z/OS

# DFSMS Access Method Services for Catalogs

SC26-7394-01

z/OS

# DFSMS Access Method Services for Catalogs

SC26-7394-01

**Second Edition, March 2002**

This edition applies to Version 1 Release 3 of z/OS™ (5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC26-7394-00.

Order publications through your IBM® representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

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

International Business Machines Corporation
RCF Processing, Department M86/050
5600 Cottle Road
San Jose, CA 95193-0001
United States of America

IBMLINK from US: STARPUBS at SJEVM5
IBMLINK from Canada: STARPUBS at TORIBM
IBM Mail Exchange: USIB3VVD at IBMMAIL
Internet: starpubs@us.ibm.com

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

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

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

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

**© Copyright International Business Machines Corporation 1973, 2002. All rights reserved.**
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

**Figures** . . . xiii

**Tables** . . . xv

**About This Book** . . . xvii
Required Product Knowledge . . . xvii
Referenced Publications . . . xvii
Accessing z/OS DFSMS Books on the Internet . . . xviii
Using LookAt to look up message explanations . . . xviii
Accessing Licensed Books on the Web . . . xix
How to Send Your Comments . . . xix
Notational Conventions . . . xx
How to Code Access Method Services Commands . . . xxi
Commands . . . xxi
Positional and Keyword Parameters . . . xxi
How to Code Subparameters . . . xxiii
Alphanumeric, National, and Special Characters . . . xxv
How to Continue Commands and Parameters . . . xxv
The Terminator . . . xxvi

**Summary of Changes** . . . xxvii
Summary of Changes for SC26-7394-01 z/OS Version 1 Release 3 . . . xxvii
New Information . . . xxvii
Deleted Information . . . xxvii
Changed Information . . . xxvii
Library Changes for z/OS Version 1 Release 3 . . . xxvii

**Chapter 1. Using Access Method Services** . . . 1
Identifying Data Sets and Volumes . . . 1
Dynamic Allocation . . . 2
JCL DD Statements . . . 3
Direct Allocation Using JCL . . . 4
Invoking Access Method Services . . . 4
As a Job or Jobstep . . . 5
From a Time Sharing Option Session . . . 5
Access Method Services Tape Library Support . . . 7
Summary of Tape Library Support . . . 7
Access Method Services Commands for Tape Library Support . . . 7
Order of Catalog Use . . . 9
Catalog Search Order for ALTER . . . 9
Catalog Selection Order for BLDINDEX . . . 10
Catalog Selection Order for DEFINE . . . 10
Catalog Search Order for DELETE . . . 10
Catalog Selection Order for EXPORT DISCONNECT . . . 12
Catalog Search Order for LISTCAT . . . 12
Generic Catalog Selection for DELETE and LISTCAT . . . 13
Specifying Attribute Selection Order . . . 14

**Chapter 2. Modal Commands** . . . 15
IF-THEN-ELSE Command Sequence . . . 15
Using Nested IF Commands: Example 1 . . . 17
Using Nested IF Commands: Example 2 . . . 17
Null Command . . . 17

© Copyright IBM Corp. 1973, 2002

DO-END Command Sequence . . . . . 18
Using the LASTCC Parameter . . . . . 18
SET Command . . . . . 18
Using the SET command and MAXCC Parameter . . . . . 19
CANCEL Command . . . . . 19
Using the CANCEL Command . . . . . 19
PARM Command . . . . . 19
Using the PARM Command: Example 1 . . . . . 21
Using the PARM Command: Example 2 . . . . . 22
Using the PARM Command: Example 3 . . . . . 22
Condition Codes . . . . . 23
Common Continuation Errors in Coding Modal Commands . . . . . 23

**Chapter 3. Functional Command Syntax** . . . . . 25
Functional Command Syntax Summary . . . . . 25

**Chapter 4. ALLOCATE** . . . . . 27
Restrictions . . . . . 28
Allocating Storage Management Subsystem Managed Data Sets . . . . . 28
Allocating Non-SMS Managed Data Sets . . . . . 29
Return Codes for the ALLOCATE Command . . . . . 30
Syntax for ALLOCATE Parameters . . . . . 30
Required Parameters . . . . . 32
Optional Parameters . . . . . 32
ALLOCATE Examples . . . . . 47
Allocate a Data Set Using SMS Class Specifications: Example 1 . . . . . 47
Allocate a VSAM Data Set Using SMS Class Specifications: Example 2 . . . 48
Allocate a New Data Set: Example 3 . . . . . 49
Allocate a non-VSAM Data Set: Example 4 . . . . . 50
Allocate a Partitioned Data Set Extended: Example 5 . . . . . 50

**Chapter 5. ALTER** . . . . . 53
Entry Types That Can Be Altered . . . . . 54
ALTER Parameters . . . . . 56
Required Parameters . . . . . 56
Optional Parameters . . . . . 57
ALTER Examples . . . . . 73
Alter a Cluster's Attributes Using SMS Keywords: Example 1 . . . . . 73
Roll-In a Generation Data Set: Example 2 . . . . . 73
Alter the Entry Names of Generically Named Clusters: Example 3 . . . . . 73
Alter the Attributes of a Generation Data Group: Example 4 . . . . . 74
Alter a Data Set Expiration Date: Example 6 . . . . . 74
Migrate a DB2® Cluster to a Linear Data Set Cluster: Example 7 . . . . . 74
Alter a Cluster Name and the Associated Data and Index Names: Example 8 75

**Chapter 6. ALTER LIBRARYENTRY** . . . . . 77
ALTER LIBRARYENTRY Parameters . . . . . 77
Required Parameters . . . . . 77
Optional Parameters . . . . . 77
ALTER LIBRARYENTRY Examples . . . . . 80
Altering a Tape Library Entry: Example 1 . . . . . 80
Altering a LIBRARY Entry: Example 2 . . . . . 80

**Chapter 7. ALTER VOLUMEENTRY** . . . . . 81
ALTER VOLUMEENTRY Parameters . . . . . 81
Required Parameters . . . . . 81

Optional Parameters . . . . . 81
ALTER VOLUMEENTRY Examples . . . . . 85
Altering a Volume Entry: Example 1 . . . . . 85
Altering a VOLUME Entry: Example 2 . . . . . 86

**Chapter 8. BLDINDEX** . . . . . 87
BLDINDEX Parameters . . . . . 87
Required Parameters . . . . . 87
Optional Parameters . . . . . 88
Calculating Virtual Storage Space for an Alternate Index . . . . . 90
DD Statements That Describe the Sort Work Files . . . . . 92
BLDINDEX Examples . . . . . 93
Build an Alternate-Index over a Key-Sequenced Data Set (KSDS): Example 1 93
Build an Alternate-Index over a Key-Sequenced Data Set (KSDS) Using DFSORT: Example 2 . . . . . 94

**Chapter 9. CREATE LIBRARYENTRY** . . . . . 95
Required Parameters . . . . . 95
Optional Parameters . . . . . 96
CREATE LIBRARYENTRY Examples. . . . . 98
Creating a Tape Library Entry: Example 1 . . . . . 98
Creating a LIBRARY Record: Example 2 . . . . . 98

**Chapter 10. CREATE VOLUMEENTRY** . . . . . 101
Required Parameters . . . . . 101
Optional Parameters . . . . . 101
CREATE VOLUMEENTRY Examples . . . . . 105
Creating a Tape Volume Entry: Example 1 . . . . . 105
Creating a VOLUME Entry: Example 2. . . . . 106

**Chapter 11. DCOLLECT** . . . . . 107
DCOLLECT Parameters . . . . . 108
Required Parameters . . . . . 109
Optional Parameters . . . . . 109
DCOLLECT in a Batch Environment. . . . . 112
Generic Volume Data Collection: Example 1. . . . . 112
Storage Group Data Collection: Example 2 . . . . . 112
Migrated and Backup Data Set Data Collection: Example 3 . . . . . 113
Combination of Options: Example 4 . . . . . 114
Collection of SMS Construct Information: Example 5 . . . . . 114

**Chapter 12. DEFINE ALIAS** . . . . . 117
DEFINE ALIAS Parameters . . . . . 117
Required Parameters . . . . . 117
Optional Parameters . . . . . 118
DEFINE ALIAS Examples . . . . . 118
Define Alias for a non-VSAM non-SMS-Managed Data Set: Example 1 118
Define an Alias for a User Catalog: Example 2 . . . . . 118

**Chapter 13. DEFINE ALTERNATEINDEX** . . . . . 121
DEFINE ALTERNATEINDEX Parameters . . . . . 122
Required Parameters . . . . . 122
Optional Parameters . . . . . 125
Data and Index Components of an Alternate Index . . . . . 135
DEFINE ALTERNATEINDEX Examples . . . . . 135
Define an Alternate Index Using SMS Data Class Specification: Example 1 135

Define an SMS-Managed Alternate Index: Example 2 . . . . . 135
Define an Alternate Index: Example 3 . . . . . 136
Define an Alternate Index with RECATALOG: Example 4 . . . . . 137

**Chapter 14. DEFINE CLUSTER** . . . . . 139
DEFINE CLUSTER Parameters . . . . . 141
Required Parameters . . . . . 141
Optional Parameters . . . . . 145
Data and Index Components of a Cluster. . . . . 160
DEFINE CLUSTER Examples . . . . . 160
Define an SMS-Managed Key-Sequenced Cluster: Example 1 . . . . . 160
Define an SMS-Managed Key-Sequenced Cluster Specifying Data and Index Parameters: Example 2 . . . . . 161
Define a Key-Sequenced Cluster Specifying Data and Index Parameters: Example 3 . . . . . 161
Define a Key-Sequenced Cluster and an Entry-Sequenced Cluster: Example 4. . . . . 162
Define a Relative Record Cluster in a Catalog: Example 5 . . . . . 163
Define a Reusable Entry-Sequenced Cluster in a Catalog: Example 6 . . . 164
Define a Key-Sequenced Cluster in a Catalog: Example 7 . . . . . 165
Define an Entry-Sequenced Cluster Using a Model: Example 8 . . . . . 166
Define a VSAM Volume Data Set: Example 9 . . . . . 166
Define a Relative Record Data Set with Expiration Date Beyond 1999: Example 10. . . . . 167
Define a Linear Data Set Cluster in a Catalog: Example 11 . . . . . 168

**Chapter 15. DEFINE GENERATIONDATAGROUP** . . . . . 169
DEFINE GENERATIONDATAGROUP Parameters . . . . . 169
Required Parameters . . . . . 169
Optional Parameters . . . . . 169
DEFINE GENERATIONDATAGROUP Examples . . . . . 171
Define a Generation Data Group and a Generation Data Set within it: Example 1 . . . . . 171
Use Access Method Services to Define a GDG and JCL to Define a GDS in that GDG: Example 2 . . . . . 172

**Chapter 16. DEFINE NONVSAM**. . . . . 173
DEFINE NONVSAM Parameters . . . . . 173
Required Parameters . . . . . 174
Optional Parameters . . . . . 176
DEFINE NONVSAM Examples . . . . . 178
Define a Non-VSAM Data Set with the RECATALOG Parameter: Example 1 178
Define a Non-VSAM Data Set: Example 2 . . . . . 178

**Chapter 17. DEFINE PAGESPACE** . . . . . 181
DEFINE PAGESPACE Parameters . . . . . 181
Required Parameters . . . . . 181
Optional Parameters . . . . . 182
DEFINE PAGESPACE Examples. . . . . 185
Define a NOSWAP Page Space: Example 1 . . . . . 186
Define a SWAP Page Space: Example 2 . . . . . 186

**Chapter 18. DEFINE PATH** . . . . . 187
DEFINE PATH Parameters . . . . . 187
Required Parameters . . . . . 187
Optional Parameters . . . . . 187

DEFINE PATH Examples . . . . . 189
Define a Path: Example 1 . . . . . 190
Define a Path (Recatalog) in a Catalog: Example 2 . . . . . 190

**Chapter 19. DEFINE USERCATALOG** . . . . . 191
DEFINE USERCATALOG Parameters . . . . . 192
Required Parameters . . . . . 192
Optional Parameters . . . . . 194
Data and Index Components of a User Catalog . . . . . 200
DEFINE USERCATALOG Examples . . . . . 200
Define an Integrated Catalog Facility User Catalog, Specifying SMS Keywords: Example 1 . . . . . 200
Define an Integrated Catalog Facility User Catalog, Taking All Defaults: Example 2 . . . . . 201
Define an Integrated Catalog Facility User Catalog, Using SMS Keywords and the VOLUME Parameter: Example 3 . . . . . 201
Define an Integrated Catalog Facility User Catalog, Using SMS Keywords and the VOLUME Parameter: Example 4 . . . . . 202
Define an Integrated Catalog Facility User Catalog: Example 5 . . . . . 203
Define a User Catalog Using the MODEL Parameter: Example 6 . . . . . 204
Define a General Tape Volume Catalog: Example 7 . . . . . 205
Define a Specific Tape Volume Catalog: Example 8 . . . . . 205

**Chapter 20. DELETE** . . . . . 207
DELETE Parameters . . . . . 207
Required Parameters . . . . . 207
Optional Parameters . . . . . 208
DELETE Examples . . . . . 217
Delete a Truename Entry in an Integrated Catalog Facility Catalog: Example 1 . . . . . 218
Delete an Integrated Catalog Facility User Catalog for Recovery: Example 2 218
Delete VSAM Volume Records: Example 3 . . . . . 219
Delete a Non-VSAM Data Sets Entry: Example 4 . . . . . 219
Delete Entries Associated with a Non-VSAM Object from VVDS and VTOC: Example 5 . . . . . 220
Delete a Key-Sequenced VSAM Cluster in a Catalog: Example 6 . . . . . 220
Delete Two Key-Sequenced Clusters in a Catalog: Example 7 . . . . . 221
Delete a User Catalog: Example 8 . . . . . 221
Delete an Alias Entry in a Catalog: Example 9 . . . . . 221
Delete Generically Named Entries in a Catalog: Example 10 . . . . . 222
List a Generation Data Group's Entries, Then Delete the Group and Its Data Sets in a Catalog: Example 11 . . . . . 222
Delete a Generation Data Group with Recovery: Example 12 . . . . . 223
Delete a Member of a Partitioned (Non-VSAM) Data Set in a Catalog: Example 13 . . . . . 224
Delete a Page Space: Example 14 . . . . . 224
Delete a VVDS with Recovery: Example 15 . . . . . 225
Delete an OAM Collection Name Catalog Entry: Example 16 . . . . . 225
Delete a Tape Library Entry: Example 17 . . . . . 226
Delete a Tape Volume Entry: Example 18 . . . . . 226

**Chapter 21. DIAGNOSE** . . . . . 227
DIAGNOSE Parameters . . . . . 227
Required Parameters . . . . . 227
Optional Parameters . . . . . 228
DIAGNOSE Examples . . . . . 230

Diagnose a VVDS: Compare the BCS and VVDS Example 1 . . . . . 230
Diagnose Only the BCS: Example 2 . . . . . 231
Diagnose the BCS: Compare the BCS and Certain VVDSs: Example 3 231
Diagnose a VVDS: Compare the BCS and VVDS: Example 4 . . . . . 232
Diagnose a VVDS: Compare the BCS and VVDS: Example 5 . . . . . 232
Diagnose a VVDS: Compare the BCS and VVDS: Example 6 . . . . . 233
Diagnose a VVDS: Compare the BCS and VVDS: Example 7 . . . . . 233

**Chapter 22. EXAMINE** . . . . . 235
EXAMINE Parameters. . . . . 235
Required Parameters . . . . . 235
Optional Parameters . . . . . 235
EXAMINE Examples . . . . . 236
Examine the Index Component of an Integrated Catalog Facility User Catalog: Example 1 . . . . . 236
Examine Both Components of a Key-Sequenced Data Set: Example 2 236
Examine the Data Component of an Integrated Catalog Facility User Catalog: Example 3 . . . . . 237

**Chapter 23. EXPORT** . . . . . 239
EXPORT Parameters . . . . . 239
Required Parameters . . . . . 239
Optional Parameters . . . . . 240
EXPORT Examples. . . . . 243
Export an Integrated Catalog Facility Catalog: Example 1. . . . . 243
Export a Key-Sequenced Cluster: Example 2 . . . . . 244
Export an Entry-Sequenced Cluster: Example 3 . . . . . 244
Export an Entry-Sequenced Cluster Using CIMODE: Example 4 . . . . . 245
Export Multiple Data Sets Using Multiple INFILE Parameters: Example 5 246

**Chapter 24. EXPORT DISCONNECT** . . . . . 247
EXPORT DISCONNECT Parameters . . . . . 247
Required Parameters . . . . . 247
EXPORT DISCONNECT Examples . . . . . 248
Export Disconnect of a User Catalog from Another User Catalog: Example 1 248
Export Disconnect of a User Catalog: Example 2 . . . . . 248

**Chapter 25. IMPORT** . . . . . 249
IMPORT Parameters . . . . . 249
Required Parameters . . . . . 249
Optional Parameters . . . . . 251
IMPORT Examples . . . . . 257
Import a Cluster Utilizing SMS Keywords: Example 1 . . . . . 257
Import an Integrated Catalog Facility Catalog: Example 2 . . . . . 258
Import a Key-Sequenced Cluster: Example 3 . . . . . 259
Import an Entry-Sequenced Cluster in a Catalog: Example 4 . . . . . 259
Import a Cluster to a Volume Other Than One on Which It Was Originally Defined: Example 5 . . . . . 260

**Chapter 26. IMPORT CONNECT** . . . . . 263
IMPORT CONNECT Parameters . . . . . 263
Required Parameters . . . . . 263
Optional Parameters . . . . . 263
Import Connect Example . . . . . 264
Import to Connect a User Catalog . . . . . 264

**Chapter 27. LISTCAT** . . . . . 267
LISTCAT Parameters . . . . . 267
Required Parameters . . . . . 267
Optional Parameters . . . . . 268
LISTCAT Examples . . . . . 273
List an SMS-Managed Data Set: Example 1. . . . . 273
List a Key-Sequenced Cluster's Entry in a Catalog: Example 2 . . . . . 273
Alter a Catalog Entry, Then List the Modified Entry: Example 3 . . . . . 274
List Catalog Entries: Example 4 . . . . . 275
List Catalog Entries: Example 5 . . . . . 275
List a Tape Library Entry: Example 6 . . . . . 275
List Tape Volume Entries: Example 7 . . . . . 276

