cylinder numbers.

**TRPL_VIOLATORS_ADDR =**

SET TO THE RESUME PSW OF THE PRB WITH A VALID POINTER (RBCDE1) TO THE 'IDCAMS' CDNAME. USE THE PREVIOUS RB (RBLINKB)

In Figure C-149 we show a sample display of data from the tracking assistance for IDCAMS LISTDATA PINNED instances.

```
07.34.08 SYSTEM1 d_opdata,tracking
07.34.08 SYSTEM1 CNZ1001I 07.34.08 TRACKING DISPLAY 673
STATUS=ON,ABEND NUM=1 MAX=1000 MEM=7T EXCL=5 REJECT=0
-----TRACKING INFORMATION----- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM
SMS-W:2 IDCAMS LISTDATA PINN 02 LISTDATA IDCAMS E48E 28 4
```

**Figure C-149   Tracking data from IDCAMS LISTDATA PINNED exceptions**
In Figure C-150 we show a sample exclusion list for IDCAMS LISTDATA PINNED.

<table>
<thead>
<tr>
<th>Jobname</th>
<th>Pgname</th>
<th>Tracking Information Mask</th>
<th>Mask</th>
<th>Comments (ignored)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS<em>LISTDATA PINNED</em></td>
<td>*</td>
<td>IDCAMS</td>
<td>All IDCAMS PGM CALLS</td>
<td></td>
</tr>
</tbody>
</table>

*IEHLIST LISTVTOC*

An IEHLIST LISTVTOC request was processed. Extent descriptors can contain cylinder addresses 65520 or larger. Free space descriptors can contain track addresses 982800 or larger and/or full cylinders 65520 or larger. The generated report will display the information in different columns as compared to reports generated on releases prior to z/OS V1.10.

When this instance occurs for any volume type, it will be recorded in the tracker as a warning message.

IEHLIST LISTVTOC processing will set the following tracking information:

TRPL_TRACK_INFO =

`'SMS-W:2 IEHLIST LISTVTOC`'

TRPL_TRACK_DATA =

Byte 0-2

Set to zero, not used.

Byte 3

Set to 02.

DFSMS Tracking category 2: EAV Migration. Warning Message. An IEHLIST LISTVTOC request was processed. Extent descriptors can contain cylinder addresses 65520 or larger.

TRPL_VIOLATORS_ADDR =

SET TO THE RESUME PSW OF THE PRB WITH A VALID POINTER (RBCDE1) TO THE 'IEHLIST' CDNAME. USE THE PREVIOUS RB (RBLINKB)

In Figure C-151 we show a sample display of data from the tracking assistance for IEHLIST LISTVTOC instances.

```
07.34.08 SYSTEM1 d opdata,tracking
07.34.08 SYSTEM1 CNZ1001I 07.34.08 TRACKING DISPLAY 673
STATUS=ON,ABEND NUM=1 MAX=1000 MEM=7T EXCL=5 REJECT=0
--------TRACKING INFORMATION-------- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM
SMS-W:2 IEHLIST LISTVTOC 02 LNKLIST LNKLIST 20 29 2
SMS-W:2 IEHLIST LISTVTOC 02 LST004 IEHLIST 2304 29 1
SMS-W:2 IEHLIST LISTVTOC 02 LST004 IEHLIST 34A6 29 1
```

*Figure C-150  IDCAMS LISTDATA PINNED tracking assistant exclusion list sample

*Figure C-151  Tracking data from IEHLIST LISTVTOC exceptions*
In Figure C-152 we show a sample exclusion list for IEHLIST LISTVTOC.

![Exclusion List Sample](image)

**IDCAMS DCOLLECT**

An IDCAMS DCOLLECT request for ‘V’ (Volume Record Field) and ‘VL’ (SMS Volume Definition Field) records was processed.

Additional data for track-managed space was recorded.

When this instance occurs for any volume type, it will be recorded in the tracker an informational message.

IDCAMS DCOLLECT processing will set the following tracking information:

- **TRPL_TRACK_INFO** =
  
  ‘SMS-I:3 IDCAMS DCOLLECT’

- **TRPL_TRACK_DATA** =
  
  Byte 0-2
  Set to zero, not used.

  Byte 3
  Set to 03.

  DFSMS Tracking category 3: EAV Migration. Informational Message. An IDCAMS DCOLLECT request for ‘V’ (Volume Record Field) and ‘VL’ (SMS Volume Definition Field) records was processed. Additional data for track-managed space was recorded.

- **TRPL_VIOLATORS_ADDR** =

  SET TO THE RESUME PSW OF THE PRB WITH A VALID POINTER (RBCDE1) TO THE ‘IDCAMS’ CDNAME. USE THE PREVIOUS RB (RBLINKB) IF THIS PRB IS NOT THE FIRST RB. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS

In Figure C-153 we show a sample display of data from the tracking assistance for IDCAMS DCOLLECT instances.

![Tracking Data Sample](image)
In Figure C-154 we show a sample exclusion list for IDCAMS DCOLLECT.

<table>
<thead>
<tr>
<th>Jobname</th>
<th>Pgname</th>
<th>Tracking Information Mask</th>
<th>Mask</th>
<th>Comments (ignored)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS<em>DCOLLECT</em></td>
<td>*</td>
<td>IDCAMS</td>
<td>IDCAMS DCOLLECT PGM</td>
<td></td>
</tr>
</tbody>
</table>

**IDCAMS LISTCAT**

An IDCAMS LISTCAT request was processed that printed extent descriptors for one or more EAS eligible data set (VSAM in z/OS V1R10). The returned extent descriptors can contain 28-bit cylinder numbers.

When this instance occurs for any volume type, it will be recorded in the tracker as a warning message. IDCAMS LISTCAT processing will set the following tracking information:

TRPL_TRACK_INFO =

'SMS-W:2 IDCAMS LISTCAT ' 

TRPL_TRACK_DATA =

Byte 0-2
   Set to zero, not used.

Byte 3
   Set to 02.

DFSMS Tracking category 2: EAV Migration. Warning Message. An IDCAMS LISTCAT request was processed that printed extent descriptors for one or more EAS eligible data set (VSAM in z/OS V1R10). The returned extent descriptors can contain 28-bit cylinder numbers. This instance will be recorded for both EAS and non-EAS capable volumes. Note that AMS Listcat output format can change as a result of service and new function support. IBM recommends applications processing LISTCAT output be updated to obtain results directly from the Catalog Search Interface (CSI). For more information about CSI, see z/OS DFSMS Managing Catalogs and HLASM Programmer's Guide.

TRPL_VIOLATORS_ADDR =

SET TO THE RESUME PSW OF THE PRB WITH A VALID POINTER (RBCDE1) TO THE 'IDCAMS' CDNAME. USE THE PREVIOUS RB (RBLINKB) IF THIS PRB IS NOT THE FIRST RB. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

In Figure C-155 we show a sample display of data from the tracking assistance for IDCAMS LISTCAT instances.

07.34.08 SYSTEM1 d opdata,tracking
07.34.08 SYSTEM1 CNZ1001I 07.34.08 TRACKING DISPLAY 673
STATUS=ON,ABEND NUM=1 MAX=1000 MEM=7T EXCL=5 REJECT=0
-----TRACKING INFORMATION----- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM
SMS-W:2 IDCAMS LISTCAT 02 LISTCAT IDCAMS xxx 28 4

**Figure C-155** Tracking data from IDCAMS LISTCAT exceptions
In Figure C-156 we show a sample exclusion list for IDCAMS DCOLLECT.

<table>
<thead>
<tr>
<th>* Jobname Pgmname *</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Tracking Information Mask Mask Mask Comments (ignored) *</td>
</tr>
<tr>
<td>SMS*IDCAMS LISTCAT *</td>
</tr>
</tbody>
</table>

Figure C-156  IDCAMS LISTCAT tracking assistant exclusion list sample

**OBTAIN (SVC 27)**

OBTAIN was issued with the search or seek option to a non-EAV volume. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs and the target data set is EAS eligible.

When this instance occurs for a non-EAV volume type, it will be recorded in the tracker as an error message.

OBTAIN processing will set the following tracking information:

TRPL_TRACK_INFO =

`'SMS-E:1 DADSM OBTAIN '`

TRPL_TRACK_DATA =

Byte 0

Set to zero, not used.

Byte 1-2

Operation code.

X'C100' SEARCH for DSNAME.

X'C080' SEEK for track address.

Byte 3

Set to 01.

DFSMS Tracking category 1: EAV Migration. Error Message. DADSM OBTAIN was issued with the search or seek option to a non-EAV volume. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs and the target data set is 'EAS eligible'.

TRPL_VIOLATORS_ADDR =

SET TO THE RESUME PSW OF THE RB THAT ISSUED SVC 27. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

In Figure C-157 we show a sample display of data from the tracking assistance for OBTAIN (SVC 27) instances.

<table>
<thead>
<tr>
<th>Jobname</th>
<th>Pgmname</th>
</tr>
</thead>
</table>
| Tracking Information Mask Mask Mask Comments (ignored) *
| SMS*IDCAMS LISTCAT | IDCAMS | IDCAMS LISTCAT PGM |

Figure C-157  Tracking data from OBTAIN (SVC27) exceptions
In Figure C-158 we show a sample exclusion list for OBTAIN (SVC 27).

<table>
<thead>
<tr>
<th>Jobname</th>
<th>Pgname</th>
<th>Comments (ignored)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS<em>OBTAIN</em></td>
<td>*</td>
<td>DADSM OBTAIN</td>
</tr>
</tbody>
</table>

**CVAFDIR**

CVAFDIR was issued with the search or seek option to a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs and the target data set is ‘EAS eligible’. CVAF return code 4 and CVSTAT of X’52’ would have been set if issued to a device that supports extended attribute DSCBs.

When this instance occurs for a volume that does not support extended attribute DSCBs, it will be recorded in the tracker as an error message.

CVAFDIR processing will set the following tracking information:

**TRPL_TRACK_INFO =**

'SMS-E:1 CVAFDIR STAT082 '

**TRPL_TRACK_DATA =**

Byte 0

Set to zero, not used.

Byte 1

CVAF Return Code = 4

Byte 2

CVAF Status Code = STAT082 (X’52’)

Byte 3

Set to 01

DFSMS Tracking category 1: EAV Migration. Error Message. CVAFDIR was issued with the search or seek option to a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs and the target data set is ‘EAS eligible’. CVAF return code 4 and CVSTAT of X’52’ would have been set if issued to a device that supports extended attribute DSCBs.

**TRPL_VIOLATORS_ADDR =**

FOR BRANCH ENTRY CALLERS, SET TO THE CALLER OF CVAF RETURN ADDRESS. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

FOR SVC CALLS, SET TO THE RESUME PSW OF THE RB THAT ISSUED SVC 139. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.
In Figure C-159 we show a sample display of data from the tracking assistance for CVAFDIR instances.

![Figure C-159 Tracking data from CVAFDIR exceptions](image)

In Figure C-160 we show a sample exclusion list for CVAFDIR.

![Figure C-160 CVAFDIR tracking assistant exclusion list sample](image)

**CVAFSEQ**

CVAFSEQ was issued for physical sequential or index order to a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs and the target data set is ‘EAS eligible’. CVAF return code 4 and CVSTAT of X’52’ would have been set if issued to a volume that supports extended attribute DSCBs.

When this instance occurs for a volume that does not support extended attribute DSCBs, it will be recorded in the tracker as an error message.

CVAFSEQ processing will set the following tracking information:

```plaintext
TRPL_TRACK_INFO =
   'SMS-E:1 CVAFSEQ STAT082'

TRPL_TRACK_DATA =
   Byte 0
      Set to zero, not used.
   Byte 1
      CVAF Return Code = 4
   Byte 2
      CVAF Status Code = STAT082 (X’52’)
   Byte 3
      Set to 01.
```

DFSMS Tracking category 1: EAV Migration. Error Message. CVAFSEQ was issued for physical sequential or index order to a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs and the target data set is ‘EAS eligible’. CVAF return code 4 and CVSTAT of X’52’ would have been set if issued to a volume that supports extended attribute DSCBs.
TRPL_VIOLATORS_ADDR =

FOR BRANCH ENTRY CALLERS, SET TO THE CALLER OF CVAF RETURN ADDRESS. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

FOR SVC CALLS, SET TO THE RESUME PSW OF THE RB THAT ISSUED SVC 139. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

In Figure C-161 we show a sample display of data from the tracking assistance for CVAFSEQ instances.

```
07.34.08 SYSTEM1 d opdata,tracking
07.34.08 SYSTEM1 CNZ1001I 07.34.08 TRACKING DISPLAY 673
STATUS=ON,ABEND NUM=1 MAX=1000 MEM=7T EXCL=5 REJECT=0
-----TRACKING INFORMATION----- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM
SMS-E:1 CVAFSEQ STAT082 045201 CVAFJBN CVAFPGM xxx 28 4
```

Figure C-161  Tracking data from CVAFSEQ exceptions

In Figure C-162 we show a sample exclusion list for CVAFSEQ.

```
* Tracking Information Mask Jobname Pgmname
*-------------------+--------+--------+------------------+
|SMS*CVAFSEQ STAT082* |* |* | CVAFSEQ |
```

Figure C-162  CVAFSEQ tracking assistant exclusion list sample

**CVAFDSM**

CVAFDSM was issued to retrieve unallocated space on a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs. CVAF return code 4 and CVSTAT of X'52' would have been set if issued to a volume that supports extended attribute DSCBs. Note that CVAFVSM interface is an internal system function that is not documented externally for general use.