**Chapter 28. PRINT** . . . . . 277
PRINT Parameters . . . . . 277
Required Parameters . . . . . 277
Optional Parameters . . . . . 277
PRINT Examples . . . . . 283
Print a Catalog: Example 1 . . . . . 283
Print a Key-Sequenced Cluster's Data Records: Example 2 . . . . . 283
Copy Records from a Non-VSAM Data Set into an Entry-Sequenced VSAM Cluster, Then Print the Records: Example 3 . . . . . 283
Print a Linear Data Set Cluster: Example 4 . . . . . 285
Print a Data Set that Contains DBCS Data: Example 5. . . . . 285

**Chapter 29. REPRO** . . . . . 287
REPRO Parameters . . . . . 288
Required Parameters . . . . . 288
Optional Parameters . . . . . 289
Cryptographic Parameters . . . . . 297
REPRO Examples . . . . . 300
Copy Records into a Key-Sequenced Data Set: Example 1 . . . . . 301
Copy Records into a VSAM Data Set: Example 2. . . . . 301
Merge an Integrated Catalog Facility User Catalog into Another Integrated Catalog Facility User Catalog: Example 3 . . . . . 302
Merge Selected Entries (Split) from a User Catalog into Another User Catalog: Example 4 . . . . . 303
Copy a Catalog: Example 5. . . . . 303
Copy a DBCS Data Set: Example 6. . . . . 305
Encipher Using System Keys: Example 7. . . . . 305
Decipher Using System Keys: Example 8. . . . . 306
Encipher Using Private Keys: Example 9 . . . . . 307
Decipher Using Private Keys: Example 10 . . . . . 308

**Chapter 30. SHCDS** . . . . . 311
SHCDS Parameters . . . . . 312
Required Parameters . . . . . 312
Optional Parameters . . . . . 312
SCHDS Examples . . . . . 317
Using PERMITNONRLSUPDATE With a Generic Data Set Name Specification: Example 1 . . . . . 317
Listing Data Sets With the High-Level Qualifier SYSPLEX: Example 2 317

**Chapter 31. VERIFY** . . . . . 319
VERIFY Parameters . . . . . 319
Required Parameter . . . . . 319

VERIFY Example . . . . . 319
Upgrade a Data Set's End-of-File Information . . . . . 319

**Appendix A. Security Authorization Levels** . . . . . 321
Required RACF Authorization Tables . . . . . 321

**Appendix B. Interpreting LISTCAT Output Listings** . . . . . 327
LISTCAT Output Keywords . . . . . 327
Alias Entry Keywords . . . . . 328
Alternate-Index Entry Keywords . . . . . 328
Cluster Entry Keywords . . . . . 328
Data Entry Keywords . . . . . 329
Index Entry Keywords . . . . . 331
Generation Data Group Base Entry Keywords . . . . . 333
Non-VSAM Entry Keywords . . . . . 333
Page Space Entry Keywords . . . . . 334
Path Entry Keywords . . . . . 334
User Catalog Entry Keywords . . . . . 335
Description of Keyword Fields . . . . . 335
ALC: Allocation Group . . . . . 335
ASN: Associations Group . . . . . 336
ATT: Attributes Group . . . . . 337
GDG: Generation Data Group Base Entry, Special Fields . . . . . 340
NVS: Non-VSAM Entry, Special Field . . . . . 340
HIS: History Group . . . . . 340
PRT: Protection Group. . . . . 342
STA: Statistics Group . . . . . 342
VLS: Volumes Group . . . . . 344
Device Type Translate Table . . . . . 346
Examples of LISTCAT Output Listings . . . . . 346
Job Control Language (JCL) for LISTCAT Jobs . . . . . 347
LISTCAT and Access Method Services Output Messages . . . . . 348
LISTCAT Output Listing . . . . . 349
LISTCAT NAMES Output Listing . . . . . 351
LISTCAT VOLUME Output Listing . . . . . 351
LISTCAT ALL Output Listing . . . . . 356
LISTCAT ALL Output Listing for a Non-VSAM Tailored Compressed Data Set . . . . . 367
LISTCAT ALLOCATION Output Listing . . . . . 368
LISTCAT HISTORY Output Listing . . . . . 374
LISTCAT LEVEL Output Listing . . . . . 378
LISTCAT ENTRIES Output Listing . . . . . 378
LISTCAT CREATION/EXPIRATION Output Listing . . . . . 378
Examples of LISTCAT in a TSO Environment . . . . . 381

**Appendix C. Interpreting SHCDS Output Listings** . . . . . 383
LISTDS . . . . . 383
LISTDS with Data Set in Retained Lock State . . . . . 383
LISTDS for Data Set Shared by Multiple Subsystems . . . . . 384
LISTDS for Data Set in Non-RLS Permitted State . . . . . 384
LISTDS with Data Set in Non-RLS Update and Permit First Time States 385
LISTDS for Data Set in Lost Lock State . . . . . 386
LISTSUBSYS . . . . . 386
LISTSUBSYS for All Subsystems Sharing Data Sets in the Sysplex . . . . 387
LISTSUBSYSDS . . . . . 388
LISTSUBSYSDS for Subsystem Sharing Multiple Data Sets . . . . . 388

LISTSUBSYSDS for All Subsystems in the Sysplex and the Shared Data Sets . . . 388
LISTRECOVERY . . . 390
LISTRECOVERY for Data Set Requiring Recovery . . . 391

**Appendix D. Invoking Access Method Services from Your Program** . . . 393
Authorized Program Facility (APF) . . . 393
Invoking Macro Instructions . . . 393
LINK or ATTACH Macro Instruction . . . 393
LOAD and CALL Macro Instructions. . . . 394
Invocation from a PL/I Program . . . 396
Processor Invocation . . . 398
Processor Condition Codes . . . 399
User I/O Routines . . . 399
VSAM Record-Level Sharing Considerations . . . 400

**Appendix E. IBM Storage Control Units—Caching Models** . . . 403

**Appendix F. DCOLLECT User Exit** . . . 405
User Exit Description . . . 405
User Exit Example . . . 406

**Appendix G. Interpreting DCOLLECT Output** . . . 411
DCOLLECT Output Record Structure . . . 413
DCOLLECT Output Record Field Descriptions . . . 444
Header Record Field . . . 444
Active Data Set Record Field . . . 445
VSAM Association Record Field . . . 449
Volume Record Field . . . 451
Data Class Construct Field . . . 452
Storage Class Construct Field . . . 456
Management Class Construct Field . . . 458
Base Configuration Field . . . 464
Aggregate Group Construct Field. . . . 466
Storage Group Construct Field . . . 467
Volume Definition Field . . . 471
Optical Drive Information Field. . . . 473
Library Information Field . . . 474
Migrated Data Set Record Field . . . 475
Backup Data Set Record Field. . . . 477
DASD Capacity Planning Record Field. . . . 479
Tape Capacity Planning Record Field . . . 481

**Appendix H. Accessibility** . . . 483
Using assistive technologies . . . 483
Keyboard navigation of the user interface. . . . 483

**Notices** . . . 485
Programming Interface Information . . . 486
Trademarks. . . . 486

**Glossary** . . . 487

**Index** . . . 499

# Figures


1. ALTER Attributes That Can be Altered and Types of Catalog Entry . . . . . 55
2. Example of Character Format. . . . . 278
3. Example of Dump Format . . . . . 278
4. Example of Hexadecimal Format . . . . . 279
5. Example of the Printed Record in DUMP Format . . . . . 283
6. Example of the Printed Record in Hexadecimal . . . . . 284
7. Example of a Printed Alphanumeric Character Record . . . . . 285
8. Messages That Follow the Entry Listing . . . . . 348
9. Example of LISTCAT Output When No Parameters Are Specified . . . . . 350
10. Example of LISTCAT NAME Output . . . . . 351
11. Example of LISTCAT VOLUME Output . . . . . 352
12. Example of LISTCAT ALL Output . . . . . 357
13. Example of LISTCAT ALL Output for a Non-VSAM Tailored Compressed Dataset. . . . . 368
14. Example of LISTCAT ALLOCATION Output. . . . . 369
15. Example of LISTCAT HISTORY Output . . . . . 375
16. Example of LISTCAT ENTRIES Output . . . . . 378
17. Example of LISTCAT CREATION(5) Output . . . . . 379
18. Example of LISTCAT EXPIRATION(365) Output . . . . . 380
19. LISTDS with Data Set in Retained Lock State. . . . . 384
20. LISTDS for Data Set Being Shared by Multiple Subsystems . . . . . 384
21. LISTDS for Data Set in NON-RLS PERMITTED State. . . . . 385
22. LISTDS for Data Set in Both NON-RLS UPDATE and PERMIT FIRST TIME States . . . . . 386
23. LISTDS for Data Set in Lost Lock State . . . . . 386
24. LISTSUBSYS for all Subsystems Sharing Data Sets in the Sysplex. . . . . 387
25. LISTSUBSYSDS for Subsystem Sharing Multiple Data Sets . . . . . 388
26. LISTSUBSYSDS for all Subsystems in the Sysplex and the Shared Data Sets. . . . . 390
27. LISTRECOVERY for Data Set Requiring Recovery . . . . . 391
28. Processor Invocation Argument List from Your Program . . . . . 397
29. Arguments Passed to and from a User I/O Routine. . . . . 401
30. DCOLLECT User Exit Example . . . . . 407


© Copyright IBM Corp. 1973, 2002

# Tables


1. Summary of AMS commands . . . . . 25
2. Allocate Command Parameters . . . . . 30
3. Data Class Attributes vs. Data Set Organization . . . . . 36
4. How NEWNAME Resolves When Change of Catalog is Required . . . . . 64
5. Required Security Authorization for Catalogs . . . . . 321
6. Required Security Authorization for VSAM Data Sets . . . . . 322
7. Required Security Authorization for Non-VSAM Data Sets . . . . . 323
8. Required Security Authorization for LISTCAT . . . . . 324
9. Required Security Authorization for Data Set Operations . . . . . 324
10. Required Security Authorization for VOLCAT Operations . . . . . 324
11. RACF Facility Class Authorization for IDCAMS Commands . . . . . 325
12. Required Authorization for SHCDS Subcommands . . . . . 326
13. Device Type Translate Table . . . . . 346
14. DCOLLECT Output Record Structure . . . . . 413
15. DCOLLECT Data Class Definition (Record Type ’DC’). . . . . 422
16. DCOLLECT Storage Class Definition (Record Type ’SC’) . . . . . 425
17. DCOLLECT Management Class Definition (Record Type ’MC’) . . . . . 426
18. DCOLLECT Storage Group Definition (Record Type ’SG’) . . . . . 430
19. DCOLLECT SMS Volume Information (Record Type ’VL’) . . . . . 431
20. DCOLLECT Base Configuration Information (Record Type ’BC’) . . . . . 433
21. DCOLLECT Aggregate Group Definition (Record Type ’AG’) . . . . . 434
22. DCOLLECT Optical Drive Information (Record Type ’DR’) . . . . . 435
23. DCOLLECT Optical Library Information (Record Type ’LB’) . . . . . 436
24. DCOLLECT Cache Names (Record Type ’CN’) . . . . . 437
25. DCOLLECT Accounting Information (Record Type ’AI’) . . . . . 438
26. DCOLLECT Output Listing: CONSTANTS . . . . . 438


© Copyright IBM Corp. 1973, 2002

# About This Book

This book is intended to help you use access method services commands. It contains reference information about the commands used to manipulate integrated catalog facility catalogs and the objects cataloged in them. It gives the syntax, a brief description, and examples of each access method services command used with integrated catalog facility catalogs and the objects cataloged in them.

Throughout this book, the term *catalog* refers to integrated catalog facility catalogs. Other types of catalogs are referred to explicitly. That is, virtual storage access method (VSAM) catalogs are referred to as VSAM catalogs.

For information on the use of commands related to integrated catalog facility catalog format and structure, see *z/OS DFSMS: Managing Catalogs*, SC26-7409.

For information on the use of commands related to VSAM data set format and structure, see *z/OS DFSMS: Using Data Sets*, SC26-7410.

In this book, the term “CICS,” refers to the CICS Transaction Server for OS/390, 5665-147. Citations are to the CICS Transaction Server for OS/390 library; CICS/MVS and CICS/ESA users should use the corresponding books in those libraries. See *CICS Transaction Server for OS/390: Planning for Installation*, GC34-5700, for more information.

## Required Product Knowledge

To use this book effectively, you should be familiar with:
- Catalog administration
- Job control language
- VSAM data management

## Referenced Publications

The following publications are referenced in this book:

| Publication Title | Order Number |
|---|---|
| *z/OS DFSMS Macro Instructions for Data Sets* | SC26-7408 |
| *z/OS DFSMS: Managing Catalogs* | SC26-7409 |
| *z/OS DFSMS: Using Data Sets* | SC26-7410 |
| *z/OS MVS JCL Reference* | SA22-7597 |
| *z/OS MVS JCL User’s Guide* | SA22-7598 |
| *z/OS MVS System Messages, Vol 1 (ABA-AOM)* | SA22-7631 |
| *z/OS MVS System Messages, Vol 2 (ARC-ASA)* | SA22-7632 |
| *z/OS MVS System Messages, Vol 3 (ASB-BPX)* | SA22-7633 |
| *z/OS MVS System Messages, Vol 4 (CBD-DMO)* | SA22-7634 |
| *z/OS MVS System Messages, Vol 5 (EDG-GFS)* | SA22-7635 |
| *z/OS MVS System Messages, Vol 6 (GOS-IEA)* | SA22-7636 |
| *z/OS MVS System Messages, Vol 7 (IEB-IEE)* | SA22-7637 |
| *z/OS MVS System Messages, Vol 8 (IEF-IGD)* | SA22-7638 |
| *z/OS MVS System Messages, Vol 9 (IGF-IWM)* | SA22-7639 |

© Copyright IBM Corp. 1973, 2002

| Publication Title | Order Number |
| --- | --- |
| *z/OS MVS System Messages, Vol 10 (IXC-IZP)* | SA22-7640 |

## Accessing z/OS DFSMS Books on the Internet

In addition to making softcopy books available on CD-ROM, IBM provides access to unlicensed z/OS softcopy books on the Internet. To find z/OS books on the Internet, first go to the z/OS home page: http://www.ibm.com/servers/eserver/zseries/zos

From this Web site, you can link directly to the z/OS softcopy books by selecting the Library icon. You can also link to IBM Direct to order hardcopy books.

## Using LookAt to look up message explanations

LookAt is an online facility that allows you to look up explanations for z/OS messages, system abends, and some codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can access LookAt from the Internet at:

`http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html`

or from anywhere in z/OS where you can access a TSO command line (for example, TSO prompt, ISPF, z/OS UNIX System Services running OMVS).

To find a message explanation on the Internet, go to the LookAt Web site and simply enter the message identifier (for example, `IAT1836` or `IAT*`). You can select a specific release to narrow your search. You can also download code from the *z/OS Collection*, SK3T-4269 and the LookAt Web site so you can access LookAt from a PalmPilot (Palm VIIx suggested).

To use LookAt as a TSO command, you must have LookAt installed on your host system. You can obtain the LookAt code for TSO from a disk on your *z/OS Collection*, SK3T-4269 or from the LookAt Web site. To obtain the code from the LookAt Web site, do the following:

1. Go to http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html.
2. Click the **News** button.
3. Scroll to **Download LookAt Code for TSO and VM.**
4. Click the ftp link, which will take you to a list of operating systems. Select the appropriate operating system. Then select the appropriate release.
5. Find the **lookat.me** file and follow its detailed instructions.

To find a message explanation from a TSO command line, simply enter: **lookat** *message-id*. LookAt will display the message explanation for the message requested.

**Note:** Some messages have information in more than one book. For example, IEC192I has routing and descriptor codes listed in *z/OS MVS Routing and Descriptor Codes*. For such messages, LookAt prompts you to choose which book to open.

## Accessing Licensed Books on the Web

z/OS licensed documentation in PDF format is available on the Internet at the IBM Resource Link Web site at:

`http://www.ibm.com/servers/resourcelink`

Licensed books are available only to customers with a z/OS license. Access to these books requires an IBM Resource Link Web userid and password, and a key code. With your z/OS order you received a memo that includes this key code.

To obtain your IBM Resource Link Web userid and password log on to:

`http://www.ibm.com/servers/resourcelink`

To register for access to the z/OS licensed books:

1. Log on to Resource Link using your Resource Link user ID and password.
2. Click on **User Profiles** located on the left-hand navigation bar.
3. Click on **Access Profile.**
4. Click on **Request Access to Licensed books.**
5. Supply your key code where requested and click on the **Submit** button.

If you supplied the correct key code you will receive confirmation that your request is being processed. After your request is processed, you will receive an e-mail confirmation.

**Note:** You cannot access the z/OS licensed books unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

To access the licensed books:

1. Log on to Resource Link using your Resource Link userid and password.
2. Click on **Library**.
3. Click on **zSeries**.
4. Click on **Software**.
5. Click on **z/OS**.
6. Access the licensed book by selecting the appropriate element.

## How to Send Your Comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other DFSMS documentation:

- Send your comments by e-mail to:
  - IBMLink™ from US: starpubs@us.ibm.com
  - IBMLink from Canada: STARPUBS at TORIBM
  - IBM Mail Exchange: USIB3VVD at IBMMAIL
  - Internet: starpubs@us.ibm.com

  Be sure to include the name of the book, the part number of the book, version and product name, and if applicable, the specific location of the text you are commenting on (for example, a page number or a table number).

- Fill out one of the forms at the back of this book and return it by mail or by giving it to an IBM representative. If the form has been removed, address your comments to IBM Corporation, RCF Processing Department M86/050, 5600 Cottle Road, San Jose, California 95193-0001, U.S.A.