When this instance occurs for a volume that does not support extended attribute DSCBs, it will be recorded in the tracker as an error message.

CVAFDSM processing will set the following tracking information:

TRPL_TRACK_INFO =

`SMS-E:1 CVAFDSM STAT082` 

TRPL_TRACK_DATA =

Byte 0
  Set to zero, not used.

Byte 1
  CVAF Return Code = 4

Byte 2
  CVAF Status Code = STAT082 (X'52')

Byte 3
Set to 01.

DFSMS Tracking category 1: EAV Migration. Error Message. CVAFDSM was issued to retrieve unallocated space on a volume (CVAFDSM ACCESS=MAPDATA, MAP=VOLUME, RTA4BYTE=YES) that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs. CVAF return code 4 and CVSTAT of X’52’ would have been set if issued to a volume that supports extended attribute DSCBs. CVAFVSM interface is an internal system function that is not documented externally for general use.

TRPL_VIOLATORS_ADDR =

FOR BRANCH ENTRY CALLERS, SET TO THE CALLER OF CVAF RETURN ADDRESS. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

FOR SVC CALLS, SET TO THE RESUME PSW OF THE RB THAT ISSUED SVC 139. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

In Figure C-163 we show a sample display of data from the tracking assistance for CVAFDSM instances.

<table>
<thead>
<tr>
<th>07.34.08 SYSTEM1</th>
<th>opdata,tracking</th>
</tr>
</thead>
<tbody>
<tr>
<td>07.34.08 SYSTEM1</td>
<td>CNZ1001I 07.34.08 TRACKING DISPLAY 673</td>
</tr>
<tr>
<td>STATUS=ON,ABEND NUM=1</td>
<td>MAX=1000 MEM=7T EXCL=5 REJECT=0</td>
</tr>
<tr>
<td>----TRACKING INFORMATION-----</td>
<td>-VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM</td>
</tr>
<tr>
<td>SMS-E:1 CVAFDSM STAT082</td>
<td>045201 CVAFJBN CVAFPGM xxx 28 4</td>
</tr>
</tbody>
</table>

Figure C-163  Tracking data from CVAFDSM exceptions

In Figure C-164 we show a sample exclusion list for CVAFDSM.

<table>
<thead>
<tr>
<th>* Tracking Information</th>
<th>Jobname</th>
<th>Pgmname</th>
</tr>
</thead>
<tbody>
<tr>
<td>* Mask</td>
<td>Mask</td>
<td>Mask</td>
</tr>
<tr>
<td>* SMS<em>CVAFDSM STAT082</em></td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

Figure C-164  CVAFDSM tracking assistant exclusion list sample

**CVAFFILT**

CVAFFILT was issued to obtain DSCB information for fully or partially qualified data set names on a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs and the qualified data set is ‘EAS eligible’. CVAF return code 4 and CVSTAT of X’56’ along with data set name status in the FCL (FCLDSNST) of X’06’ would have been set if the request was issued to a volume that supports extended attribute DSCBs.

When this instance occurs for a volume that does not support extended attribute DSCBs, it will be recorded in the tracker as an error message.

CVAFFILT processing will set the following tracking information:

TRPL_TRACK_INFO =

‘SMS-E:1 CVAFFILT STAT086 ’
TRPL_TRACK_DATA =

Byte 0
  CVAF Return Code = 4

Byte 1
  CVAF Status Code = STAT086 (X'56')

Byte 2
  FCL data set status code = X'06'

Byte 3
  Set to 01.

DFSMS Tracking category 1: EAV Migration. Error Message. CVAFFILT was issued to obtain DSCB information for fully or partially qualified data set names on a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK that it supports the extended attribute DSCBs and the qualified data set is 'EAS eligible'. CVAF return code 4 and CVSTAT of X'56' along with data set name status in the FCL (FCLDSNST) of X'06' would have been set if the request was issued to a volume that supports extended attribute DSCBs.

TRPL_VIOLATORS_ADDR =

FOR BRANCH ENTRY CALLERS, SET TO THE CALLER OF CVAF RETURN ADDRESS. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

FOR SVC CALLS, SET TO THE RESUME PSW OF THE RB THAT ISSUED SVC 139. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

In Figure C-165 we show a sample display of data from the tracking assistance for CVAFFILT instances.

| 07.34.08 SYSTEM1 d opdata,tracking |
| 07.34.08 SYSTEM1 CNZ1001I 07.34.08 TRACKING DISPLAY 673 |
| STATUS=ON,ABEND NUM=1 MAX=1000 MEM=7T EXCL=5 REJECT=0 |
| ----TRACKING INFORMATION----- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM |
| SMS-E:1 CVAFFILT STAT086 04560601 CVAFJBN CVAFPGM xxx 28 4 |

* Jobname Pgmname *
* Tracking Information Mask Mask Comments (ignored) *
|----------------------------+--------+--------+----------------------|
|SMS*CVAFFILT STAT086* |* |* | CVAFFILT |

Figure C-165  Tracking data from CVAFFILT exceptions

In Figure C-166 we show a sample exclusion list for CVAFFILT

CVAFVSM
CVAFVSM was issued to allocate space for a volume that is not an EAV. The caller did not specify with EADSCB=OK that it supports an EAV. CVAF return code 4 and CVSTAT of X'52' would have been set if issued to an EAV.
When this instance occurs for a non EAV, it will be recorded in the tracker as an error message.

Note that the CVAFVSM interface is an internal system function that is not documented externally for general use.

CVAFVSM processing will set the following tracking information:

TRPL_TRACK_INFO =
   ‘SMS-E:1 CVAFVSM STAT082 ‘

TRPL_TRACK_DATA =
   Byte 0
      Set to zero, not used.
   Byte 1
      CVAF Return Code = 4
   Byte 2
      CVAF Status Code = STAT082 (X’52’) 
   Byte 3
      Set to 01

   DFSMS Tracking category 1: EAV Migration. Error message. CVAFVSM was issued to allocate space for a volume that is not an EAV. The caller did not specify with EADSCB=OK That it supports an EAV. CVAF return code 4 and CVSTAT of X’52’ would have been set if issued to an EAV. CVAFVSM interface is an internal system function that is not documented externally for general use.

   When this instance occurs for a non EAV, it will be recorded in the tracker as an error message.

TRPL_VIOLATORS_ADDR =
   FOR BRANCH ENTRY CALLERS, SET TO THE CALLER OF CVAF RETURN ADDRESS. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

   FOR SVC CALLS, SET TO THE RESUME PSW OF THE RB THAT ISSUED SVC 139. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS

In Figure C-167 we show a sample display of data from the tracking assistance for CVAFVSM instances.

| 07.34.08 SYSTEM1 d opdata,tracking  |
| 07.34.08 SYSTEM1 CNZ1001I 07.34.08 TRACKING DISPLAY 673 |
| STATUS=ON,ABEND NUM=1 MAX=1000 MEM=7T EXCL=5 REJECT=0 |
| ----TRACKING INFORMATION---- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM |
| SMS-E:1 CVAFVSM STAT082 045201 CVAFJBN CVAFPGM xxx 28 4 |

*Figure C-167  Tracking data from CVAFVSM exceptions*
In Figure C-168 we show a sample exclusion list for CVAFVSM.