## Notational Conventions

IBM uses a uniform notation to describe the syntax of access method services commands. This notation is not part of the language; it is a way of describing the syntax of the commands, and uses these conventions:

**[ ]** Brackets enclose an optional entry. You can, but need not, include the entry. Examples are:
- **[***length***]**
- **[MF=E]**

**|** An OR sign separates alternative entries. You must include one, and only one, of the entries unless you allow an indicated default. Examples are:
- **[REREAD|LEAVE]**
- **[***length***|'S']**

**{ }** Braces enclose alternative entries. You must use one, and only one, of the entries. Examples are:
- **BFTEK={S|A}**
- **{K|D}**
- **{***address***|S|O}**

Sometimes alternative entries are shown in a vertical stack of braces. An example is:

```
MACRF={{(R[C|P])}
       {(W[C|P|L])}
       {(R[C],W[C])}}
```

In the example above, you must choose only one entry from the vertical stack.

**...** An ellipsis indicates that the entry immediately preceding the ellipsis can be repeated. For example:
- **(***dcbaddr***,[(***options***)],. . .)**

**‘ ’** A blank indicates that a blank must be present before the next parameter.

**REQUIRED KEYWORDS AND SYMBOLS**

Entries shown **IN THE FORMAT SHOWN HERE** (notice the type of highlighting just used) must be coded exactly as shown. These entries have keywords and the following punctuation symbols: commas, parentheses, and equal signs. Examples are:
- **CLOSE , , , ,TYPE=T**
- **MACRF=(PL,PTC)**

The format (the type of highlighting) used to identify this type of entry depends on the display device used to view a softcopy book. The published version of this book displays this type of entry in uppercase boldface type.

**DEFAULT VALUES**

Values shown **IN THE FORMAT SHOWN HERE** (notice the type of highlighting just used) indicate the default used if you do not specify any of the alternatives. Examples are:

- **[EROPT={ACC|SKP|ABE}]**
- **[BFALN={F|D}]**

The format (the type of highlighting) used to identify this type of entry depends on the display device used to view a softcopy book. The published version of this book displays this type of entry in underscored uppercase boldface type.

*User Specified Value*

Values shown *in the format shown here* (notice the type of highlighting just used) indicate a value that you supply according to specifications and limits described for each parameter. Examples are:

- *number*
- *image-id*
- *count*

The format (the type of highlighting) used to identify this type of entry depends on the display device used to view a softcopy book. The published version of this book displays this type of entry in lowercase italic type.

## How to Code Access Method Services Commands

All access method services commands have the following general structure:

**COMMAND** *parameters* **...** [*terminator*]

The command defines the type of service requested. The parameters further describe the service requested. The terminator indicates the end of the command statement.

## Commands

Commands can begin at, or to the right of, the left margin. For batch processing jobs, the default margins are 2 and 72.

Commands are separated from their parameters by one or more separators (blanks, commas, or comments). For some parameters, parentheses are used as separators. Comments are strings of characters surrounded by /* and */. Comments can contain any characters except */.

The defaulted character set does not contain lower case. See “PARM Command” on page 19 for information on changing the character set used by IDCAMS. These parameters appear throughout this book, primarily in a table at the beginning of each command.

Many of the commands and keyword parameters can be abbreviated. Acceptable abbreviations appear after the description of each keyword parameter throughout “Chapter 3. Functional Command Syntax” on page 25. Keyword parameters in plural form can also be coded in singular form. Not all abbreviations acceptable under access method services are acceptable in TSO. Abbreviation restrictions in TSO are described in “From a Time Sharing Option Session” on page 5.

## Positional and Keyword Parameters

A parameter can either be a positional parameter or a keyword parameter. Positional parameters must always appear first in a parameter set. In access method services, positional parameters are never optional. For example, in:

```
DELETE -
     USERCAT -
```

USERCAT is a positional parameter that specifies the entry name to be deleted.

A keyword parameter is a specific character string that can have a value following it. For example, in:

```
VOLUME (25DATA)
```

VOLUME is a keyword that indicates that the value 25DATA is a volume serial number.

A keyword parameter can have a set of subparameters. Subparameters follow the same rules as parameter sets in general. When the subparameters are positional, the first subparameter is always required.

Positional parameters and subparameters sometimes have lists of items. Unless the list contains only one item, it must be enclosed in parentheses that can be preceded and followed by blanks, commas, or comments. For example:

**DELETE(***entryname* **[...])**

indicates that the list of entry names must be enclosed in parentheses if more than one entry is to be deleted. If only one entry name is given, the parentheses are not required.

An item in a list can be a parameter set itself. Each such item, as well as the list of items, is enclosed in parentheses. Given:

**OBJECTS((***entryname* **NEWNAME (***newname***))...)**

the following are valid:

```
OBJECTS -
  (ENTRY1 NEWNAME(NEWNAME1))
```

Here, only one entry is to be renamed. The entry name and its new name are enclosed in parentheses.

```
OBJECTS (-
  (ENTRY1 NEWNAME(NEWNAME1)) -
  (ENTRY2 NEWNAME(NEWNAME2)) -
        )
```

Here, each entry name and its new name are enclosed in parentheses and the entire list is enclosed in parentheses.

All parameters and subparameters must be separated from each other by one or more separators (commas, blanks, or comments). There is one exception: parameters do not need to be separated from the closing parenthesis when they immediately follow a subparameter set already enclosed in parentheses.

A value cannot have commas, semicolons, blanks, parentheses, or slashes unless the entire value is enclosed in single quotation marks. A single quotation mark in a field enclosed in single quotation marks must be coded as two single quotation marks.

The values you specify in the parameters can be surrounded by separators. Some values can be longer than a single record. When a value is longer than a single record, you indicate that it is continued by coding a plus sign followed only by

blanks or a comment. The first nonseparator character found in a record following the plus sign is treated as a continuation of the value.

## How to Code Subparameters

You can use decimal (n), hexadecimal (X'n'), or binary (B'n') form to define parameters.

These coding conventions apply to the subparameters in this section:

- When the subparameter contains a special character, enclose the subparameter in single quotation marks; for example, OWNER('*IBM*').
- When the subparameter contains a special character and a single quotation mark, code the embedded quotation mark as two single quotation marks; for example, VOLUMES('one' '&').
- When you code the subparameter in hexadecimal form, two hexadecimal characters represent one alphanumeric or special character. For example, FROMKEY(X'C1C2C3') is the same as FROMKEY(ABC). When you code a character string in hexadecimal, use an even number of hexadecimal characters because it will be justified to the right.
- When the subparameter contains a lowercase alphabetic character, it is changed to an uppercase alphabetic character.

The subparameters in this book are:

*aliasname*
: can contain 1 to 44 alphanumeric or national characters, and 2 special characters (the hyphen and the 12-0 overpunch (X'C0')).

  Names that have more than 8 characters must be segmented by periods; 1 to 8 characters can be specified between periods.

  The first character of any name or name segment must be either an alphabetic character or a national character.

*code*
: can contain 1 to 8 alphanumeric or special characters.

*entryname*
: can contain 1 to 44 alphanumeric or national characters, and 2 special characters (the hyphen and the 12-0 overpunch (X'C0')).

  Names that contain more than 8 characters must be segmented by periods; 1 to 8 characters can be specified between periods. A name segmented by periods is called a qualified name. Each name segment is referred to as a qualifier.

  The first character of any name or qualifier must be either an alphabetic character or a national character.

  Use an asterisk to replace a qualifier to indicate a generic command with certain commands. However, an asterisk cannot be used as the high level (leftmost) qualifier, as a partial replacement for a qualifier, or to replace more than one qualifier. The following examples show you how to use an asterisk for a generic name:

  ```
  A.*
  A.*.C
  ```

  The following examples are **not** acceptable ways to use a generic name:

```
A.*.*
A.B*
*.B.C
```

Refer to the *entryname* subparameter of each command to determine if a generic name is allowed and for more information on using one.

For a partitioned data set, the entry name must be given in the format: pdsname(membername). Blank characters are not allowed between the left and right parentheses enclosing the member name, or between the pdsname and the left parenthesis.

If you use an entry name in the format entry name(modifier), and the entry name is not the name of a partitioned data set, only that portion of the name preceding the left parenthesis is used. The modifier enclosed in parentheses is ignored.

*entrypoint*
can contain 1 to 8 alphanumeric or national characters, and 2 special characters (the hyphen and the 12-0 overpunch (X'C0')).

The first character of any name or name segment must be either an alphabetic character or a national character.

*newname*
can contain 1 to 44 alphanumeric or national characters, and 2 special characters (the hyphen and the 12-0 overpunch (X'C0')).

Names that contain more than 8 characters must be segmented by periods; 1 to 8 characters can be specified between periods. A name segmented by periods is called a qualified name. Each name segment is referred to as a qualifier.

The first character of any name or name segment must be either an alphabetic character or a national character.

Use an asterisk, to replace a qualifier to indicate a generic command with certain commands. Do not use an asterisk as the high level (leftmost) qualifier, a partial replacement for a qualifier, or to replace more than one qualifier. The following examples show how you can use an asterisk for a generic name:

```
A.*
A.*.C
```

The following examples are **not** acceptable ways to use a generic name:

```
A.*.*
A.B*
*.B.C
```

Refer to the *newname* subparameter of each command to determine if a generic name is allowed and for more information on using one.

*ownerid*
can contain 1 to 8 EBCDIC characters.

*pdsname(membername)*
is the name of a partitioned data set (PDS) or partitioned data set extended (PDSE) and a member within that data set. The membername can contain 1 to 8 alphanumeric or national characters, or a character X'C0' The first character must be alphabetic or national. Blank characters are not allowed between the left and right parentheses enclosing the member name, or between pdsname and the left parenthesis.

*string*
can contain 1 to 255 EBCDIC characters.

*volser*
a volume serial number have 1 to 6 alphanumeric, national, or special characters.

## Alphanumeric, National, and Special Characters

The following is a list of alphanumeric, national, and special characters used in this book:

- Alphanumeric characters:
  alphabetic characters A through Z
  numeric characters 0 through 9
- National characters
  at sign @
  dollar sign $
  pound sign #
- Special characters
  ampersand &
  asterisk *
  blank
  braces { }
  brackets [ ]
  comma ,
  equal sign =
  hyphen -
  parenthesis ( )
  period .
  plus sign +
  semicolon ;
  single quotation mark '
  slash /

## How to Continue Commands and Parameters

Commands can be continued on several records or lines. Except for the last line, each record or line must have a hyphen or a plus sign as the last nonblank character before, or at, the right margin.

A hyphen continues the command. A plus sign continues both the command and a value within the command.

Examples of these two types of continuation are:

```
DELETE -
  (ENTRY1 -
  ENTRY2 -
  ENTR+
    Y3) -
  NONVSAM
```

A blank record, or a record ending with a complete comment, must end with a continuation mark when it appears in the middle of a command, and when it precedes or follows the THEN and ELSE clauses of an IF command.

```
IF LASTCC = 0 -
  THEN -
  REPRO ...
  /*COMMENT WITH NO CONTINUATION MARK AFTER*/
  ELSE -
  PRINT ...
```

Because no continuation mark (hyphen) follows the comments, a null command is assumed. The ELSE keyword will not match the THEN keyword.

Records ending with partial comments must always end with a continuation mark. Only blank characters can appear between a continuation mark and the end of the record.

**Exception:** The DO-END sequence does not require continuation characters. If you use continuation characters, they can be read as a null command or cause unpredictable results.

## The Terminator

The terminator ends the command, and can be either a semicolon or the absence of a continuation mark.

If you use the semicolon as the terminator, do not close it in quotation marks or embed it in a comment. Everything to the right of the semicolon is ignored.

For example, if you code:

```
PARM TEST (TRACE); PARM -
GRAPHICS (CHAIN(TN))/*COMMENT*/ -
PRINT ...
REPRO ...
```

the characters following the semicolon terminator are ignored. Because a continuation mark (hyphen) appears at the end of the second record, the PRINT command is also ignored. The first PARM command and the REPRO command are the only recognized commands.

# Summary of Changes

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

## Summary of Changes for SC26-7394-01 z/OS Version 1 Release 3

This book contains information previously presented in *z/OS Version 1 Release 1 DFSMS Access Method Services for Catalogs* (SC26-7394-00).

The following sections summarize the changes to that information.

### New Information

This edition includes the following new information:

- A new parameter, FRLOG, is supported for the ALTER and DEFINE CLUSTER commands. This parameter allows you to enable CICSVR batch logging using either of these commands.

### Deleted Information

The following parameters are no longer supported and information about the parameters has been deleted from this edition:

- IMBED
- KEYRANGE
- ORDERED
- REPLICATE

If you use these parameters, no error message is displayed and no processing occurs.

### Changed Information

The following information was changed in this edition:

- For the GDG BASE information on the LISTCAT output, the EXPIRATION field has been changed to the LAST ALTER DATE which indicates the last time the GDG data was changed.

## Library Changes for z/OS Version 1 Release 3

You might notice changes in the style and structure of some content in this book—for example, headings that are more task-oriented, notes with headings that are specific and clear in their intent, additional index entries for easier information retrieval, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our books.

© Copyright IBM Corp. 1973, 2002

# Chapter 1. Using Access Method Services

Access method services is a utility you can use to establish and maintain catalogs and data sets. The Storage Management Subsystem (SMS) and its classes, in conjunction with the automatic class selection (ACS) routines, automate many access method services commands and their specified parameters. However, if your storage administrator has not established these routines, you can construct and execute them manually. You can find these commands and parameters in this book.

There are two types of access method services commands: *Functional commands*, which you use to define data sets or list catalogs; and *modal commands*, which you use to specify the conditional execution of functional commands. If you are a time sharing option (TSO) user, you are restricted to using only the functional commands. For detailed information about these commands, their parameters and subparameters, refer to the subsequent chapters in this book.

Use this book as a reference only. It is not a complete source of information about access method services. For more information about access method services tasks, refer to *z/OS DFSMS: Managing Catalogs* and *z/OS DFSMS: Using Data Sets*. To find out more about SMS, its keywords, the ACS routines and requirements, see *z/OS DFSMS Introduction*.

This chapter covers the following topics:


| For Information About | See Page |
|---|---|
| Notational Conventions | xx |
| How to Code Access Method Services Commands | xxi |
| Identifying Data Sets and Volumes | 1 |
| Invoking Access Method Services | 4 |
| Access Method Services Tape Library Support | 7 |
| Order of Catalog Use | 9 |
| Specifying Attribute Selection Order | 14 |


## Identifying Data Sets and Volumes

When you use access method services commands, you must identify data sets and volumes. Data sets must be identified when they are accessed. Volumes must be identified when VSAM accesses the volume table of contents (VTOC), allocates or releases space using OS/VS DADSM functions, or accesses a VSAM volume data set.

VSAM data sets or volumes can be identified through the ALLOCATE command, through job control language (JCL), or by the data set name or volume serial number within the command that requires the data set or volume for its execution. If you do not use JCL or the ALLOCATE command, an attempt is made to dynamically allocate the data set or volume as required.

Under the SMS, you should not explicitly identify volumes. The system identifies the necessary volumes when a storage class is assigned to the data set. You can allocate your data set to a *specific* volume only if your storage administrator has set

© Copyright IBM Corp. 1973, 2002

GUARANTEED SPACE=YES in the storage class assigned to your data set. See *z/OS DFSMSdfp Storage Administration Reference* for further information about SMS volume selection.

## Dynamic Allocation

To dynamically allocate either a VSAM data set or a non-VSAM data set, the data set name must exist and be cataloged. The catalog containing the entry must be one of the following:

- A user catalog identified with a JOBCAT or STEPCAT DD statement. However STEPCAT DD statements are not supported for ICF catalogs that have a UCB above the 16MB line.
- An integrated catalog facility catalog whose name or alias matches one or more of the qualifiers of the qualified data set name

All referenced integrated catalog facility catalogs must be connected to the system master catalog.

Access method services dynamically allocates VSAM and non-VSAM data sets with a disposition of OLD.

To dynamically allocate a volume, the volume must already be mounted as permanently resident or reserved. You should carefully consider the PRIVATE and PUBLIC use attributes when you mount a volume.

### Security Authorization

Passwords are no longer honored for protecting an ICF catalog, or data sets contained in an ICF catalog. If they are specified, they will be ignored and no message will be issued. Previously, passwords were ignored only for SMS-managed data sets. You should use RACF®, a component of the Security Server for z/OS, or an equivalent security package, to protect your data. Most instances of passwords have been deleted from this book. Passwords in keywords such as ATTEMPTS, AUTHORIZATION, CODE, and LOCK will be ignored.

For information on RACF authorization levels, see “Appendix A. Security Authorization Levels” on page 321. RACF applies to both SMS-managed and non-SMS-managed data sets and catalogs.

If an ICF catalog is shared with a downlevel system, data sets will remain password-protected in the downlevel system, but not in a DFSMS/MVS® Version 1 Release 5 system or z/OS V1R1 system.

If you are transferring data from a system with RACF to a system which does not have RACF, data sets in an ICF catalog will not be protected.

### Storage Management Subsystem (SMS) Considerations

You should not direct a SMS-managed data set to a specific catalog. Allow the system to determine the catalog through the usual catalog search order. Naming a catalog for a SMS-managed data set requires authority from the RACF STGADMIN.IGG.DIRCAT facility class profile. For information about specifying catalogs with an SMS-managed data set, see *z/OS DFSMS: Managing Catalogs*.

JOBCAT and STEPCAT DD statements are not allowed for SMS-managed data sets and catalogs except when using the LISTCAT command. If a job contains a JOBCAT or STEPCAT DD statement, the first step in the job referencing the SMS-managed data set or catalog, and all subsequent steps, will be unsuccessful and the job will end.

## JCL DD Statements

When you use a JCL DD statement to identify a data set, include this on the DD statement:

- User data set name
- Catalog name of the BCS
- VVDS name
- Unit and volume serial numbers, if the data set is not cataloged
- Disposition
- AMP='AMORG' is required for VVDSs

### JCL DD Statement for a VSAM Data Set

You can allocate VSAM data sets directly with the access method services ALLOCATE command. The following DD statements demonstrate two additional methods of describing and allocating a VSAM data set:

- For allocating and creating a new data set:

```
//ddname DD DSNAME=dsname,DISP=(NEW,CATLG),RECORG=KS,
//          SPACE=(TRK,10,10),STORCLAS=xxxxx
```

- For allocating an existing data set:

```
//ddname DD DSNAME=dsname,DISP=OLD
```

Access method services does not provide protection for data sets in a shared environment. Therefore, you should use DISP=OLD on the DD statement for any data set that can be accessed improperly in a shared environment.

### JCL DD Statement for a Volume

To identify and allocate a volume, include:

- Volume serial number
- Disposition
- Unit

This DD statement identifies and allocates volume VSER01:

```
//VOLDD    DD   VOL=SER=VSER01,UNIT=3380,DISP=OLD
```

For information on concatenated DD statements, see the FILE parameter descriptions in Chapter 3. Functional Command Syntax. Examples using concatenated DD statements follow the description of the REPRO command that begins on “Chapter 29. REPRO” on page 287. For additional information about the various types of concatenated DD statements, see the section on special DD statements in *z/OS MVS JCL Reference*.

### JCL DD Statement for a Non-VSAM Data Set

You can allocate non-VSAM data sets with the access method services ALLOCATE command. See the DD statements in the examples that follow the descriptions of the BLDINDEX, EXPORT, IMPORT, REPRO, and PRINT commands for additional methods of describing and allocating non-VSAM data sets.

### JCL DD Statement for a Snap Dump

If access method services encounters a condition that requires it to abnormally end a job, it takes a snap dump of virtual storage. You must write an AMSDUMP DD statement to get the snap dump; that is,

```
//AMSDUMP  DD   SYSOUT=A
```

If you do not supply an AMSDUMP DD statement and access method services encounters a condition requiring the job to abnormally end, it produces only an abbreviated dump.

### JCL DD Statement for a Target Data Set

The usual target data set for listing is SYSPRINT. The default parameters of this data set are:

- Record format: variable blocked (VBA)
- Logical record length: 125, that is, (121+4)
- Block size: 0

Print lines are 121 bytes long. The first byte is the ANSI (American National Standards Institute) control character. The minimum LRECL is 121 (U-format records only). If a smaller size is used, it is overridden to 121.

You can alter the defaults by placing other values in the DCB parameter of the SYSPRINT statement. You cannot, however, use a record format of F or fixed block (FB); those are changed to VBA.

### JCL DD Statement for an Alternate Target Data Set

In several commands you can use an alternate target data set for listing, but do not use F or FB record formats.

JCL statements, system messages, and job statistics are written to the SYSPRINT output device, not to the alternate target data set.

## Direct Allocation Using JCL

You can directly allocate VSAM data sets through JCL.

The following example allocates a new data set and, with DATACLAS, uses the allocation attributes predetermined by the storage administrator through the ACS routines.

```
//DD1     DD DSNAME=EXAMPLE1,DATACLAS=DCLAS01,
//        DISP=(NEW,KEEP)
```

See *z/OS MVS JCL User's Guide* and *z/OS MVS JCL Reference* for information about JCL keywords.

## Invoking Access Method Services

When you want to use an access method services function, enter a command and specify its parameters. Your request is decoded one command at a time; the appropriate functional routines perform all services required by that command.

You can call the access method services program:

- As a job or jobstep
- From a TSO session
- From within your own program

You can run the IDCAMS program (the access method services operating system) and include the command and its parameters as input to the program. You can also call the IDCAMS program from within another program and pass the command and its parameters to the IDCAMS program.

Time sharing option (TSO) users can run access method services functional commands from a TSO session as though they were TSO commands.

See "Appendix D. Invoking Access Method Services from Your Program" on page 393, for more information.

## As a Job or Jobstep

You can use (JCL) statements to call access method services. PGM=IDCAMS identifies the access method services program.

For example:

```
//YOURJOB  JOB  YOUR INSTALLATION'S JOB=ACCOUNTING DATA
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *

 access method services commands and their parameters

/*
```

- //YOURJOB, the JOB statement, is required to describe your job to the system. You might be required to supply user identification, accounting, and authorization information with the JOB statement's parameters.
- //STEP1, the EXEC statement, is required. With PGM=IDCAMS, this statement calls access method services to decode and process the access method services commands and parameters contained in the SYSIN data set. You can use the PARM operand of the EXEC statement to pass parameters to the access method services program. "Chapter 2. Modal Commands" on page 15, describes the PARM command and explains the options you can use.
- //SYSPRINT, the SYSPRINT DD statement, is required. It identifies the output device to which access method services sends messages and output information.
- //SYSIN, the SYSIN DD statement, is required to identify the source of the input statements. An input statement is a functional or modal command and its parameters. When you code SYSIN DD *, you identify the following statements as input.

The last input statement can be followed by a delimiter statement that has an * in the first two columns.

## From a Time Sharing Option Session

You can use the time sharing option (TSO) with VSAM and access method services to:

- Run access method services functional commands
- Run a program to call access method services

Each time you enter an access method services command as a TSO command, TSO builds the appropriate interface information and calls access method services.

You can enter one command at a time. Access method services processes the command completely before TSO lets you continue processing. Except for ALLOCATE, all the access method services functional commands are supported in a TSO environment.

To use IDCAMS and some of its parameters from TSO, you must update the IKJTSO*xx* member of SYS1.PARMLIB. Add IDCAMS to the list of authorized programs (AUTHPGM). If you want to use SHCDS, SETCACHE, LISTDATA, DEFINE, and IMPORT from TSO, you must add them (and abbreviations) to the authorized command list (AUTHCMD). If you use the CSECT IKJEGSCU instead of IKJTSO*xx,* make the updates to the CSECT. Please see *z/OS TSO/E Customization* for more information.

The restricted functions performed by access method services that cannot be requested in an unauthorized state are:

- DEFINE—when the RECATALOG parameter is specified
- DEFINE—when the define is for an alias of a UCAT
- DELETE—when the RECOVERY parameter is specified
- EXPORT—when the object to be exported is a BCS
- IMPORT—when the object to be imported is a BCS
- PRINT—when the object to be printed is an integrated catalog facility catalog
- LISTDATA—all functions
- REPRO—when a BCS is copied or merged
- SETCACHE—all functions
- SHCDS—all functions
- VERIFY—when a BCS is to be verified.

When you use TSO with access method services, note that:

- You can use the first characters of a keyword as an abbreviation of the keyword. You must use enough initial characters to make the keyword unique. TRACKS, for example, can be abbreviated TR, TRA, or TRAC, because no other keyword within the same command can be abbreviated in the same way.

  You cannot use some abbreviations (such as CYL, CYLINDER, REC, RECORD) under TSO because the abbreviations do not have enough initial characters to make the keyword unique. For example, TSO cannot tell whether you mean CYLINDERS or CYLINDERFAULT if you use CYL or CYLINDER.
- When a parameter's value is one or more parenthesized parameter sets, the outer parentheses surrounding the list are always required. For example, if `lowkey` and `highkey` form a parameter set that can be repeated several times, then the outer parentheses are required even when just one parameter set is specified; as follows:

  **KEYWORD((***lowkey highkey***))**
- In TSO, a volume serial number can contain alphanumeric characters, national characters ($, @, or #), and hyphens (-). alphabetic, national, numeric, or hyphen only; if any other characters are used, the volume serial number cannot be used as an entry name.
- A data set name can be placed within quotation marks or not. In TSO however, you add a prefix (for example, the user ID) to a name not in quotation marks. The prefix becomes the first qualifier in the name.
- In TSO, you add the necessary qualifiers to the specified name. However, you can be prompted to complete a fully qualified name. You are also prompted to supply any required parameters you have omitted.
- The modal commands, IF-THEN-ELSE, DO-END, SET, and PARM, are not allowed in TSO.
- In TSO, the SYSPRINT data set is not used. The OUTFILE parameter can be used with certain commands to specify a data set to receive access method services output.

Other TSO restrictions are noted in "Chapter 3. Functional Command Syntax" on page 25.

For details about the format of a displayed catalog entry (resulting from a LISTCAT command) for a TSO user, see "Appendix B. Interpreting LISTCAT Output Listings" on page 327.

For details about writing and executing programs with TSO, see *z/OS TSO/E User's Guide* and *z/OS TSO/E Command Reference*.

## Access Method Services Tape Library Support

Access method services provides support for tape library functions. The access method services ALTER, CREATE, and DELETE commands, however, should be used only to recover from tape volume catalog errors. Because access method services cannot change the library manager inventory in an automated tape library, ISMF should be used for usual tape library ALTER, CREATE, and DELETE functions.

## Summary of Tape Library Support

Access method services supports the following tape library functions:

- Creating, altering, deleting, copying, and listing catalog entries for tape library and tape volume entries
- Merging tape volume entries into other volume catalogs
- Providing support for functions that maintain an up-to-date tape library inventory.

The CATALOG parameter is ignored when specified on any command that affects a tape library entry except for the LISTCAT command.

A tape library entry is the record for a tape library. A tape volume entry is the record for a cartridge tape in a tape library.

## Access Method Services Commands for Tape Library Support

Use the following access method service commands to interact with a tape library:

### ALTER LIBRARYENTRY

Lets you alter all tape library entry fields except the library name.

### ALTER VOLUMEENTRY

Lets you alter all tape volume entry fields except for the tape volser.

### CREATE LIBRARYENTRY

Lets you create a tape library entry.

### CREATE VOLUMEENTRY

Lets you create a tape volume entry.

### DEFINE USERCATALOG

Lets you specify the VOLCATALOG parameter to define a volume catalog. A volume catalog is an integrated catalog facility catalog that contains only tape library and tape volume entries.

### DELETE

Use this command to delete tape library and tape volume entries.

- Specify the LIBRARYENTRY parameter to delete a tape library entry.
- Specify the VOLUMEENTRY parameter to delete a tape volume entry.
- The NOSCRATCH/SCRATCH parameter does not apply to tape library or tape volume entries, because the entries have no VVDS or VTOC entries.
- You can use the PURGE parameter to delete tape volume entries, regardless of expiration dates.
- If you use the FORCE parameter in LIBRARYENTRY, the tape library entry will be deleted. Any tape volume entries associated with the deleted tape library entry remain in the volume catalog. If you do not use FORCE, the tape library entry will be deleted only if it has no associated tape volume entries.

### DIAGNOSE

Identifies tape library and tape volume record types. DIAGNOSE checks the cell structure of the volume catalog.

### EXPORT/IMPORT

Imports and exports volume catalogs.

### LISTCAT

Displays fields that are associated with tape library and tape volume entries.

- Use LIBRARYENTRIES to list tape library entries.
- Choose VOLUMEENTRIES to list tape volume entries.
- Use the CATALOG parameter to retrieve tape volume entries from a specified volume catalog.
- To group tape library and tape volume entries, use the ALL parameter. The HISTORY, VOLUME, and ALLOCATION parameters are not valid and will be ignored.

### REPRO MERGECAT

Merges entries from one volume catalog to another. REPRO retrieves tape library or tape volume entries and redefines them in a target volume catalog. You cannot use the LEVEL parameter when merging volume catalogs.

If the character prior to the last character in both VOLCATs is 'V', both VOLCATs are specific. You cannot use MERGECAT of two specifics as this would mix VOLSER names.

For a target VOLCAT that is SPECIFIC (not VGENERAL), you must specify the VOLUMEENTRIES parameter. You do not want all entries merged into a specific catalog.

If the case is VGENERAL to SPECIFIC, the specified entry characters must match the first two third qualifier characters of the target catalog.

You may not use the REPRO command to copy the catalogs of two VGENERALs, whether MERGECAT or NONMERGECAT is specified.

### REPRO NOMERGECAT

Copies volume catalogs. When you copy a volume catalog to another volume catalog, REPRO verifies that the target is a volume catalog.

For additional information on tape volume catalogs, VOLUMEENTRIES, and LIBRARYENTRIES, refer to *z/OS DFSMS OAM Planning, Installation, and Storage Administration Guide for Tape Libraries*. Also see *z/OS DFSMS: Managing Catalogs*.

The VOLUMEENTRIES parameter cannot be specified on a NONMERGECAT. It always copies all entries in the volume catalog and cannot be restricted to a subset.

If NOMERGECAT or the default of NOMERGECAT is specified, the first qualifiers of the VOLCATs must not match while their third qualifiers must match. This allows users to make a copy of a VOLCAT, but ensures that the copy is of the same type. You do not want VA entries to be copied into the VB catalog, for example.

### Tape Library Naming Conventions

***Tape LibraryTape Library Names:*** The 1-to-8 character names of tape library entries can include only alphanumerics and the national characters $, @, or #. The first character of the name must be non-numeric and must not be the letter 'V'.

***Tape Volume Names:*** Tape volume names have a 'V' concatenated with a 1-to-6 character tape volser. The tape volser can include only uppercase alphabetics A–Z and numerics 0–9.

### Tape Library Date Formats

For all tape library parameters that require date entry, the date is in the form *YYYY-MM-DD,* where:

*YYYY* is 0000–2155
*MM* is 01–12
*DD* is 01–28, 29, 30, or 31

Specify a day of 32 for the 'never expire' date of 1999-12-32.

Enter all dates with leading zeros when appropriate.

## Order of Catalog Use

To select the catalog to be searched or chosen for an entry, you can use either the CATALOG parameter or a JOBCAT or STEPCAT DD statement. If used with an alias name, the catalog associated with the name is searched or selected.

JOBCAT and STEPCAT DD statements are not allowed for SMS-managed data sets. If a job contains a JOBCAT or STEPCAT DD statement, the first step in the job referencing the SMS-managed data set and all subsequent steps are unsuccessful and the job ends.

The multilevel alias facility enhances catalog selection that is based on high-level qualifiers of the data set name. It employs a right to left search of the multiple levels of qualifiers of the data set name for a matching alias name or user catalog name. The alias name or user catalog name with the greatest number of matching qualifiers is selected. For additional information on the multilevel alias facility, see *z/OS DFSMS: Managing Catalogs*.

Throughout the following "catalog search" and "catalog selection" sections, the catalog name cannot be specified for SMS-managed data sets unless you have authority from the RACF STGADMIN.IGG.DIRCAT facility class. With this authorization, a data set can be directed to a specific catalog. For more information about this facility class, see *z/OS DFSMSdfp Storage Administration Reference*.

## Catalog Search Order for ALTER

1. If a catalog is given in the CATALOG parameter, only that catalog is searched. If the entry is not found, a no-entry-found error is returned.
2. Any user catalog specified in the current job step with a STEPCAT DD statement is searched. If more than one catalog is given for the job step, the catalogs are searched in order of concatenation. If the entry is found, no other catalog is searched.

   If a STEPCAT catalog is given and the entry is not found, the JOBCAT catalog is not searched. The catalog search continues with step 3.

   If no STEPCAT catalog is given for the job step, and a user catalog is specified for the current job with a JOBCAT DD statement, the JOBCAT catalog is

searched. If more than one catalog is listed for the job, the catalogs are searched in order of concatenation. If the entry is found, no other catalog is searched.

3. If the entry is identified with a qualified entry name, and it is not generic, and its qualifiers are the same as the name or the alias of an integrated catalog facility catalog, or if the entry is found, no other catalog is searched.
4. The master catalog is searched. If the entry is not found in any of the indicated catalogs, a no-entry-found error is returned.

## Catalog Selection Order for BLDINDEX

This section applies only to users of BLDINDEX who code NOSORTCALL.

1. If a catalog is specified with the CATALOG parameter, that catalog is selected to contain work file entries.
2. The user catalog in the current job step (STEPCAT) or, if none is given, the user catalog specified for the current job (JOBCAT) is selected to contain the work file entries. If more than one catalog is listed for the job step or job, the first STEPCAT or JOBCAT catalog is selected to contain the work file entries.
3. If the entry (data set name on the DD statement) is identified with a qualified entry name, and:
   - One or more of its qualifiers is the same as the name or the alias of an integrated catalog facility catalog, or
   - The first qualifier is the same as the name or the alias of a VSAM user catalog

   then the user catalog so identified is selected to contain the work file entries.
4. The master catalog is selected to contain the work file entries.

## Catalog Selection Order for DEFINE

1. If a catalog is defined in the CATALOG parameter, that catalog is selected to contain the to-be-defined entry.
2. When a non-VSAM generation data group (GDG) data set is defined, the catalog containing the GDG base is selected to contain the to-be-defined non-VSAM entry.
3. The first user catalog listed in the current job step (STEPCAT) or, if none is specified for the job step, the first user catalog in the current job (JOBCAT) is selected to contain the to-be-defined entry.
4. If no user catalog is specified for the current job step or job, the entry's name is a qualified name, and:
   - One or more of its qualifiers is the same as the name or the alias of an integrated catalog facility catalog, or
   - The first qualifier is the same as the name or the alias of a VSAM user catalog,

   then the catalog so identified is selected to contain the to-be-defined entry.
5. If no catalog has been identified, either explicitly or implicitly, VSAM defines an object in the master catalog.

## Catalog Search Order for DELETE

If this is not a generic delete, the order in which catalogs are searched to locate an entry to be deleted is:

1. If a catalog is given in the CATALOG parameter, only that catalog is searched. If the entry is not found, a no-entry-found error is returned.

2. Any user catalog in the current job step (with a STEPCAT DD statement) is searched. If more than one catalog is specified for the job step, the catalogs are searched in order of concatenation. If the entry is found, no other catalog is searched.

   If a STEPCAT catalog is specified and the entry is not found, the JOBCAT catalog is not searched. The catalog search continues with step 3.

   If no STEPCAT catalog is specified for the job step, and a user catalog is given for the current job (with a JOBCAT DD statement), the JOBCAT catalog is searched. If more than one catalog is listed for the job, the catalogs are searched in order of concatenation. If the entry is found, no other catalog is searched.
3. If the entry is identified with a qualified entry name, and:
   - One or more of its qualifiers is the same as the name or the alias of an integrated catalog facility catalog,

   If the entry is found, no other catalog is searched.
4. If the entry is not found, the master catalog is searched. If the entry is not found in the master catalog, a no-entry-found error is returned.

If this is a generic delete, the order in which catalogs are searched to locate all applicable entries to be deleted is:

1. If a catalog is given in the CATALOG parameter, only that catalog is searched. If an entry that matches the supplied qualifiers is not found, a no-entry-found error is returned.
2. Any user catalog in the current job step (with a STEPCAT DD statement) is searched. If more than one catalog is in the job step, the catalogs are searched in order of concatenation. The JOBCAT catalog is not searched. The catalog search continues with step 3.