<table>
<thead>
<tr>
<th>*</th>
<th>Tracking Information Mask</th>
<th>Mask</th>
<th>Mask</th>
<th>Comments (ignored)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS<em>CVAFVSM STAT085</em></td>
<td>*</td>
<td>*</td>
<td>CVAFVSM</td>
<td></td>
</tr>
</tbody>
</table>

**Figure C-168  CVAFVSM tracking assistant exclusion list sample**

**DCB Open of a VTOC**

A DCB Open of a VTOC was issued to a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK on the DCBE macro that it supports the extended attribute DSCBs in the VTOC. Open would have issued an ABEND, MSGIEC142I 113-48 if an attempt was made to open the VTOC of a volume that supported extended attribute DSCBs.

When this instance occurs for a volume that does not support extended attribute DSCBs, it will be recorded in the tracker as an error message.

OPEN processing will set the following tracking information:

**TRPL_TRACK_INFO** =

'SMS-E:1 DCB OPEN VTOC 113-48'

**TRPL_TRACK_DATA** =

Byte 0-2
Set to zero, not used.

Byte 3
Set to 01

DFSMS Tracking category 1: EAV Migration. Error Message. A DCB Open of a VTOC was issued to a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK on the DCBE macro that it supports the extended attribute DSCBs in the VTOC. Open would have issued an ABEND, MSGIEC142I 113-48 if an attempt was made to open the VTOC of a volume that supported extended attribute DSCBs.

**TRPL_VIOLATORS_ADDR** =

SET TO THE RESUME PSW OF THE RB THAT ISSUED THE OPEN SVC. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

In Figure C-169 we show a sample display of data from the tracking assistance for DCB Open of a VTOC instances.

<table>
<thead>
<tr>
<th>Jobname</th>
<th>Pgmname</th>
<th>Tracking Information Mask</th>
<th>Mask</th>
<th>Mask</th>
<th>Comments (ignored)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS<em>CVAFVSM STAT085</em></td>
<td>*</td>
<td>*</td>
<td>CVAFVSM</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure C-169  Tracking data from DCB Open of a VTOC exceptions**
In Figure C-170 we show a sample exclusion list for DCB Open of a VTOC.

<table>
<thead>
<tr>
<th>Tracking Information Mask</th>
<th>Jobname</th>
<th>Pgname</th>
<th>Comments (ignored)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS<em>DCB OPEN VTOC</em></td>
<td>*</td>
<td>*</td>
<td>MSGIEC142I 113-48</td>
</tr>
</tbody>
</table>

**DCB Open of EAS eligible data set (VSAM)**

A DCB Open (MACRF = E for EXCP) of an EAS eligible data set (VSAM) was issued to a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK on the DCBE macro that it supports the extended attribute DSCBs for an EAS eligible data set. Open would have issued an ABEND, MSGIEC142I 113-44 if an attempt was made to open the EAS eligible data set on a volume that supported extended attribute DSCBs.

When this instance occurs for a volume that does not support extended attribute DSCBs, it will be recorded in the tracker as an error message.

OPEN processing will set the following tracking information:

**TRPL_TRACK_INFO** =

'SMS-E:1 DCB OPEN VSAM 113-44'

**TRPL_TRACK_DATA** =

Byte 0-2

Set to zero, not used.

Byte 3

Set to 01.

DFSMS Tracking category 1: EAV Migration. Error Message. A DCB Open (MACRF = E for EXCP) of an EAS eligible data set (VSAM) was issued to a volume that does not support extended attribute DSCBs. The caller did not specify with EADSCB=OK on the DCBE macro that it supports the extended attribute DSCBs for an EAS eligible data set. Open would have issued an ABEND, MSGIEC142I 113-44 if an attempt was made to open the EAS eligible data set on a volume that supported extended attribute DSCBs.

**TRPL_VIOLATORS_ADDR** =

SET TO THE RESUME PSW OF THE RB THAT ISSUED THE OPEN SVC. TRACKER CODE WILL DETERMINE JOB AND PROGRAM NAMES FROM THIS ADDRESS.

In Figure C-171 we show a sample display of data from the tracking assistance for DCB Open of EAS eligible data set instances.

```
07.34.08 SYSTEM1 d opdata,tracking
07.34.08 SYSTEM1 CNZ1001I 07.34.08 TRACKING DISPLAY 673
STATUS=ON,ABEND NUM=1 MAX=1000 MEM=7T EXCL=5 REJECT=0
-----TRACKING INFORMATION----- -VALUE-- JOBNAME PROGNAME+OFF-- ASID NUM
SMS-E:1 DCB OPEN VSAM 113-44 01 OPENJBN OPENPGM xxx 28 4
```

**Figure C-171** Tracking data from DCB Open of EAS eligible data set (VSAM) exceptions
In Figure C-172 we show a sample exclusion list for DCB Open of EAS eligible data set.

<table>
<thead>
<tr>
<th>Tracking Information Mask</th>
<th>Jobname</th>
<th>Pgmname</th>
<th>Comments (ignored)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMS<em>DCB OPEN VSAM</em></td>
<td>*</td>
<td>*</td>
<td>MSGIEC142I 113-44</td>
</tr>
</tbody>
</table>

Figure C-172  DCB Open of EAS eligible data set (VSAM) tracking assistant exclusion list sample

Figure C-173 shows an example of a display of trace data collected by the Migration Assistance Tracker:

```
D OPDATA,TRACKING
CNZ1001I 11.08.01 TRACKING DISPLAY 160
STATUS=ON  NUM=92  MAX=1000  MEM=00  EXCL=0  REJECT=0

-----TRACKING INFORMATION-----  -VALUE-- JOBNAME  PROGNAME+OFF-- ASID  NUM
SMS-I:3 DEVTYPE  03 *MASTER*  ICHSEC06  1DEF8  01  2
SMS-I:3 DEVTYPE  03 BOBH   ISPICP   80   6A  1
SMS-I:3 DEVTYPE  03 BOBH   ISRUDA   3834C   6A  3
SMS-I:3 DEVTYPE  03 DFHSM70  ARCCITL  227FC4  69  2
SMS-I:3 DEVTYPE  03 DFRMM  EDGPARM  258   5F  1
SMS-I:3 DEVTYPE  03 DFRMM  EDGPD0  108   5F  1
SMS-I:3 LSPACE EXPMSG=  803 MHLRES1  ISRUDA   3765E  6F  5
SMS-I:3 LSPACE EXPMSG=  803 MHLRES3  ISRUDA   3765E  6D  7
SMS-I:3 LSPACE EXPMSG=  803 MHLRES5  ISRUDA   3765E  6A  1
SMS-I:3 LSPACE EXPMSG=  803 MHLRES5  ISRUDA   3765E  72  6
SMS-W:2 IDCAMS LISTCAT  02 MHLRES5K IDCAMS   E866  1D  1

TO REPORT THESE INSTANCES, SEND THIS MESSAGE VIA E-MAIL TO CONSOLESBUS.IBM.COM. FOR ADDITIONAL INFORMATION OR TO OBTAIN A CURRENT EXCLUSION LIST, SEE APAR II13752.
```