   If no STEPCAT catalog is given for the job step, and a user catalog is specified for the current job (with a JOBCAT DD statement), the JOBCAT catalog is searched. If more than one catalog is given for the job, the catalogs are searched in order of concatenation. The catalog search continues with step 3.
3. If the entry is identified with a qualified entry name, and:
   - One or more of its qualifiers is the same as the name of an integrated catalog facility catalog, or
   - One or more of its qualifiers is the same as the alias of an integrated catalog facility catalog, or
   - The first qualifier is the same as the name of a VSAM user catalog, or

   the catalog search continues with step 4.
4. The master catalog is searched.
5. If an entry matching the supplied qualifiers is not found in any of the catalogs searched, a no-entry-found error is returned.

### Caution

Unwanted deletions can take place if the catalog is not specified with the CATALOG parameter. Other catalogs are searched, according to the order previously described, and any entries matching the supplied qualifiers are deleted.

For information about generic catalog selection for the DELETE command, see “Generic Catalog Selection for DELETE and LISTCAT” on page 13.

## Catalog Selection Order for EXPORT DISCONNECT

1. If a catalog is specified with the CATALOG subparameter, that catalog is selected. If the data set is not found in that catalog, the command will be unsuccessful.
2. If a JOBCAT or STEPCAT is present, that catalog is selected.
3. If the entry is identified with a qualified entry name, and:
   - One or more of its qualifiers is the same as the name or the alias of an integrated catalog facility catalog, or
   - The first qualifier is the same as the name or the alias of a VSAM user catalog
   - the user catalog so identified is searched.

   The catalog search continues with step 4.
4. Then the master catalog is searched. If the entry is not found in the master catalog, a no-entry-found error is returned.

## Catalog Search Order for LISTCAT

When you do not use the ENTRIES parameter, or the command is not run through TSO and it is not a generic LISTCAT, the order in which catalogs are searched when entries are to be listed using the LISTCAT command is:

1. If a catalog is specified in the CATALOG parameter, only that catalog is listed.
2. The first user catalog in the current job step (STEPCAT) or, if none is given, the first user catalog specified in the current job (JOBCAT) is listed.
3. If no user catalog is named in the current job step or job, the master catalog is listed.

If the command is not a generic LISTCAT and the ENTRIES or LEVEL parameter is used, or when the command is run through TSO, the order in which catalogs are searched when entries are to be listed using the LISTCAT command is:

1. If a catalog is in the CATALOG parameter, only that catalog is searched. If the entry is not found, a no-entry-found error is returned.
2. Any user catalog in the current job step (STEPCAT) is searched, or, if none is given for the job step, any user catalog specified for the current job (JOBCAT) is searched. If more than one catalog is given for the job step or job, the job step or job catalogs are searched in order of concatenation. If the entry is found, no other catalog is searched.
3. If the entry is not found, the entry's name is a qualified name, and:
   - One or more of its qualifiers is the same as the name or the alias of an integrated catalog facility catalog

   that user catalog is searched. If the entry is found, no other catalog is searched.
4. The master catalog is searched. If the entry is not found, a no-entry-found error is returned.

When the ENTRIES parameter is used and this is a generic LISTCAT, the order in which catalogs are searched when entries are to be listed using the LISTCAT command is:

1. If a catalog is shown in the CATALOG parameter, only that catalog is searched. If an entry is not found that matches the supplied qualifiers, a no-entry-found error is returned.
2. Any user catalog in the current job step (STEPCAT) is searched, or, if none is given for the job step, any user catalog given for the current job (JOBCAT) is

searched. If more than one catalog is specified for the job step or job, the job step or job catalogs are searched in order of concatenation. The catalog search continues with step 3.

3. If the entry's name is a qualified name, and:
   - One or more of its qualifiers is the same as the name or the alias of an integrated catalog facility catalog is searched. The catalog search continues with step 4.
4. The master catalog is searched. If an entry has not been found in any of the catalogs searched that matched the supplied qualifiers, a no-entry-found error is returned.

## Generic Catalog Selection for DELETE and LISTCAT

The multilevel alias facility enhances generic catalog selection. If you use generic catalog selection with multilevel aliases, you can select several catalogs if the number of qualification levels of the generic name is less than the maximum your system allows. See *z/OS DFSMS: Managing Catalogs* for information about setting multilevel alias levels in the catalog address space. If the number of qualification levels in the data set name is less than the maximum your system allows, and aliases exist that match the generic data set name, then every catalog related to those aliases (including the master catalog) is selected.

The multilevel alias facility and the system-generated name format require special attention, such as:

- During the DEFINE of a VSAM data set, if the specified data/index name does not point to the same catalog as the cluster, an error occurs.
- During the DEFINE of a VSAM cluster or a GDG, if the name of the cluster or GDG matches an existing alias or user catalog, the DEFINE request is denied with a duplicate-name error. This is to prevent the data/index component or a GDS from becoming inaccessible.
- When you add an alias to the catalog, make sure that it does not cause existing data sets to become inaccessible.

With the multilevel alias facility, a non-VSAM data set with the same high-level qualifier as an existing alias of a user catalog can be defined. For more details, see *z/OS DFSMS: Managing Catalogs*.

The selection order is based upon alias names encountered that match the generic data set name, not upon the catalogs or the data set names selected. For LISTCAT, therefore, entries appear in the data set within alias entry order.

Should two or more aliases relate to the same catalog, only the first catalog reference is used.

If no catalogs are found in the prior searches, the master catalog is searched.

**Examples:**

Given that,

| | |
|---|---|
| **Alias A** | is related to ICFUCAT1, |
| **Alias A.B** | is related to ICFUCAT2, |
| **Alias A.C** | is related to ICFUCAT3, |
| **Alias A.C.D** | is related to ICFUCAT4, |
| **Alias B** | is related to SYSCATLG.V338001 and, |

ICFMAST is the master catalog for the system,

1. LISTCAT ENTRY(A.*) selects:
   ICFUCAT1
   ICFUCAT2
   ICFUCAT3
   ICFMAST

   Because the master catalog is selected, the alias entries appear in the listing.
2. LISTCAT ENTRY(B.*) selects:
   SYSCATLG.V338001

The master catalog is not searched.

## Specifying Attribute Selection Order

You can select attributes in more than one way with the DEFINE command. Because more than one value for the same attribute can be given, attributes are selected in the following order of precedence:

1. Explicitly specified attributes
2. Modeled attributes
3. Data class attributes
4. Access method services command defaults

Model processing is done after automatic class selection (ACS) processing. For this reason, modeled attributes are not available to pass to ACS, and default attribute values can be passed to ACS instead. For example, if you state that recordsize be selected from a modeled data set, the AMS default recordsize of 4089 is passed to ACS instead.

The INDEXED|LINEAR|NONINDEXED|NUMBERED parameter is an exception to the attribute selection order. If you do not specify this parameter, the command default (INDEXED) overrides the data class attribute.

# Chapter 2. Modal Commands

With Access Method Services, you can set up jobs to execute a sequence of modal commands with a single invocation of IDCAMS. Modal command execution depends on the success or failure of prior commands.

**In this chapter you will find:**

- Modal commands
- Condition codes
- Examples of how to use modal commands.

**Modal Commands**

You cannot use these commands when Access Method Services is running in Time Sharing Option (TSO).

- IF-THEN-ELSE command sequence, which controls command execution on the basis of condition codes
- NULL command, which causes the program to take no action
- DO-END command sequence, which specifies more than one functional access method services command and its parameters
- SET command, which resets condition codes
- CANCEL command, which ends processing of the current job step
- PARM command, which chooses diagnostic aids and options for printed output.

**Commonly used single job step command sequences**

A sequence of commands commonly used in a single job step includes DELETE-DEFINE-REPRO or DELETE-DEFINE-BLDINDEX.

- You can specify either a data definition (DD) name or a data set name with these commands.
- When you refer to a DD name, allocation occurs at job step initiation. The allocation can result in a job failure, if a command such as REPRO follows a DELETE-DEFINE sequence that changes the location (volser) of the data set. (Such failures can occur with either SMS-managed data sets or non-SMS-managed data sets.)

**Avoiding Potential Command Sequence Failures**

To avoid potential failures with a modal command sequence in your IDCAMS job, perform either one of the following tasks:

- Specify the data set name instead of the DD name
- Use a separate job step to perform any sequence of commands (for example, REPRO, IMPORT, BLDINDEX, PRINT, or EXAMINE) that follow a DEFINE command.

## IF-THEN-ELSE Command Sequence

The syntax of the IF-THEN-ELSE command sequence, which controls command execution, is:

| | |
|---|---|
| **IF** | **{LASTCC\|MAXCC}** *operator number* |

© Copyright IBM Corp. 1973, 2002

| | |
|---|---|
| | **THEN[** *command***\|**<br>**DO**<br>*command set*<br>**END]**<br>**[ELSE[** *command***\|**<br>**DO**<br>*command set*<br>**END]]** |

where:

**IF** States that one or more functional commands should run based on a test of a condition code. A SET command sets the condition code, or the condition code reflects the completion status of previous functional commands.

**Nested IF commands**

When an IF command appears within a THEN or ELSE clause, it is called a nested IF command. To see an example of a nested IF command, see *Using Nested IF Commands, Example 1,*on page 22.

- The maximum level of nesting allowed is 10, starting with the first IF.
- Within a nest of IF commands:
- The innermost ELSE clause is associated with the innermost THEN clause,
- The next innermost ELSE clause with the next innermost THEN clause, and so on. (Each ELSE is matched with the nearest preceding unmatched THEN.)
- If there is an IF command that does not require an ELSE clause, follow the THEN clause with a null ELSE clause (ELSE) unless the nesting structure does not require one.

**LASTCC**
LASTCC specifies that the condition code resulting from the preceding function command be compared, as indicated by the *operator*, to the number that follows the *operator* to determine if the THEN action is to be done.

**MAXCC**
MAXCC specifies that the maximum condition code value established by any previous function command or by a SET command be compared, as indicated by the *operator*, to the number following the *operator* to determine if the THEN action is to be done.

*operator*
*operator* requires a comparison to be made between the variable and *number*. There are six possible comparisons:

Equal to, written as = or EQ
Not equal to, written as ¬= or NE
Greater than, written as > or GT
Less than, written as < or LT
Greater than or equal to, written as >= or GE
Less than or equal to, written as <= or LE

*number*
*number* is the decimal integer that the program compares to MAXCC or LASTCC. Access method services initializes both LASTCC and MAXCC to zero upon entry. See “Condition Codes” on page 23 for the meaning of condition codes.

**THEN**
THEN states that a single command or a group of commands (introduced by DO) is to be run if the comparison is true. THEN can be followed by another IF command.

**ELSE**
ELSE specifies that a single command or a group of commands (introduced by DO) is to be run if the previous comparison is false. ELSE can be followed by another IF command.

## Using Nested IF Commands: Example 1

In this example, nested IF commands are used to determine whether or not a REPRO, DELETE, or PRINT command is run.

```
IF LASTCC > 4  -
  THEN IF MAXCC < 12  -
    THEN REPRO...
    ELSE DELETE...
  ELSE IF LASTCC = 4  -
    THEN
    ELSE PRINT...
```

If LASTCC is greater than 4, MAXCC is tested. If MAXCC is less than 12, the REPRO command is run; if the value of MAXCC is 12 or greater, the DELETE command is run instead. If the value of LASTCC is 4 or less, LASTCC is tested for being exactly 4; if it is, the program takes no action. If LASTCC is less than 4, the program runs the PRINT command.

## Using Nested IF Commands: Example 2

In this example, nested IF commands are used to determine whether the program should run a REPRO command or a PRINT command.

```
IF LASTCC > 4  -
  THEN IF MAXCC < 12  -
    THEN REPRO ...
    ELSE
  ELSE IF LASTCC = 4  -
    THEN PRINT ...
```

If LASTCC is greater than 4, and MAXCC is 12 or greater, no functional commands are run. Use the null ELSE command to indicate that the next ELSE is to correspond to the first THEN.

# Null Command

The null command is a THEN or ELSE command that is not followed by a command continuation character. If THEN or ELSE is not followed by either a continuation character or by a command in the same record, the THEN or ELSE results in no action. The null command supports an ELSE command that balances an IF-THEN-ELSE command sequence, and allows null THEN commands.

If you want to indicate a null ELSE command, say:

```
ELSE
```

If you want to indicate a null THEN command, say:

```
IF ... THEN
ELSE ...
```

Use the null command to indicate that no action is to be taken if the IF clause is satisfied (a null THEN command) or if the IF clause is not satisfied (a null ELSE command).

## DO-END Command Sequence

**DO**
Requires that the group of commands that follow is to be treated as a single unit. That is, the group of commands run as a result of a single IF command. The END command ends the set of commands. A command following a DO must begin on a new line.

**END**
Specifies the end of a set of commands initiated by the nearest unended DO. END must be on a line by itself.

**Restriction:** Do not use continuation characters in the DO-END sequence; they are taken as a null command or cause unpredictable results

## Using the LASTCC Parameter

If the last condition code is 0, the program prints lists a catalog and prints a data set. If the last condition code is greater than 0, the catalog is listed before and after a VERIFY command.

```
IF LASTCC=0
  THEN DO
    LISTCAT
    PRINT INFILE (AJK006)
    END
  ELSE DO
    LISTCAT ENTRY (AJK006) ALL
    VERIFY FILE (AJKJCL6)
    LISTCAT ENTRY (AJK006) ALL
    END
```

## SET Command

Use the SET command to change or reset a previously defined condition code. You can end all processing by setting MAXCC or LASTCC to 16. The syntax of the SET command is:

| | |
|---|---|
| **SET** | **{MAXCC\|LASTCC}=***number* |

where:

**SET**
States that a condition code value is to be set. A SET command that follows a THEN or ELSE that is not run does not alter LASTCC or MAXCC.

**MAXCC**
Requires that the value to be reset is the maximum condition code set by a previous functional command. Setting MAXCC does not affect LASTCC.

**LASTCC**
Specifies that the value to be reset is the condition code set by the immediately preceding functional command.

*number*
Is the value to be assigned to MAXCC or LASTCC. The maximum value is 16; a greater value is reduced to 16. If the value of LASTCC is greater than MAXCC, MAXCC is set equal to the larger value.

## Using the SET command and MAXCC Parameter

In this example, if the maximum condition code is 0, the program lists an entry from a catalog and prints a data set. If the maximum condition code is not 0, set the maximum condition code to 8.

```
IF MAXCC=0
  THEN DO
    LISTCAT  CATALOG (AMASTCAT/MST27) ENT (MN01.B005)
    PRINT INFILE (AJK006)
    END
  ELSE ...
    SET MAXCC=8
```

# CANCEL Command

You can use the CANCEL command to end processing of the current job step. When you use the CANCEL command, the remainder of the command stream is not processed, including any part of an unprocessed IF-THEN-ELSE statement or DO-END pair. The step ends with a return code in register 15 equal to the highest condition code encountered before the CANCEL command was run. A termination message is printed indicating that the CANCEL command was issued. The syntax of the CANCEL command is:

| | |
|---|---|
| **CANCEL** | |

It has no parameters.

## Using the CANCEL Command

In this example, if the maximum condition code is not 0, the maximum condition code is set to 12 and the step ends with CANCEL.

```
IF MAXCC=0
  THEN DO
    LISTCAT  CATALOG (AMASTCAT/MST27) ENT (MN01.B005)
    PRINT INFILE (AJK006)
    END
  ELSE DO
    SET MAXCC=12
    CANCEL
    END
```

# PARM Command

The PARM command specifies processing options to be used during execution. These options remain in effect until changed by another PARM command. You can also use these options in the PARM field of an EXEC statement (in the job control language (JCL)). The syntax of the PARM command is:

| | |
|---|---|
| **PARM** | **[TEST({[TRACE]**<br>**[AREAS(***areaid***[** *areaid***...])]**<br>**[FULL((***dumpid***[** *begin***[** *count***]])**<br>**[(***dumpid***...)...])]\|** |

| | |
|---|---|
| | **OFF})]**<br>**[GRAPHICS(CHAIN(***chain***)\|TABLE(***mname***))]**<br>**[MARGINS(***leftmargin rightmargin***)]** |

where:

**TEST(**

**{[TRACE]**

**[AREAS(***areaid***[** *areaid***...])]**

**[FULL((***dumpid***[** *begin***[** *count***]])**

**[(***dumpid***...)...])]|**

**OFF})**

Specifies the diagnostic aids that the program should use. After the TEST option has been established, it remains in effect until another PARM command resets it. You should use the TRACE, AREAS, and FULL parameters concurrently. See *z/OS DFSMSdfp Diagnosis Reference* for a description of the IDCAMS diagnostic aids and lists of the dump points and area identifiers.

**TRACE**

Specifies that the program should print trace tables whenever it encounters a dump point.

**AREAS(***areaid***[** *areaid***...])**

Lists the modules that are to have selected areas of storage that is dumped at their dump points. *areaid* is a 2-character area identifier defined within the implementation.

**FULL((***dumpid***[** *begin***[** *count***]])[(***dumpid***...)...])**

States that a full region dump, as well as the trace tables and selected areas, is to be provided at the specified dump points. *dumpid* specifies the 4-character identifier of the dump point.

*begin*

Is a decimal integer that specifies the iteration through the named dump point at which the dump is to be produced. (The default is 1.)

*count*

Is a decimal integer that specifies the number of times that the program should produce dumps. (The default is 1.)

If you use the FULL keyword, you must also use an AMSDUMP DD statement; for example:

```
//AMSDUMP DD SYSOUT=A
```

**OFF**

Stops the testing.

**GRAPHICS(CHAIN(***chain***)|TABLE(***mname***))**

Indicates the print chain–graphic character set or a special graphics table that the program should use to produce the output.

**CHAIN(AN|HN|PN|QN|RN|SN|TN)**

Is the graphic character set of the print chains you want to use. The processor usesPN unless the program explicitly directs it to use another set of graphics.