Figure C-173  Example of Migration Assistance Output from D opdata,tracking command

The information returned by the EAV Migration Assistance Tracker describes the occurrence of an instance in text. Like the Console ID Tracking facility, the EAV Migration Assistance Tracker returns tracking information and a tracking value. The tracking information can be from 1 to 28 characters in length and can be set to any EBCDIC value. The tracking value is four bytes of binary data associated with this instance.

Exclusion list

To reduce the size of the tracking list, an exclusion list facility is available which continues to be able to be used to suppress tracking of console messages, but has been extended for DFSMS EAV support.

For DFSMS these values include data to associate an instance to a specific DFSMS function and to define the reason for the instance being recorded. This standard allows for maximum flexibility in defining exclusion records that apply to DFSMS records. IBM provides an exclusion list to prevent common instances from being recorded. It is filtered based on the following information:

- Tracking Information Mask
- Job Name Mask
- Program Name Mask
Download the most recent exclusion list to your SYS1.PARMLIB(CNIDTRxx) member from the Web site:

http://www-03.ibm.com/systems/z/os/zos/downloads/

(Look for: z/OS consoles 1-byte tracker exclusion list.)

From entries tracked, it should be possible to identify the possible improper use of returned information, like:

- Parsing 28-bit cylinder numbers in output as 16-bit cylinder numbers (Warning) in programs like: IEHLIST LISTVTOC, IDCAMS LISTCAT, IDCAMS LISTDATA PINNED.
  - Identify programs that might want to use new services (Informational) such as: LSPACE, DEVTYPE, IDCAMS and DCOLLECT.

Tracking Information: SMS- to represent this as a DFSMS instance, “E, W, or I” for error, warning or informational - followed by a colon numeric categorized reason for the recorded instance. Remaining tracking information set to any EBCDIC value.

A recommended planning activity should be to look into any code doing functions such as:

- OBTAIN, REALLOC, CVAFDIR, CVAFSEQ, CVAFDSM, CVAFVSM and CVAFFILT macro calls.
- Programs that calculate volume or data set size by any means, including reading a VTOC or VTOC index directly with a BSAM or EXCP DCB. The system cannot distinguish between programs that read DSCBs for space information and those that read them for metadata. Both types of programs require attention.
- EXCP and STARTIO macros for DASD channel programs and other programs that examine DASD channel programs or track addresses.
- Programs that examine any of the many operator messages that contain a DASD track or block address or data set or volume size. Such track or block addresses generally are represented in the documentation as cchh, cchhr, mbbccchhr or track-address.
- Programs that access the INDEX and VVDS data sets.

After the reason for appearance of an entry in the tracking list has been identified and remedial action initiated, the exclusion list can be updated to prevent the entries re-appearing.

Use the new TRKADDR macro for all track address comparisons and calculations. Programs should not need to do 28-bit manipulation themselves. Use this for all track address computations even those not directly affected by this support for VSAM data sets using EAS. This prepares one for additional support in the future. For programs that are written in a high level language such as C, C++, COBOL or PL/I a new IECTRKAD routine can be called to execute the same functions as in TRKADDR macro.
Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks publications

For information about ordering these publications, see “How to get Redbooks publications” on page 661. Note that some of the documents referenced here may be available in softcopy only.

- z/OS V1R8 DFSMS Technical Update, SG24-7435

Other publications

These publications are also relevant as further information sources:

- DFSMS Storage Administrator Reference, SG24-7402
- z/OS MVS Programming: Authorized Assembler Services Guide, SA22-7608
- MVS Setting Up a Sysplex, SA22-7625
- z/OS Migration, GA22-7499
- z/OS Planning for Installation, GA22-7504

Online resources

These Web sites are also relevant as further information sources:

- DFSMSrmm installation wizard:
- DFSMSrmm CIM location:
  [http://www.openpegasus.org](http://www.openpegasus.org)

How to get Redbooks publications

You can search for, view, or download Redbooks publications, Redpapers publications, Technotes, draft publications, and Additional materials, as well as order hardcopy Redbooks publications, at this Web site:

[ibm.com/redbooks](http://ibm.com/redbooks)
Help from IBM

IBM Support and downloads:
ibm.com/support

IBM Global Services:
ibm.com/services
Index

Numerics
3390 Model A 354
3592 Model E05 115
3-way audit 67

A
Abend 878 reduction 24, 150
absolute block address 310
absolute track addresses 309
access bias 7
ACCESSIBILITY parameter 20
ACS routines 348
ADDDATASET dsname 86
AddDS 125
ADDVOLUME DSNAME 86
ADRXMAIA application interface 24
alias address considerations 353
alias considerations 353
allocating data sets 315
ANTMAIN parmlib member 164
ANTMIN00 PARMLIB member 164
ANYUSE definitions 285
Apache Tomcat server 298
APAR A22500 162
APAR II3752 436
APAR II4250 15, 427
APAR OA13742 25, 420
APAR OA15698 117, 393
APAR OA16552 390
APAR OA16562 117, 391
APAR OA16912 417
APAR OA17574 395
APAR OA17965 84, 387
APAR OA18184 419
APAR OA18455 397
APAR OA18455 117
APAR OA18720 433
APAR OA20169 422
APAR OA20749 377
APAR OA2078 416
APAR OA2449 406
APAR OA22452 414
APAR OA26265 409
APAR OA2627 411
APAR OA26263 413
APAR OA2738 38, 385
APAR OA2804 156, 398
APAR OA29900 402
APAR OA29903 408
APAR OA2806 377
APAR OA29595 424
APAR OA24276 376
APAR OW43870 218
APAR PK48337 425
APAR PK58292 346, 404
ARC1MIGR address space 24
ARCFSR macro 26
ARCHBACK macro 150
ARCHRCAL service 29
ARCRPEXT exit 27
AS DSNAME 86
ATL 254
CBRUXENT installation exits 254
EDGUX200 installation exit 254
shared CDS and TCDB 254

B
BACK 24
base addressing space 308
best-fit allocation 22
BPV
determining 340
BPXPRMxxx 124
break point value 305
BreakPointValue 314
Storage Group attribute 314
BSAM data buffers 4
BSAM support 5
Buffer 129