**AN**
Arrangement A, standard EBCDIC character set, 48 characters

**HN**
Arrangement H, EBCDIC character set for FORTRAN and COBOL, 48 characters

**PN**
PL/1 alphanumeric character set

**QN**
PL/1 preferred alphanumeric character set for scientific applications

**RN**
Preferred character set for commercial applications of FORTRAN and COBOL

**SN**
This character set contains lower case and is the preferred character set for text printing

**TN**
Character set for text printing, 120 characters

**TABLE(***mname***)**
Is the name of a table you supply. This 256-byte table defines the graphics for each of the 256 possible bit patterns. Any character sent to the printeris translated to the bit pattern found in the specified table at the position corresponding to its numeric value (0-255). If the print chain does not have a graphic for a byte's bit pattern, the table should specify a period as the output graphic. The table must be stored as a module accessible through the LOAD macro.

**MARGINS(***leftmargin rightmargin***)**
Changes the margins of input records on which command statements are written. The usual left and right margins are 2 and 72, respectively. If you code MARGINS, the program scans all subsequent input records in accordance with the new margins. You can use this function in conjunction with the comment feature: You can use respecification of margins to cause the /* and */ characters to be omitted from the scan. This causes comments to be treated as commands.

*leftmargin*
Locates the location of the left margin.

*rightmargin*
Locates the location of the right margin. The right margin must be greater than the left margin value.

## Using the PARM Command: Example 1

In this example, the program produces dumps on the third and fourth time through the dump point ZZCA.

```
//LISTC JOB ...
//STEP1  EXEC PGM=IDCAMS
//AMSDUMP  DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN  DD  *
     PARM -
        TEST -
        (FULL -
            (ZZCA 03 02))
     LISTCAT -
```

```
          LEVEL(SYS1) -
          ALL
     PARM -
        TEST(OFF)
/*
```

The JCL statement, AMSDUMP DD, describes the dump data set, and is required when FULL is specified.

The PARM command parameters are:

- TEST indicates diagnostic testing is to be done.
- FULL(ZZCA 03 02) requires that a region dump, as well as the trace tables and selected areas, is to be printed the third and fourth time execution passes through dump point ZZCA.

## Using the PARM Command: Example 2

In this example, a dump is produced the first time the program goes through dump points ZZCA or ZZCR:

```
//LISTC JOB ...
//STEP1  EXEC PGM=IDCAMS
//AMSDUMP  DD SYSOUT=A
//SYSPRINT DD SYSOUT=A
//SYSIN  DD  *
     PARM -
        TEST -
        (FULL( -
            (ZZCA 01 01) -
            (ZZCR 01 01)))
     LISTCAT -
          LEVEL(SYS1) -
          ALL
     PARM -
        TEST(OFF)
/*
```

The JCL statement AMSDUMP DD describes the dump data set and is required when FULL is specified.

The parameters are:

- TEST requires diagnostic testing.
- FULL((ZZCA 01 01)(ZZCR 01 01)) states that a region dump, as well as the trace tables and selected areas, is printed the first time through dump points ZZCA and ZZCR.

## Using the PARM Command: Example 3

In this example, selected areas of storage are displayed for all dump points starting with ZZ or LC. An AMSDUMP DD card is not required in this example.

```
//LISTC JOB ...
//STEP1  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN  DD  *
     PARM -
        TEST -
        (AREAS -
             (ZZ LC))
     LISTCAT -
          LEVEL(SYS1) -
```

```
          ALL
     PARM -
        TEST(OFF)
/*
```

The PARM command parameters are:

- TEST indicates diagnostic testing is to be done.
- AREAS(ZZ LC) specifies that trace tables and selected areas of storage are printed. This information is used by service personnel for diagnostic purposes.

## Condition Codes

The csondition codes that are tested in the IF-THEN-ELSE command sequence are:

**0** The function ran as directed and expected. Some informational messages can be issued.

**4** A problem occurred in executing the complete function, but it continued. The continuation might not provide you with exactly what you wanted, but no permanent harm was done. A warning message appears. An example is:

```
The system was unable to locate an entry in a LISTCAT
command.
```

**8** A requested function was completed, but major specifications were unavoidably bypassed. For example, an entry to be deleted or altered could not be found in the catalog, or a duplicate name was found while an entry was being defined and the define action ended.

**12** The program could not perform requested function. The program sets this condition code as a result of a logical error. A logical error condition exists when inconsistent parameters are given, when required parameters are missing, or when a value for key length, record size, or buffer space is too small or too large. More information on logical errors that occur during VSAM record processing is in *z/OS DFSMS Macro Instructions for Data Sets*.

**16** A severe error occurred that erased the remainder of the command stream. This condition code results from one of the following:

- The program cannot open a system output data set. (For example, a SYSPRINT DD statement was missing.)
- An irrecoverable error occurred in a system data set
- An access method services encountered improper IF-THEN-ELSE command sequences.

Condition codes that are tested in the IF-THEN-ELSE command sequence or set by the SET command cannot be passed from one job step to the next. However, the maximum condition code value established by any previous functional command or SET command is passed to the operating system when the access method services processor returns control to the system.

## Common Continuation Errors in Coding Modal Commands

Use continuation rules cautiously when modal commands appear in the input stream. (See “How to Continue Commands and Parameters” on page xxv.) The following examples show common continuation errors:

- ```
  IF LASTCC = 0 -
    THEN
      LISTCAT
  ```
  A continuation mark (hyphen) is missing after the THEN keyword. A null command is assumed after the THEN keyword, and the LISTCAT command is unconditionally run.
- ```
  IF LASTCC = 0 -
    THEN -
      REPRO ...
      /*ALTERNATE PATH*/
    ELSE -
      PRINT ...
  ```
  Because no continuation mark (hyphen) follows the comment, the program assumes a null command. The ELSE keyword will not match the THEN keyword. Note the correct use of the continuation marks on the other records.
- ```
  IF LASTCC = 0 -
    THEN -
      REPRO ...
    ELSE -

      PRINT ...
  ```
  Because a blank line with no continuation mark (hyphen) follows the ELSE keyword, the ELSE becomes null and the PRINT command is unconditionally run.
- ```
  PARM TEST ( - /*COMMENT*/
  TRACE)
  ```
  The program does not continue the PARM command onto the second record, because characters other than blanks appear between the continuation mark (hyphen) and the end of the record.
- ```
  PARM TEST ( TRA+
  /*FIELD CONTINUATION*/
  CE)
  ```
  The processor finds the end of the PARM command after the second record, because no continuation was indicated. The processor rejects command.
  ```

# Chapter 3. Functional Command Syntax

This chapter provides an overview of the access method services functional commands for integrated catalog facility catalogs and for objects that are cataloged in them. The following chapters discuss each command in detail.

Examples of each command appear at the end of each chapter.

See “Notational Conventions” on page xx for an explanation of the symbols used in the command syntax. See “How to Code Subparameters” on page xxiii for coding conventions that apply.

## Functional Command Syntax Summary

This chapter provides reference information about the following functional commands.

*Table 1. Summary of AMS commands*

| Commands | Functions |
|---|---|
| ALLOCATE | Allocates Virtual Storage Access Method (VSAM) and non-VSAM data sets. |
| ALTER | Alters attributes of data sets, catalogs, tape library entries, and tape volume entries that have already been defined. |
| BLDINDEX | Builds alternate indexes for existing data sets. |
| CREATE | Creates tape library entries and tape volume entries. |
| DCOLLECT | Collects data set, volume usage, and migration utility information. |
| DEFINE | Defines the following objects:<br>• **ALIAS** Defines an alternate name for a non-VSAM data set or a user catalog.<br>• **ALTERNATEINDEX** defines an alternate index.<br>• **CLUSTER** Defines a cluster for an entry-sequenced, key-sequenced, linear, or relative record data set.<br>• **GENERATIONDATAGROUP** Defines a catalog entry for a generation data group.<br>• **NONVSAM** Defines a catalog entry for a non-VSAM data set.<br>• **PAGESPACE** Defines an entry for a page space data set.<br>• **PATH** Defines a path directly over a base cluster or over an alternate index and its related base cluster.<br>• **USERCATALOG\|MASTERCATALOG** Defines a user catalog. |
| DELETE | Deletes catalogs, VSAM data sets, and non-VSAM data sets. |

© Copyright IBM Corp. 1973, 2002

*Table 1. Summary of AMS commands (continued)*

| | |
|---|---|
| DIAGNOSE | Scans an integrated catalog facility basic catalog structure (BCS) or a VSAM volume data set (VVDS) to validate the data structures and detect structure errors. |
| EXAMINE | Analyzes and reports the structural consistency of either an index or data component of a key-sequence data set cluster. |
| EXPORT | Disconnects user catalogs, and exports VSAM data sets and integrated catalog facility catalogs. |
| EXPORT DISCONNECT | Disconnects a user catalog. |
| IMPORT | Connects user catalogs, and imports VSAM data sets and integrated catalog facility catalogs. |
| IMPORT CONNECT | Connects a user catalog or a volume catalog. |
| LISTCAT | Lists catalog entries. |
| PRINT | Used to print VSAM data sets, non-VSAM data sets, and catalogs. |
| REPRO | Performs the following functions:<br>• Copies VSAM and non-VSAM data sets, user catalogs, master catalogs, and volume catalogs<br>• Splits integrated catalog facility catalog entries between two catalogs<br>• Merges integrated catalog facility catalog entries into another integrated catalog facility user or master catalog<br>• Merges tape library catalog entries from one volume catalog into another volume catalog. |
| SHCDS | Lists SMSVSAM recovery related to on-line applications and spheres accessed in record-level subscriber (RLS) mode. |
| VERIFY | Causes a catalog to correctly reflect the end of a data set after an error occurred while closing a VSAM data set. The error might have caused the catalog to be incorrect. |

# Chapter 4. ALLOCATE

Access method services identifies the verb name ALLOCATE and attaches the terminal monitor program (TMP) that runs Time Sharing Option (TSO) commands in the background. The ALLOCATE command should be used only to allocate new data sets to the job step. If you use ALLOCATE through access method services for anything else (the handling of SYSOUT data sets, for example), you can get unpredictable results. Refer to *z/OS TSO/E Programming Guide* for additional information on using this command. Table 2 on page 30 separates the parameters to that you should use under access method services from the parameters that cause unpredictable results.

When ALLOCATE is used, the data set is allocated to the job step. If your job contains multiple allocations, you might need to use the DYNAMNBR parameter on the job control language (JCL) EXEC statement. DYNAMNBR establishes a control limit used by TMP when allocating a data set. The control limit is the number of data definition (DD) statements that are coded plus the value coded in DYNAMNBR. If you do not use DYNAMNBR, the system sets it to 0 (the default). If you code DYNAMNBR incorrectly, the system uses the default and issues a JCL warning message. See *z/OS MVS JCL User's Guide* for a description of how to code the DYNAMNBR parameter. For an example illustrating the use of DYNAMNBR, see "Allocate a Data Set Using SMS Class Specifications: Example 1" on page 47.

When you use the ALLOCATE command within access services, you must follow the data set naming conventions of TSO when the TMP is run in batch mode. That is:

- If the data set name is not in quotation marks and a USER parameter is given in the JCL, the value in the USER parameter is prefixed to all data set names given by ALLOCATE.
- If the USER parameter is not in the JCL, no prefix is added to any data set name given by ALLOCATE.

For information about the naming conventions of TSO and other considerations when you use access method services commands from a TSO background job, see *z/OS TSO/E User's Guide*. For information about the USER parameter and its Resource Access Control Facility (RACF) requirements, see *z/OS MVS JCL Reference*.

You can use the ALLOCATE command to define data set attributes in several ways:

- You can use the Storage Management Subsystem (SMS) parameters STORCLAS, MGMTCLAS, and DATACLAS. You can either define these parameters explicitly, or you can let them to use the parameters assigned by the ACS routines that your storage administrator defines. Contact your storage administrator for about storage administration policies and about how the ACS routines might apply.

  You cannot override attributes that the STORCLAS and MGMTCLAS parameters assign. You can override attributes that the DATACLAS parameter assigns. For example, if you use both the DATACLAS parameter and the SPACE parameter, SMS assigns all the attributes defined in the DATACLAS, but uses the values you defined in the SPACE parameter when allocating the data set.

© Copyright IBM Corp. 1973, 2002

- You can use the LIKE parameter to allocate a data set with the same attributes as an existing (model) data set. The model data set must be a cataloged data set. You can override any of the model data set attributes by stating them in the ALLOCATE command.
- You can identify a data set and explicitly describe its attributes.

## Restrictions

- If the access method services job step contains either the SYSTSIN or SYSTSPRT DD statements, the ALLOCATE command is unsuccessful. Access method services allocates the SYSTSIN and SYSTSPRT DD statements to pass the command to the TMP and to retrieve any error messages that are issued. This is done for every ALLOCATE command. Any TMP error messages appear in the SYSPRINT data set, and access method services prints a summary message to show the final status of the command.
- The access method services ALLOCATE command is not supported if access method services is called in the foreground of TSO or if Time Sharing Option Extensions (TSO/E) Release 2 or later is not installed.
- You cannot use ALLOCATE if you have used the ATTACH macro to call IDCAMS from an application program. If you do, ALLOCATE fails with an ATTACH return code.

## Allocating Storage Management Subsystem Managed Data Sets

If SMS is active, it can handle data set storage and management requirements for you. The storage administrator defines SMS classes with ACS routines, which assign classes to a new data set. When a storage administrator assigns a storage class to a new data set, the data set becomes an SMS-managed data set. Data class and management class are optional for SMS-managed data sets. For information on writing ACS routines, see *z/OS DFSMSdfp Storage Administration Reference*.

Your storage administrator writes routines that assign SMS classes to a data set. The SMS classes are:

- *Storage class* Contains performance and availability attributes you can use to select a volume for a data set. You do not need to use the volume and unit parameters for a data set that is SMS-managed.
- *Data class* Contains the attributes related to the allocation of the data set, such as LRECL, RECFM, and SPACE. The data set attributes, if not specified on the ALLOCATE statement, are derived from the model specified on LIKE, or from the data class. If the system cannot allocate the requested amount of space on the eligible volumes in the selected storage group, SMS retries allocation with a reduced space quantity. However, SMS will not do any retries, including reduced space quantity, unless Space Constraint Relief =Y is specified. If the data class assigned to the data set allows space constraint relief, other limits can be bypassed.

  For a list of the attributes for a data class, see the description of the DATACLAS parameter in this section.
- *Management class* Contains the attributes related to the migration and backup of the data set by DFSMShsm™.

## Allocating Non-SMS Managed Data Sets

You can define the DATACLAS parameter to allocate non-SMS-managed data sets. Do not specify the STORCLAS and MGMTCLAS parameters.

## Return Codes for the ALLOCATE Command

| Code | Description |
|---|---|
| **0** | Allocation successful. |
| **12** | Allocation unsuccessful. An error message has been issued. |

Refer to SYSPRINT for the error message.

## Syntax for ALLOCATE Parameters

In Table 2, the access method services ALLOCATE parameters appear in the column “Acceptable Parameters”. Parameters that might cause unpredictable results if used within access method services appear in the column “Parameters to Use with Caution”.

*Table 2. Allocate Command Parameters*

| Command | Acceptable Parameters | Parameters to Use with Caution |
|---|---|---|
| **ALLOCATE** | {**DATASET**(*dsname*)[**FILE**(*ddname*)]} | {*\|*dsname-list*}\|**DUMMY** |
| | [**ACCODE**(*access code*)][1] | |
| | [**ALTFILE**(*name*)] | |
| | [**AVGREC**(*U*\|*K*\|*M*)] | |
| | [**BFALN**(*F*\|*D*)][2] | |
| | [**BFTEK**(*S*\|*E*\|*A*\|*R*)][2] | |
| | [**BLKSIZE**(*value*)][2] | |
| | [**BUFL**(*buffer-length*)][2] | |
| | [**BUFNO**(*number-of-buffers*)] | |
| | [**BUFOFF**({*block-prefix-length*\|*L*)]}[2] | |
| | | [**BURST\|NOBURST**] |
| | | [**CHARS**[*tablename-list*]] |
| | | [**COPIES**((*number*),[*group-value-list*])] |
| | [**DATACLAS**(*data-class-name*)] | |
| | [**DEN**(*0*\|*1*\|*2*\|*3*\|*4*)][1] | |
| | | [**DEST**(*destination*\|*destination.userid*)] |
| | [**DIAGNS**(*TRACE*)][2] | |
| | [**DIR**(*integer*)] | |
| | [**DSNTYPE(LIBRARY\|PDS)**] | |
| | [**DSORG**(*DA*\|*DAU*\|*PO*\|*POU*\|*PS*\|*PSU*)][2] | |
| | [**EROPT**(*ACC*\|*SKP*\|*ABE*)] | |
| | [**EXPDT**(*year-day*)\|**RETPD**(*no.-of-days*)] | |
| | | [**FCB**(*image-id*,**ALIGN,VERIFY**)] |
| | | [**FLASH**(*overlay-name*,[*copies*])] |
| | | [**FORMS**(*forms-name*)] |
| | | [**HOLD**\|**NOHOLD**] |
| | | [**INPUT\|OUTPUT**] |
| | [**KEEP\|CATALOG**] | [**DELETE\|UNCATALOG**] |

*Table 2. Allocate Command Parameters (continued)*

| Command | Acceptable Parameters | Parameters to Use with Caution |
|---|---|---|
| | [**KEYLEN**(*bytes*)] | |
| | [**KEYOFF**(*offset*)] | |
| | [**LABEL**(*type*)][1] | |
| | [**LIKE**(*model-dsname*)] | \|[**USING**(*attr-list-name*)] |
| | [**LIMCT**(*search-number*)] | |
| | [**LRECL**({*logical-record-length*\| | |
| | (*nnnnnK*\|*X*)})] | |
| | [**MGMTCLAS**(*management-class-name*)] | |
| | [**MAXVOL**(*count*)] | |
| | | [**MODIFY**(*module-name*,[*trc*])] |
| | [**NEW**] | [**OLD\|SHR\|MOD**] |
| | [**NCP**(*no.-of-channel-programs*)][2] | |
| | [**OPTCD**(*A,B,C,E,F,H,J,Q,R,T,W,Z*)][2] | |
| | | [**OUTDES**(*output-descriptor-name*,...)] |
| | [**POSITION**(*sequence-no.*)][1] | |
| | [**PRIVATE**] | |
| | [**PROTECT**] | |
| | [**RECFM**(*A,B,D,F,M,S,T,U,V*)][2] | |
| | [**RECORG**(*ES*\|*KS*\|*LS*\|*RR*)] | |
| | [**REFDD**(*file-name*)] | |
| | [**RELEASE**][2] | |
| | [**REUSE**] | |
| | [**ROUND**][2] | |
| | [**SECMODEL**(*profile-name*[,**GENERIC**])] | |
| | [**SPACE**(*quantity*[,*increment*]) | |
| | {**BLOCK**(*value*)\|**AVBLOCK**(*value*)\| | |
| | **CYLINDERS**\|**TRACKS**}] | |
| | [**STORCLAS**(*storage-class-name*)] | |
| | | [**SYSOUT**(*class*)] |
| | [**TRTCH**(*C*\|*E*\|*ET*\|*T*)][1] | |
| | [**UCOUNT**(*count*)\|**PARALLEL**] | |
| | | [**UCS**(*universal-character-set-name*)] |
| | [**UNIT**(*type*)] | |
| | [**VOLUME**(*serial-list*)] | |
| | [**VSEQ**(*vol-seq-no.*)] | |
| | | [**WRITER**(*external-writer-name*)] |

[1] Parameters applicable to tape data sets only.
[2] Parameters applicable to non-VSAM data sets only.

**Abbreviation for ALLOCATE command**: ALLOC

Descriptions of the parameters within access method services follow. For information about ALLOCATE parameters not described in this section, see *z/OS TSO/E Command Reference*.

## Required Parameters

**DATASET(***dsname***)**

Gives the name of the data set to be allocated. The data set name must be fully qualified.

- You must follow the data set naming conventions of TSO when you run TMP in batch mode.
- All temporary SMS-managed data sets must either have a DSNAME starting with & or &&, or have no DSNAME at all.

  Non-VSAM temporary data sets are the only uncataloged data sets that you can create.

  For more information about temporary data sets, see *z/OS MVS JCL Reference*. For more information about VSAM temporary data sets, see *z/OS DFSMS: Using Data Sets*.
- You cannot concurrently allocate data sets that reside on the same physical tape volume.
- To allocate a member of a generation data group, provide the fully qualified data set name, including the generation number.

**Abbreviation:** DA, DSN, DSNAME

**FILE(***ddname***)**

This is the name of the data set, and can have up to eight characters. If you omit this parameter, the system assigns an available system file name (ddname). Do not use special ddnames unless you want to use the facilities those names represent to the system. See “JCL DD Statement for a Snap Dump” on page 3 for more information about AMSDUMP. See *z/OS MVS JCL Reference* for more information about the following special ddnames:

```
AMSDUMP         SYSABEND
JOBCAT          SYSCHK
JOBLIB          SYSCKEOV
STEPCAT         SYSMDUMP
STEPLIB         SYSUDUMP
```

See *z/OS TSO/E Command Reference* for more information on these special ddnames:

```
SYSTSIN         SYSTSPRT
```

You cannot use SYSTSIN and SYSTSPRT in a job step that runs the ALLOCATE command. See “Restrictions” on page 28 for further information.

## Optional Parameters

**ACCODE(***access code***)**

Assigns the accessibility code for an ISO/ANSI output tape data set, which protects it from unauthorized use. You can use up to eight characters in the access code, but ISO/ANSI validates only the first character. The ACCODE can now be any of the following 57 ISO/ANSI a-type characters: blank, upper case letters A-Z, numeric 0-9, or one of the special characters !*″%&’()+,-./:;<=>?_ Note that Version 3 still only supports upper case A-Z. Password protection is

supported for ANSI tape data sets under the PASSWORD/NOPWREAD options on the LABEL parameter. Password access overrides any ACCODE value if you use both options.

**ALTFILE(*name*)**
The name of the SYSIN subsystem data set that is to be allocated, and can be up to eight characters. The system uses this parameter primarily in the background.

This gives the length in bytes of the average block.

**AVGREC(*U*|*K*|*M*)**
Determines the size of the average record block. You can use the following values:

**U** Use the primary and secondary quantities as given on the SPACE parameter.

**K** Multiply primary space quantity and secondary space quantity by 1024 (1 KB).

**M** Multiply primary space quantity and secondary space quantity by 1,048,576 (1 MB).

Use the AVGREC parameter to define a new data set when:

- The units of allocation that is requested for storage space are records.
- The primary and secondary space quantities used with the SPACE parameter represent units, thousands, or millions of records.

When you use AVGREC with the SPACE parameter, the first subparameter for the SPACE parameter must give the average record length of the records.

Use the AVGREC parameter when you want to show records as the units of allocation. You can also use the AVGREC parameter to override the space allocation defined in the data class for the data set.

If SMS is not active, the system checks the syntax and then ignores the AVGREC parameter.

**BFALN(*F*|*D*)**
Gives is the boundary alignment of each buffer:

**F** Each buffer starts on a fullword boundary that might not be a doubleword boundary.

**D** Each buffer starts on a doubleword boundary.

If you do not use this parameter, the system defaults to a doubleword boundary.

**BFTEK(*S*|*E*|*A*|*R*)**
Is the type of buffering that you want the system to use, such as:
**S** Simple buffering
**E** Exchange buffering
**A** Automatic record area buffering
**R** Record buffering

BFTEK(R) is not compatible with partitioned data sets extended (PDSE) and results in an error if used with the DSNTYPE(LIBRARY) parameter.

**BLKSIZE(*value*)**
The data control block (DCB) block size for the data set. The maximum

allowable decimal for the block size that is recorded in the DCB is 32,760. You can specify BLKSIZE for NEW or MOD data sets.

*For direct access storage device (DASD) data sets:* If you do not use BLKSIZE, the system determines an optimal DCB block size for the new data set. To create the DCB block size:

- The system determines the block size if SMS is active and you do not assign the block size.
- You can assign the block size through the BLKSIZE parameter.
- You can use the LIKE parameter to obtain the block size from an existing model data set.
- If you do not assign BLKSIZE or LIKE, the system can determine the block size from the BLOCK parameter.

The block size that you assign for the DCB must be consistent with the requirements of the RECFM parameter. If you use:

- RECFM(F), the block size must be equal to, or greater than, the logical record length.
- RECFM(FB), the block size must be an integral multiple of the logical record length.
- RECFM(V), the block size must be equal to, or greater than, the largest block in the data set. (For unblocked variable-length records, the size of the largest block must allow space for the 4-byte block descriptor word, in addition to the largest logical record length. The logical record length must allow space for a 4-byte record descriptor word.)
- RECFM(VB), the block size must be equal to, or greater than, the largest block in the data set. For block variable-length records, the size of the largest block must allow space for the 4-byte block descriptor word, in addition to the sum of the logical record lengths that will go into the block. Each logical record length must allow space for a 4-byte record descriptor word.

  Because the number of logical records can vary, estimate the optimum block size and the average number of records for each block, based on your knowledge of the application that requires the I/O.
- RECFM(U) and BLKSIZE(80), one character is truncated from the line. That character (the last byte) is reserved for an attribute character.

For PDSEs:

- The system chooses the BLKSIZE if you do not explicitly specify it. If BLKSIZE is given, the system treats the BLKSIZE as the length of the simulated block. For create mode processing, the logical record length is equal to the block size if LRECL is not given. If you use LRECL, BLKSIZE must conform to the LRECL and RECFM definitions. If you use:

| | |
|---|---|
| **RECFM(F)** | BLKSIZE must equal LRECL |
| **RECFM(FB) or RECFM(FBS)** | BLKSIZE must be a multiple of LRECL |
| **RECFM(V) or RECFM(VB)** | BLKSIZE must be at least four bytes larger than LRECL |
| **RECFM(VBS)** | BLKSIZE must be at least eight bytes. |

- For input or update processing, the block size must conform to the currently defined record length. The BLKSIZE given when the data set was created is the default. However, you can use any BLKSIZE if it conforms to the record length definition.

**BUFL(***buffer-length***)**
The length, in bytes, of each buffer in the buffer pool. Substitute a decimal number for *buffer-length*. The number must not exceed 32,760. If you omit this parameter and the system acquires buffers automatically, the BLKSIZE and KEYLEN parameters supply the information needed to establish buffer length.

**BUFNO(***number-of-buffers***)**
The number of buffers that are assigned for data control blocks. Substitute a decimal number for number-of-buffers. The number must never exceed 255. You can be limited to a smaller number of buffers depending on the limit established when the operating system was generated. The following shows how to get a buffer pool and the action required:

| Method | Action |
|---|---|
| **BUILD macro instruction** | You must use BUFNO |
| **GETPOOL macro instruction** | The system uses the number that you assign for GETPOOL |
| **Automatically with BPAM, BSAM** | You must use BUFNO |
| **Automatically with QSAM** | You can omit BUFNO and accept two buffers |

**BUFOFF({***block-prefix-length***|***L***})**
Defines the buffer offset. The *block-prefix-length* must not exceed 99. L specifies the block prefix field is 4 bytes long and contains the block length.

**DATACLAS(***data-class-name***)**
This is the 1-to-8 character name of the data class for either SMS or non-SMS-managed data sets. If you do not assign DATACLAS for a new data set and the storage administrator has provided an automatic class selection (ACS) routine, the ACS routine can select a data class for the data set. If you assign DATACLAS for an existing data set, SMS ignores it. If SMS is not active, the system checks the syntax and then ignores the DATACLAS parameter.

If you use the data class, you do not need to list all the attributes for a data set. For example, the storage administrator can provide RECFM, LRECL, RECORG, KEYLEN, and KEYOFF as part of the data class definition. However, you can override the DATACLAS parameter by explicitly defining the appropriate parameters in the ALLOCATE command.

The data class defines these data set allocation attributes:

- Data set organization:
  - Record organization (RECORG)
  - Record format (RECFM)
- Record length (LRECL)
- Key length (KEYLEN)
- Key offset (KEYOFF)
- Space allocation
  - AVGREC
  - SPACE
- Expiration date (EXPDT) or retention period (RETPD)
- Volume count (VOLUME)
- For VSAM data sets, the following:
  - Control interval size (CISIZE)

– Percent free space (FREESPACE)
– Sharing options (SHAREOPTIONS)

SHAREOPTIONS is assumed to be (3,3) when you use RLS.

*Table 3. Data Class Attributes vs. Data Set Organization*

| Attributes | KS | ES | RR | LDS |
|---|---|---|---|---|
| CISIZE | X | X | X | X |
| FREESPACE | X | | | |
| KEYLEN | X | | | |
| KEYOFF | X | | | |
| LRECL | X | X | X | |
| SHAREOPTIONS | X | X | X | X |
| SPACE | X | X | X | X |
| Volume Count | X | X | X | X |

**DEN(***0***|***1***|***2***|***3***|***4***)**
Gives the magnetic tape density as follows:
**0** 200 bpi/7 track
**1** 556 bpi/7 track
**2** 800 bpi/7 and 9 track
**3** 1600 bpi/9 track
**4** 6250 bpi/9 track (IBM® 3420 Models 4, 6, and 8)

**DIAGNS(***TRACE***)**
The Open/Close/EOV trace option that gives a module-by-module trace of the Open/Close/EOV work area and your DCB.

**DIR(***integer***)**
Gives the number of 256 byte records for the directory of a new partitioned data set. You must use this parameter to allocate a new partitioned data set.

**DSNTYPE(LIBRARY|PDS)**
Determines allocation of either a partitioned data set (PDS) or a partitioned data set extended (PDSE). A PDSE must be SMS-managed. If SMS is not active, the system checks the syntax and then ignores the DSNTYPE parameter.
**LIBRARY** A PDSE in record format

For more information on PDSE, see *z/OS DFSMS: Using Data Sets*.

**DSORG(***DA***|***DAU***|***PO***|***POU***|***PS***|***PSU***)**
The data set organization as:
**DA** Direct access
**DAU** Direct access unmovable
**PO** Partitioned organization
**POU** Partitioned organization unmovable
**PS** Physical sequential
**PSU** Physical sequential unmovable

When you allocate a new data set and do not use the DSORG parameter, these occur:

- If you assign a non-zero to the DIR parameter, DSORG defaults to the partitioned organization (PO) option.
- If you do not assign a value to the DIR parameter, DSORG defaults to the physical sequential (PS) option.

- The system does not store default DSORG information into the data set until a program opens and writes to the data set.

With PDSEs, the PSU and POU options are incompatible and result in an error if used with DSNTYPE(LIBRARY) while the data set is open for output. If the data set is open for input or update, PSU and POU are ignored.

To indicate the data set organization for VSAM data sets, see RECORG.

**EROPT(***ACC***|***SKP***|***ABE***)**
The option you want to run if an error occurs when the system reads or writes a record. They are:
**ACC** Accept the block of records in which the error was found
**SKP** Skip the block of records in which the error was found
**ABE** End the task abnormally

**EXPDT(***year-day***)|RETPD(***no.-of-days***)**
Expiration date or the retention period. The MGMTCLAS maximum retention period, if given, limits the retention period in this parameter. The system ignores these parameters for temporary data sets.

**EXPDT(***year-day***)**
This shows the data set expiration date. Include the year and day as either:

- *yyddd*, where *yy* is the last two digits of the year and *ddd* is the three-digit number for the day of the year. The maximum for the year is 99 (for 1999), and for the day is 366.

  If you enter 99365 or 99366, the system retains your data sets permanently. Do not use those dates as expiration dates. Use them as no-scratch dates only.
- *yyyy/ddd*, where *yyyy* is the four-digit number for the year and *ddd* is the three-digit number for the day of the year. This syntax requires a slash. The maximum for the year is 2155. The maximum for the day is 366.

  If you use 1999/365 or 1999/366, the system retains your data sets permanently. Do not use those dates as an expiration date. Use them as no-scratch dates only.

EXPDT and RETPD are mutually exclusive.

**RETPD(***no.-of-days***)**
Data set retention period in days. It can be a one-to-four-digit decimal number.

RETPD and EXPDT are mutually exclusive.

**KEEP|CATALOG**
A command processor can modify the final disposition with these parameters.

**KEEP**
This retains the data set by the system after step termination.

**CATALOG**
This retains the data set in a catalog after step termination.

**KEYLEN(***bytes***)**
This is the length, in bytes, of each of the keys used to locate blocks of records in the data set when the data set resides on a direct access device.

If an existing data set has standard labels, you can omit this parameter and let the system retrieve the key length from the standard label. If a key length is not

supplied by any source before you enter, the system assumes an OPEN macro instruction of zero (no keys). This parameter is mutually exclusive with TRTCH.

When you want to define the key length or override the key length defined in the data class (DATACLAS) of the data set, use KEYLEN. The number of bytes is:

- 1 to 255 for a record organization of key-sequenced (RECORG(KS))
- 0 to 255 for a data set organization of physical sequential (PS) or partitioned (PO)

For PDSEs, you can use 0 or 8. Use 8 only when opening the PDSE for input. Any other value results in an error.

**KEYOFF(***offset***)**
This shows the key position (offset) of the first byte of the key in each record. Use it to define key offset or override the key offset defined in the data class of the data set. It is only for a key-sequenced data set (RECORG=KS).

Use KEYOFF parameter to allocate both SMS-managed and non-SMS-managed data sets. If SMS is not active, however, the system checks syntax and then ignores the KEYOFF parameter.

**LABEL(***type***)**
This selects the label processing, one of: SL, SUL, AL, AUL, NSL, NL, LTM, or BLP, which correspond to the JCL label-types.

For VSAM data sets, the system always uses SL, whether you define SL or SUL or neither. NSL, NL, and BLP do not apply to VSAM data sets.

**LIKE(***model-dsname***)**
This names a model data set. The system uses these attributes as the attributes of the new data set that is being allocated. The model data set must be cataloged and must reside on a direct access device. The volume must be mounted when you enter the ALLOCATE command.

**Note:** TSO naming conventions apply when assigning *model-dsname*.

When the ALLOCATE command assigns attributes to a new data set, these attributes are copied from the model data set if SMS is active:

| | |
|---|---|
| **AVGREC** | Size of average record block (kilobyte, megabyte) |
| **BLOCK, AVBLOCK, TRACKS, CYLINDERS** | Space unit |
| **DIR** | Directory space quantity |
| **DSORG** | Non-VSAM data set organization |
| **KEYLEN** | Key length |
| **KEYOFF** | Key offset |
| **LRECL** | Logical record length |
| **RECFM** | Record format |
| **RECORG** | VSAM data set organization |
| **SPACE** | Primary and secondary space quantities. |

The system copies these attributes only if SMS is not active:

| | |
|---|---|
| **BLKSIZE** | Block size |
| **EXPDT** | Data set expiration date |
| **OPTCD** | Optional services code (for ISAM data sets only) |
| **VSEQ** | Volume sequence number. |

You can still use the LIKE parameter even if you do not have an existing data set with the exact attributes you want to assign to a new data set. You can use ALLOCATE attributes to override any model data set attributes you do not want assigned to the new data set.

When you use this:

- LIKE must be used with the NEW parameter; it cannot be used with OLD, SHR, or MOD.
- Use LIKE with the DATASET parameter; it cannot be used with FILE.
- Only one *dsname* can be given in the DATASET parameter.
- The system does not copy the block size from the model data set when SMS is active. If you do not show a block size in the ALLOCATE command, the system determines an optimal block size to assign to the data set.
- When SMS is active, attributes copied from the model data set override attributes from the data class.
- If you allocate the new data set with a member name (indicating a partitioned data set), the system prompts you for directory blocks unless that quantity is either shown in the ALLOCATE command or defaulted from the LIKE data set.
- If the new data set name is indicated with a member name, but the model data set is sequential and you have not given the quantity for directory blocks, you are prompted for directory blocks.

If you define the directory value as zero and the model data set is a PDS, the system allocates the new data set as a sequential data set.

The LIKE, REFDD, and USING operands are mutually exclusive. Refer to *z/OS TSO/E Command Reference* for additional information on the USING operand.

**LIMCT(***search-number***)**
This is the number of blocks or tracks that the system should search for a block or available space. The number must not exceed 32760.

**LRECL({***logical-record-length***|(***nnnnnK***|***X***)})**
This is the length, in bytes, of the largest logical record in the data set. You must define this parameter for data sets that consist of either fixed-length or variable-length records.

Use the DATACLAS parameter in place of LRECL to assign the logical record length. If SMS is active and you use LRECL, the system determines the block size.

If the data set contains undefined-length records, omit LRECL.

The logical record length must be consistent with the requirements of the RECFM parameter and must not exceed the block size (BLKSIZE parameter), except for variable–length spanned records. If you use:

- RECFM(V) or RECFM(V B), then the logical record length is the sum of the length of the actual data fields plus four bytes for a record descriptor word.
- RECFM(F) or RECFM(F B), then the logical record length is the length of the actual data fields.
- RECFM(U), omit the LRECL parameter.

LRECL(nnnnnK) allows users of ANSI extended logical records and users of QSAM “locate mode” to assign a K multiplier to the LRECL parameter. *nnnnn* can be a number within 1-16384. The K indicates that the value is multiplied by 1024.

For variable-length spanned records (VS or VBS) processed by QSAM (locate mode) or BSAM, use LRECL (X) when the logical record exceeds 32,756 bytes.

For PDSEs, the meaning of LRECL depends upon the data set record format:

- **Fixed Format Records**. For PDSEs opened for output, the logical record length (LRECL) defines the record size for the newly created members. You cannot override the data set control block (DSCB) (LRECL); an attempt to do so will result in an error.
- **Variable Format Records**. The LRECL is the maximum record length for logical records that are contained in members of the PDSE.
- **Undefined Format Records**. The LRECL is the maximum record length for records that are contained in members of the PDSEs.