C
capacity planning 352
CBROOMxx 42
CBROOMxx member 187
CBROOMxx parmlib member 43
CBRSRMR19 40
CBRSRMR19 job
SAMPLIB
CBRSRMR19 41
CBRXENT exit 258, 289
CBRXLCS 67
CD NEWDSNAME 86
CDSB 24
CDSID enqueue setting 84
CFRM policy 175
change use attribute 62
CHANGEVRS DSNAME 86
chkdvol command 366
CIM agent 293
CIM provider 117
CIM schema classes 117
CLIST Enhancements 117
code samples
SMG85TP 496
commands
D SMS, CFLS 178
D SMS, SMSVAM 178
DISPLAY HPSTATS 140
MODIFY OAM 184, 188
MODIFY OAM,UPDATE 184
OSREQ QUERY 184
TSO LISTCONTROL 209

Common Information Model (CIM) 117
Concurrent Copy 164
COPY Services 345, 363
Coupling Facility 174
cross-memory function 25
CSDID serialization 78
Ctrace 128
CV AF DNAME 86
CV AF macros 497
CV AF DIR 321
CV AF DIR macro 497, 530
CV AF DIR processing 321
CV AF DSM macro 497, 560
CV AF FILT macro 497, 572
CV AF FILT service 580
CV AF SEQ macro 497, 599
CV AF TST macro 497
CVDIR1EJ job 557
CV DSM1O program 570
CV SEQ service 602
CV SEQ8O program 617
cylinder-managed space 306

D
DADSM macros 497
daily statistics records (DSR) 26
data class space override 134
data extent block 5
Data Set Name Mask 90
data set name rules,DFSMSrmm
data set name rules 89
DB2 considerations 346
DCB enhanced 623
DCB extension macro 4
DCB OPEN of VTOC 619
DCBE macro 4
DCBE option
FIXED=USER 4
DCOLLECT 314, 316
Volume Record (V) 340
DEFRAG 160
MAXTIME keyword 342
VERSION 1 keyword 341
DEFRAG function 160, 318, 341
MMOVPC T 318
DEFRAG Version 1 341
DEFRAG Version 2 318
DEVSERV PATHS command 350, 360
DEVSERV QDASD 350
DEVSERV QDASD command 360
DEVTYPE macro 324
DFS MS
QUERY ACTIVE command 51
DFS MS QUERY ACTIVE 51
DFS MS VERSION 1 346
EDGHSKP inventory management 50
PDS enhancements 37
DFS MS V1.10 134
constraint relief messages 137
informational messages 137
DFS MS V1.9
BSAM 4
BSAM support 5
performance enhancements 4
QSAM support 5
sample code 441
system page data set 9
DFS MS volume allocation 21
DFS MS dfp
maintenance 15
DFS MS dfp V1.9 3
DFS MS ds datamover 320
DFS MS dss
ADR556W message 162
CONSOLIDATE 160
data movement preference 169
DEFRAG 160
EAV compatibility 162
large block interface function 25
message ADR129E 159
NEWNAMEUNCONDITIONAL 158
DFS MS dss considerations 346
DFS MS dss CONSOLIDATE RACF profile 161
DFS MS dss COPY 196
DFS MS dss Fast Replication
SFI considerations 22
DFS MS dss Fast Replication works 20
DFS MS dss invoked by DFSMS Sh sm 24
DFSMS Sh sm 45
ARCRPEXT exit 27
concurrent copy 155
cross-memory function 25
cross-memory service 24
EAV 370
EAV considerations 156
extent reduction 27
GENVSAMCOMP NAMES 153
HBDELETE command 154
hourly space management 344
NEWNAME parameter 150, 153
RECOVER 154
REPORT command 27
SDSF display 25
space management 343
STGADMIN profile 153
Index 665

System Logger 28
system recalls 28
TAPE COPY 156
track-managed threshold 343
VERSIONS subparameter 154
XCF requests 154
DFSMShsm address spaces 24
DFSMShsm patch
cross-memory function 25
DFSMsrm 49
&SYSCLONE symbol 95
APAR OA16524 117
APAR OA17965 84
association classes 118
CANCEL command 56
CDS DSS copy 196
CDS serialization 78
CDSID parmlib option 79
changed TSO subcommands 86
chunking returned entries 117
CIM provider
120
CLIST data set 101
CLIST parameter 99
continue_information 103
EDG@CONT variable 112
EDG6101E message 61
EDGGDSNM migration aid 94
EDGRRRPTE reporting exec 115
EEFMT2 dataclass 115
enhancements 195
GRSRNlx consideration 83
invokeMethod 119
ISPF considerations 113
ISPF panels 99
LISTCONTROL option 96
LISTCONTROL subcommand 278
location names 61
MEMBER operand 95
MEMBER option 97
operator commands 51
pending release 114
quoted data set name strings 84
RACF resource 94
RACF SECELEV 99
REJECT commands 285
RELEASE command 56
REPORT17 115
RMMCLIST data set 99
SEARCH subcommand 100
SEARCHBIN command 105
SEARCHDATASET command 106
SEARCHOWNER command 107
SEARCHPRODUCT command 107
SEARCHRACK command 108
SEARCHVOLUME command 109
SearchVolume command 114
SEARCHVRS command 110
SEARCHVRS operand 114
STOP command 56
STORAGEGROUP parameter 114
SYZRMM MASTER.RESERVE 78
TCDB 255
TSO subcommand parsing 98
volume chains 118
VRSEL processing 92
DFSMsrm commands
QUERY ACTIVE 55
DFSMsrm messages
EDG6007E 82
DFSMsrm messages
EDG0204I 97
EDG0237E message 81
EDG1107D 57
EDG1108E 58
EDG1122I 58
EDG1123I 59
EDG3012I 103, 110
EDG3025I 104, 111–113
EDG3025I 112
EDG3201I 89
EDG3203I 103, 110
EDG3360E 102
EDG6414E 82
EDG6846I 66
EDG6901I 83
DFSMsrm utilities
EDGHSKP 59
EDGSPLCS 68
EDGUTIL 59
DFSMsrm V1.1.0
report generator 232
DFSMsrm V1.10
ANYUSE definitions 285
ATL 254
BACKUP (COPY) 197
CBRUXENT exit 258
CIM agent 293
ccontrol data set 201
creating reports 215
dDSS copy services 196
EDG1017D message 288
EDG2106D message 202
EDG2243I message 212
EDG6846I 66
EDG6901I 83
definitions 285
EDGMUPD usermod 218
EXPDT-retained volume 205
Extract Data Set 217
Fast Replication 196
forward recover 202
library partitioning 253
LISTCONTROL OPENRULE 269
LISTCONTROL OPTION 204
MEDINF command 218
MEDINF commands 224
OpenPegasus 295
OPENRULE command 286
percent used information 231
PTITION command 285
RACF FACILITY profiles 287
REJECT commands 285
<table>
<thead>
<tr>
<th>Index</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDGUX200 exit</td>
<td>289</td>
</tr>
<tr>
<td>Enterprise Encrypted Format 2 (EEFMT2)</td>
<td>115</td>
</tr>
<tr>
<td>Enterprise Format 1 (EFMT1)</td>
<td>115</td>
</tr>
<tr>
<td>Enterprise Format 2 (EFMT2)</td>
<td>115</td>
</tr>
<tr>
<td>enumerateInstanceNames</td>
<td>120</td>
</tr>
<tr>
<td>EXPDTDROP</td>
<td>205</td>
</tr>
<tr>
<td>DFSMSrmm r V1.11EXPDTDROP</td>
<td>205</td>
</tr>
<tr>
<td>EXPDTDROP limit</td>
<td>205</td>
</tr>
<tr>
<td>EXPROC</td>
<td>59, 215</td>
</tr>
<tr>
<td>EXPROC command</td>
<td>60</td>
</tr>
<tr>
<td>EXPROC processing</td>
<td>60</td>
</tr>
<tr>
<td>extended address volume</td>
<td>304</td>
</tr>
<tr>
<td>Extended Addressing Space</td>
<td>308</td>
</tr>
<tr>
<td>extended attribute DSCB</td>
<td>304, 311</td>
</tr>
<tr>
<td>Extent Reduction</td>
<td>27</td>
</tr>
<tr>
<td>EXTRDCTN</td>
<td>28</td>
</tr>
<tr>
<td>F</td>
<td>55</td>
</tr>
<tr>
<td>F DFRMM,M=xx</td>
<td>55</td>
</tr>
<tr>
<td>F OAM,DISPLAY command</td>
<td>44</td>
</tr>
<tr>
<td>F OAM,RESTART</td>
<td>42</td>
</tr>
<tr>
<td>fallback from multiple lock structure</td>
<td>182</td>
</tr>
<tr>
<td>Fast Replication</td>
<td>196</td>
</tr>
<tr>
<td>FIXED=USER</td>
<td>4</td>
</tr>
<tr>
<td>FORCE keyword</td>
<td>192</td>
</tr>
<tr>
<td>free space statistics</td>
<td>313</td>
</tr>
<tr>
<td>FSR type 18</td>
<td>26</td>
</tr>
<tr>
<td>FSR type 19</td>
<td>26</td>
</tr>
<tr>
<td>FSRF_RECALL_TAKEAWAY bit</td>
<td>26</td>
</tr>
<tr>
<td>FSRSTAT program</td>
<td>26</td>
</tr>
<tr>
<td>FSRTYPE</td>
<td>26</td>
</tr>
<tr>
<td>function statistics record</td>
<td>26</td>
</tr>
<tr>
<td>Functional Statistics Record (FSR)</td>
<td>26</td>
</tr>
<tr>
<td>G</td>
<td>345</td>
</tr>
<tr>
<td>Global Mirror</td>
<td>345</td>
</tr>
<tr>
<td>keyword XTNTLSTX</td>
<td>345</td>
</tr>
<tr>
<td>Global Resource Serialization (GRS)</td>
<td>83</td>
</tr>
<tr>
<td>GRSRNlx parmlib member</td>
<td>83</td>
</tr>
<tr>
<td>H</td>
<td>154</td>
</tr>
<tr>
<td>HBDELETE command</td>
<td>154</td>
</tr>
<tr>
<td>I</td>
<td>122</td>
</tr>
<tr>
<td>IBM WebSphere Application Server</td>
<td>122</td>
</tr>
<tr>
<td>IBMmm_Control</td>
<td>118</td>
</tr>
<tr>
<td>IBMmm_PolicyRule</td>
<td>118</td>
</tr>
<tr>
<td>IBMmm_Product</td>
<td>118</td>
</tr>
<tr>
<td>IBMmm_SearchOperands</td>
<td>119</td>
</tr>
<tr>
<td>ICKDSF</td>
<td>360</td>
</tr>
<tr>
<td>REFVTTOC parameter</td>
<td>365</td>
</tr>
<tr>
<td>ICKDSF initialization</td>
<td>360</td>
</tr>
<tr>
<td>IDCAMS DEFINE PAGESPACE</td>
<td>9</td>
</tr>
<tr>
<td>IDCAMS LISTCAT</td>
<td>10</td>
</tr>
<tr>
<td>IDCNOGFL alias</td>
<td>10</td>
</tr>
<tr>
<td>IEC161I</td>
<td>7</td>
</tr>
<tr>
<td>IEC161I message</td>
<td>7</td>
</tr>
<tr>
<td>IG17279I message</td>
<td>316</td>
</tr>
<tr>
<td>IGDSMSxx</td>
<td>348</td>
</tr>
<tr>
<td>USEEAV parameter</td>
<td>348</td>
</tr>
<tr>
<td>IGWLOCK00 lock table</td>
<td>174</td>
</tr>
<tr>
<td>Increase Capacity panel</td>
<td>361</td>
</tr>
<tr>
<td>Initial Access Response Seconds (IARS)</td>
<td>41</td>
</tr>
<tr>
<td>installation wizard</td>
<td>97</td>
</tr>
<tr>
<td>Interval Migration</td>
<td>344</td>
</tr>
<tr>
<td>IRMM</td>
<td>296</td>
</tr>
<tr>
<td>ISPF</td>
<td>34</td>
</tr>
<tr>
<td>configuration table</td>
<td>34</td>
</tr>
<tr>
<td>viewing UNIX files</td>
<td>34</td>
</tr>
<tr>
<td>ISPF Data Set Utility View</td>
<td>35</td>
</tr>
<tr>
<td>totaling of tracks</td>
<td>35</td>
</tr>
<tr>
<td>ISPF panels</td>
<td>99</td>
</tr>
<tr>
<td>ISPF profile</td>
<td>35</td>
</tr>
<tr>
<td>system symbolics</td>
<td>35</td>
</tr>
<tr>
<td>ISPF profile sharing</td>
<td>34</td>
</tr>
<tr>
<td>J</td>
<td>7</td>
</tr>
<tr>
<td>JCL AMP keyword, MSG=SMBBIAS</td>
<td>7</td>
</tr>
<tr>
<td>L</td>
<td>42</td>
</tr>
<tr>
<td>L2DATACLASS</td>
<td>42</td>
</tr>
<tr>
<td>L2DATACLASS keyword</td>
<td>42</td>
</tr>
<tr>
<td>L2TAPEUNITNAME</td>
<td>42</td>
</tr>
<tr>
<td>L2TAPEUNITNAME(unitname)</td>
<td>42</td>
</tr>
<tr>
<td>Label Anomaly processing</td>
<td>117</td>
</tr>
<tr>
<td>large format data set support</td>
<td>16</td>
</tr>
<tr>
<td>LDFSMSrmm V1.10</td>
<td>203</td>
</tr>
<tr>
<td>LISTCONTROL command</td>
<td>203</td>
</tr>
<tr>
<td>LEVEL subcommand</td>
<td>98</td>
</tr>
<tr>
<td>library partitioning</td>
<td>253</td>
</tr>
<tr>
<td>LIBSERV</td>
<td>68</td>
</tr>
<tr>
<td>LISTCAT</td>
<td>10</td>
</tr>
<tr>
<td>LISTCAT LEVEL processing</td>
<td>10</td>
</tr>
<tr>
<td>LISTCONTROL OPENRULE</td>
<td>269</td>
</tr>
<tr>
<td>LISTWS command</td>
<td>171</td>
</tr>
<tr>
<td>location names</td>
<td>61</td>
</tr>
<tr>
<td>lock set</td>
<td>173–174, 176</td>
</tr>
<tr>
<td>Lock Update parameter</td>
<td>176</td>
</tr>
<tr>
<td>Lock_Release</td>
<td>129</td>
</tr>
<tr>
<td>Lock_Request</td>
<td>129</td>
</tr>
<tr>
<td>Lock_Result</td>
<td>129</td>
</tr>
<tr>
<td>Locking</td>
<td>127</td>
</tr>
</tbody>
</table>
OBTMODO subroutine 511
ODLOCFL column 45
OpenPegasus 295
OpenPegasus CIM Server 2.5.3 122
OpenPegasus CIMOM 117
OPENRULE command 285
optical 40
OSREQ API 184, 192
OSREQ macro 40
OSREQ QUERY processing 184
out of space abends 347

P
page data set usage 9
page fixing 5
PARTREL macro 497
PDS
Volcount restriction 38
PDSE
Display HSPSTATS 140
LRUtime 148
SMSPDSE address space 141
Storage Utilization 148
UNMANAGED data sets 142
Volcount restriction 38
VSTOR command 147
PDSE enhancements 139
Pegasus CIM Server 120
Pegasus CIM Server 2.5.3 121
performance issues 367
PGM=IDCNOGFL 11
PGSER FIX macro 5
PLISTVER keyword 324
pre-DFSMS V1.10 system 364
primary space management 27
primary threshold 316
PROFILE PREFIX 88
PRTITION command 285, 288
PTF OA21487 380
public 128

Q
QSAM
BUFNO value 6
QSAM support 5
QUERY ACTIVE command 51
quiesce the DFSMSrmm subsystem 57
quoted data set names
DFSMSrmm
quoted data set names 87

R
RACF class FACILITY 94
RACF Data Labeling 126
RACF FACILITY class 268
RACF FACILITY profiles 287
RACF profile for DFSMSdss 161
RACF profiles
STGADMIN.EDG.* profile 271
RCVR 24
REALLOC macro 497
RECALL request
System Logger 28
Redbooks Web site 661
Contact us xvi
REFORMAT command 365
REFVTOC parameter 365
REJECT 95
REJECT commands 285
relative block address 310
REPORT 17 98
report generator 93, 232
REST 24
restart DFRMM 97
Return Priority (RP) 27
REXX exec
EDG@CONT variable 112
RLS
delete lock structure 176
DFSMSStvs 174
enabling multiple lock structures 177
IGW526A message 182
IGWLOCKxx table 174
lock set 176
Lock Update parameter 176
multiple lock structures 177
recoverable 174
secondary lock definition 176
SHCDS 175
Storage Class 175
RLS*CFRM policy 175
RMM CIM agent
V1R9 considerations 295
RMM CIM classes 117
RMM CIM provider 117
RMM CIM provider v1.9 122
RMM SUBCOMMANDS 85–86
RMM subcommands
data set name mask 88
RMM TSO CHANGEDATASET 87
RMM Web Service 112
RPCSEC_GSS Security 127
RPTEXT command options 215

S
SAF or SAFEXP 130
sample code 441
CVAFDIR macro 530
CVAFDSM macro 560
CVAFFILT macro 572
CVAFSEQ macro 599
CVDIR1EJ program 557
CVDSM10 program 570
CVSEQ80 program 617
DCB enhanced 623
DCB OPEN of VTOC 619
DFSMS V1.10 495
LSPACEE program 502
ADR735W message 171
ADR888l message 171
CONCURRENT keyword 168
LISTWS command 171
overview 164
VCC enhancement 164
VCINOUT file 68
VERIFY(SMSTAPE), MEND 67
VERIFY/MEND(VOL) 65
VERSIONS subparameter 154
Virtual Concurrent Copy 163
vital record specification
  VRSEL processing 92
VLPOOL 95
volume expansion 363
volume selection 317
volume statistics records (VSR) 26
volume-level threshold 343
VRSDROP limit, 205
VRSDROP processing
  DFSMSrmm V1.10
  VRSDROP processing 207
VRSEL 61
  processing data set names 93
VRSRETAIN processing 208
VSAM RLS 48
VSAM RLS*lock set 173–174
VSAM striping considerations 315
VTOC index block size 320
VTOC index space map 320
VTOC management 496
VTOC records 321
VTOC REFORMAT
  SYSLOG 365
VVDS
  EAV 325
  VVDS structure 325

W
Working Space Data Sets 165
WORM volumes 230
WSDS
  naming convention 166
  RACF protection 167
  sample definition 166
WTO IEA486E 218

X
XCF requests 154

Z
z/OS CIM Server v1.9 122
z/OS NFS RPCSEC 124
Z/OS V1.9
  RLS dump space 48
z/OS V1.9
  Program Binder 38
DFSMS V1.10 and EAV Technical Guide

Learn the function and features of z/OS V1R9 and V1R10

Each release of DFSMS builds upon the previous version to provide enhanced storage management, data access, device support, program management, and distributed data access for the z/OS platform in a system-managed storage environment.

This IBM Redbooks publication provides a summary of the functions and enhancements in z/OS V1R9 DFSMS. It then provides an in-depth technical description of the functions and enhancements in z/OS V1R10 DFSMS. It provides you with the information that you need to understand and evaluate the content of this DFSMS release, along with practical implementation hints and tips. Also included are enhancements that were made available through an enabling PTF that has been integrated into z/OS DFSMS V1R10.

z/OS V1R10 provides Extended Address Volumes (EAV), a capability that enables support for over 54 GB of addressable storage, removing a limitation in place today. Initially EAV is planned to support up to 262,668 cylinders (up to 223 GB of addressable storage) per volume, allowing you to simplify storage management by providing the ability to manage fewer, larger volumes as opposed to many small volumes. This book provides detailed information about planning, implementing and migrating to EAVs.

This book was written for storage professionals and system programmers who have experience with the components of DFSMS. It provides sufficient information so that you can start prioritizing the implementation of new functions and evaluating their applicability in your DFSMS environment.

Plan for EAV volumes and migrate to them

Includes worked implementation examples