**MGMTCLAS(***management-class-name***)**
*For SMS-managed data sets:* This is the 1-to-8 character name of the management class for a new data set. When possible, do not use MGMTCLAS. Allow it to default through the ACS routines.

After the system allocates the data set, attributes in the management class define:

- The migration of the data set. This includes migration both from primary storage to migration storage, and from one migration level to another in a hierarchical migration scheme.
- The backup of the data set. This includes frequency of backup, number of versions, and retention criteria for backup versions.

If SMS is not active, the system checks the syntax and ignores the MGMTCLAS parameter.

**MAXVOL(***count***)**
This is the maximum number (1-255) of volumes upon which a data set can reside. This number corresponds to the count field on the VOLUME parameter in JCL. Use this to override the volume count attribute defined in the data class of the data set.

If VOLUME and PRIVATE parameters are not given, and MAXVOL exceeds UCOUNT, the system removes no volumes when all the mounted volumes have been used, causing abnormal termination of your job. If PRIVATE is given, the system removes one of the volumes and mounts another volume in its place to continue processing.

MAXVOL overrides any volume count in the data class (DATACLAS) of the data set.

Your user attribute data set (UADS) must contain the MOUNT attribute. Use of this parameter implies PRIVATE.

**NEW**
This creates a data set. For new partitioned data sets, you must use the DIR parameter. If you assign a data set name, the system keeps and catalogs a NEW data set. If you do not assign a data set name, the system deletes the data set at step termination.

**NCP(***number-of-channel-programs***)**
This gives the maximum number of READ or WRITE macro instructions that are allowed before a CHECK macro instruction is entered. The number must not exceed 99 and must be less than 99 if a lower limit was established when the operating system was generated. If you are using chained scheduling, you must assign an NCP value greater than 1. If you omit the NCP parameter, the default value is 1.

**OPTCD(***A,B,C,E,F,H,J,Q,R,T,W,Z***)**
This lists optional services: (See also the OPTCD subparameter of the DCB parameter in *z/OS MVS JCL Reference* for details.)

**A** Requires the actual device addresses be presented in READ and WRITE macro instructions.

**B** Requires the end-of-file (EOF) recognition be disregarded for tapes.

**C** Uses chained scheduling.

**E** Asks for an extended search for block or available space.

**F** Returns device address feedback from a READ or WRITE macro instruction in the form it is presented to the control program.

**H** Requests the system to check for and bypass. For further information, see *z/OS MVS JCL Reference*.

**J** Makes the character after the carriage control character the table reference character for that line. The table reference character tells TSO which character arrangement table to select when printing the line.

**Q** Translates a magnetic tape from ASCII to EBCDIC or from EBCDIC to ASCII.

**R** Requires relative block addressing.

**T** Requests the user totaling facility.

**W** Tells the system to perform a validity check when it writes data on a direct access device.

**Z** Asks the control program to shorten its normal error recovery procedure for input on magnetic tape.

You can use any or all the services by combining the characters in any sequence, separating them with blanks or commas.

For PDSEs, the system ignores OPTCD values other than OPTCD(J). OPTCD(J) requires that the first data byte in the output data line is a 3800 table reference character.

**POSITION(***sequence-no.***)**
This is the relative position (1-9999) of the data set on a multiple data set tape. The sequence number corresponds to the data set sequence number field of the label parameter in JCL.

**PRIVATE**
This assigns the private–volume use attribute to a volume that is neither reserved nor permanently in resident. It corresponds to the PRIVATE keyword of the VOLUME parameter in JCL.

If you do not use VOLUME and PRIVATE parameters and MAXVOL exceeds UCOUNT, the system removes no volumes when all the mounted volumes have

been used, causing abnormal termination of your job. If you use PRIVATE, the system removes one of the volumes and mounts another volume to continue processing.

**PROTECT**
This RACF-protects the DASD data set or the first data set on a tape volume.

- For a new permanent DASD data set, the status must be NEW or MOD, treated as NEW, and the disposition must be either KEEP, CATALOG, or UNCATALOG. With SMS, SECMODEL overrides PROTECT.
- For a tape volume, the tape must have an SL, SUL, AL, AUL, or NSL label. The file sequence number and volume sequence number must be one (except for NSL). You must assign PRIVATE as the tape volume use attribute.

The PROTECT parameter is not valid if a data set name is not given, or if the FCB parameter or status other than NEW or MOD is used.

**RECFM(***A,B,D,F,M,S,T,U,V***)**
This sets the format and characteristics of the records in the data set. They must be completely described by one source only. If they are not available from any source, the default is an undefined-length record. See also the RECFM subparameter of the DCB parameter in *z/OS MVS JCL Reference* for a detailed discussion.

Use these with the RECFM parameter:

**A** To show the record contains ASCII printer control characters

**B** To indicate the records are blocked

**D** For variable length ASCII records

**F** For fixed length records.

**M** For records with machine code control characters.

**S** For fixed-length records, the system writes the records as standard blocks (there must be no truncated blocks or unfilled tracks except for the last block or track). For variable-length records, a record can span more than one block. Exchange buffering, BFTEK(E), must not be used.

**T** The records can be written onto overflow tracks, if required. Exchange buffering, BFTEK(E), or chained scheduling, OPTCD(C), cannot be used.

**U** The records are of undefined length.

**V** Shows variable length records.

You must provide one or more values for this parameter.

For PDSEs, these statements apply:

- RECFM can be partially modified from the value that is saved in the DSCB when creating members.
- In a PDSE that is created as fixed or fixed blocked, members must always be created with fixed–length logical records. However, the attribute of blocked might change between member creates. The first record format assigned to the PDSE is the default for member creates. The characteristic of blocked might not change during an open.
- Attempts to overwrite the record format characteristic of F, U, or V with another value from that set causes a system error.

- RECFM(A) and RECFM(M) are compatible with PDSEs.

RECFM and RECORG are mutually exclusive.

**RECORG(***ES***|***KS***|***LS***|***RR***)**
Determines the organization of the records in a new VSAM data set. To override the record organization defined in the data class (DATACLAS) of the data set, use RECORG.

You can assign:

**ES** For a VSAM entry-sequenced data set

**KS** For a VSAM key-sequenced data set

**LS** For a VSAM linear space data set. You cannot access linear data sets with VSAM record level sharing (RLS).

**RR** For a VSAM relative record data set

If you do not use RECORG, SMS assumes a non-VSAM data set.

RECORG and RECFM are mutually exclusive. To define the data set organization for a non-VSAM data set, see DSORG.

**Exception:** You can use the RECORG parameter to allocate both SMS-managed and non-SMS-managed data sets. If SMS is not active, however, the system checks the syntax and ignores the RECORG parameter.

**REFDD(***file-name***)**
This is the file name of an existing data set whose attributes are copied to a new data set. The system copies these attributes to the new data set:
- Data set organization:
  - Record organization (RECORG)
  - Record format (RECFM)
- Record length (LRECL)
- Key length (KEYLEN)
- Key offset (KEYOFF)
- Space allocation
  - AVGREC
  - SPACE

The system does not copy the retention period (RETPD) or expiration date (EXPDT) to the new data set.

LIKE and REFDD are mutually exclusive.

**Exception:** You can use the REFDD parameter to allocate both SMS-managed and non-SMS-managed data sets. If SMS is not active, however, the system checks the syntax and then ignores the REFDD parameter.

**RELEASE**
To delete unused space when the data set is closed.

If you use RELEASE for a new data set with the BLOCK or BLKSIZE parameter, then you must also use the SPACE parameter.

**REUSE**
Frees and reallocates the file name if it is currently in use.

You cannot use the REUSE parameter to reallocate a file from a disposition of OLD to a disposition of SHR. However, you can first free the file with OLD, then reallocate it with SHR.

**ROUND**
Allocates space equal to one or more cylinders. Use this only when you request space in units of blocks. This parameter corresponds to the ROUND parameter in the SPACE parameter in JCL.

**SECMODEL(*profile-name*[,GENERIC])**
Names an existing RACF profile to copy to the discrete profile. Use SECMODEL when you want a different RACF data set profile from the default profile selected by RACF, or when there is no default profile. The model profile can be a:

- RACF model profile
- RACF discrete data set profile
- RACF generic data set profile

Use GENERIC to state the profile name as a generic data set profile.

The system copies this information from the RACF data set profile to the discrete data set profile of the new data set:

- OWNER indicates the user or group assigned as the owner of the data set profile.
- ID is the access list of users or groups that are authorized to access the data set.
- UACC gives universal access authority that is associated with the data set.
- AUDIT|GLOBALAUDIT selects which access attempts are logged.
- ERASE indicates that the data set when it is deleted (scratched).
- LEVEL is the installation-defined level indicator.
- DATA is installation-defined information.
- WARNING indicates that an unauthorized access causes RACF to issue a warning message, but allows access to the data set.
- SECLEVEL is the name of an installation-defined security level.

**Exception:** You can use the SECMODEL parameter to allocate both SMS-managed and non-SMS managed data sets. If SMS is not active, however, the system checks the syntax and then ignores the SECMODEL parameter.

For more information about RACF, see *z/OS Security Server RACF Command Language Reference*.

**SPACE(*quantity*[,*increment*])**
Allocates the amount of space for a new data set. If you omit this parameter and:

- You are running MVS/ESA™ Version 3. The system uses the IBM-supplied default value of SPACE(10,50) AVBLOCK (1000).
- You are running MVS/ESA SP™ Version 4. The system uses the IBM-supplied default value of SPACE(4,24) AVBLOCK (8192).

However, your installation might have changed the default. For more information about default space, see *z/OS MVS Programming: Authorized Assembler Services Guide*.

To have the system determine the amount of space, include the AVGREC parameter in place of BLOCK, AVBLOCK, CYLINDERS, and TRACKS. To supply your own space value, define one of the following: BLOCK(*value*), BLKSIZE(*value*), AVBLOCK(*value*), CYLINDERS, or TRACKS. The amount of space requested is determined as follows:

- BLOCK(*value*) or BLKSIZE(*value*): The BLOCK or BLKSIZE parameter's *value* is multiplied by the SPACE parameter's *quantity*.
- AVBLOCK(*value*): The AVBLOCK parameter's *value* is multiplied by the SPACE parameter's *quantity*.
- CYLINDERS: The SPACE parameter's *quantity* is given in cylinders.
- TRACKS: The SPACE parameter's *quantity* is given in tracks.

Use SPACE for NEW and MOD data sets.

*quantity*
: Allocates the initial number of units of space for a data set. For a partitioned data set, a directory quantity is not necessary.

*increment*
: This is the number of units of space to be added to the data set each time the previously allocated space has been filled. You must provide the primary quantity along with the increment value.

**BLOCK(***value***)**
: Shows the average length (in bytes) of the records written to the data set. The maximum block value used to determine space to be allocated is 65,535. The block value is the unit of space that is used by the SPACE parameter. A track or a cylinder on one device can represent a different amount of storage (number of bytes) from a track or a cylinder on another device. Determine the unit of space value from the:
  - Default value of (10 50) AVBLOCK(1000) if no space parameters (SPACE, AVBLOCK, BLOCK, CYLINDERS, or TRACKS) are given.
  - The BLOCK parameter.
  - The model data set, if the LIKE parameter is used and BLOCK, AVBLOCK, CYLINDERS, or TRACKS is not given.
  - The BLKSIZE parameter if BLOCK is not used.

**AVBLOCK(***value***)**
: This shows only the average length (in bytes) of the records that are written to the data set.

**CYLINDERS**
: Requests allocation in cylinders as the unit of space.

**TRACKS**
: Requests allocation in tracks as the unit of space.

**Exception:** If you specify tracks for a VSAM data set, the space allocated will be contiguous. See *z/OS DFSMS: Using Data Sets* "Optimizing Control Area Size," for more information.

**STORCLAS(***storage-class-name***)**
: *For SMS-managed data sets:* Gives the 1-to-8 character name of the storage class. When possible, allow STORCLAS to default through the ACS routines established by your storage administrator. Attributes assigned through storage class and the ACS routines replace storage attributes such

as UNIT and VOLUME. If SMS is not active, the system checks the syntax and then ignores the STORCLAS parameter.

**TRTCH(***C*|*E*|*ET*|*T***)**
Selects the recording technique for 7-track tape as follows:

**C** Data conversion with odd parity and no translation.

**E** Even parity with no translation and no conversion.

**ET** Even parity and no conversion. BCD to EBCDIC translation when reading, and EBCDIC to BCD translation when writing.

**T** Odd parity and no conversion. BCD to EBCDIC translation when reading, and EBCDIC to BCD translation when writing.

The TRTCH and KEYLEN parameters are mutually exclusive.

**UCOUNT(***count***)|PARALLEL**
Shows device allocation.

**UCOUNT(***count***)**
This allocates the maximum number of devices, where count is a value from 1-59.

If you do not use VOLUME and PRIVATE parameters and MAXVOL exceeds UCOUNT, the system removes no volumes when the mounted volumes have been used, causing abnormal termination of your job. If you use PRIVATE, the system removes one of the volumes and mounts another volume in its place to continue processing.

**PARALLEL**
Mounts one device for each volume given on the VOLUME parameter or in the catalog.

**UNIT(***type***)**
Defines the unit type to which a file or data set is to be allocated. You can list an installation-defined group name, a generic device type, or a specific device address. If you do not supply volume information (the system retrieves volume and unit information from a catalog), the unit type that is coded overrides the unit type from the catalog. This condition exists only if the coded type and class are the same as the cataloged type and class.

For VSAM data sets, use the AFF subparameter carefully. If the cluster components and the data and its index reside on unlike devices, the results of UNIT=AFF are unpredictable.

When you allocate a new SMS-managed data set, the system ignores the UNIT parameter. The system determines the UNIT and VOLUME from the storage class associated with the data set. Use UNIT only if you want to allocate a non-SMS-managed data set to a specific unit type.

If the storage administrator has set up a default unit under SMS regardless of whether the data set is SMS-managed, you do not have to use UNIT. If you do not, the system determines the default UNIT for both SMS-managed and non-SMS-managed data sets.

**VOLUME(***serial-list***)**
This is the serial number of an eligible direct access volume on which a new data set is to reside or on which an old data set is located. If you use VOLUME for an old data set, the data set must be on the specified volume for allocation to take place. If you do not include VOLUME, the system allocates new data sets to any eligible direct access volume. The UNIT

information in your procedure entry in the user attribute data set (UADS) determines eligibility. You can use up to 255 volume serial numbers.

For VSAM data sets you must use this subparameter carefully. See the section that discusses DD parameters to avoid when processing VSAM data sets in *z/OS MVS JCL User's Guide* before using the VOLUME subparameters REF, volume-sequence-number, or volume-count.

When you allocate new SMS-managed data sets, you can let the ACS routines select the volume for you. The ACS routines assign your data set to a storage class containing attributes such as VOLUME and UNIT. You can allocate your data set to a *specific* volume only if your storage administrator has stated GUARANTEED SPACE=YES in the storage class assigned to your data set. The volume serial numbers you provide might then override the volume serial numbers used by SMS. If space is not available on the given volume, however, your request is not successful.

**Abbreviation:** VOL

**VSEQ(***vol-seq-no.***)**
This locates which volume (1-255) of a multivolume begins data set processing. This parameter corresponds to the volume sequence number on the VOLUME parameter in JCL. Use VSEQ only when the data set is a cataloged data set.

## ALLOCATE Examples

The following scenarios use the ALLOCATE command to perform various functions:

### Allocate a Data Set Using SMS Class Specifications: Example 1

In this example, the ALLOCATE command is used to allocate a new data set. By providing the SMS data class, management class, and storage class, you can take advantage of the attributes assigned by your storage administrator through the ACS routines.

Although this example includes DYNAMNBR, it is not required in this example. Because this example contains two DD statements, you can do up to two allocations. DYNAMNBR is required only when the number of allocations exceeds the number of DD statements. This example sets DYNAMNBR to 1. This allows up to three allocations for each DD statement (2) plus DYNAMNBR (1).

```
//ALLOC    JOB  ...
           EC PGM=IDCAMS,DYNAMNBR=1
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALLOC -
           DSNAME(ALX.ALLOCATE.EXAMP1) -
           NEW CATALOG -
           DATACLAS(STANDARD) -
           STORCLAS(FAST) -
           MGMTCLAS(VSAM)
/*
```

Because the system syntax checks and ignores SMS classes when SMS is inactive, and because no overriding attributes are given, this example works only if SMS is active. The parameters are:

- DSNAME states that the name of the data set being allocated is ALX.ALLOCATE.EXAMP1.
- NEW creates a data set.

- CATALOG retains the data set by the system in the catalog after step termination. This is mandatory for SMS-managed data sets.
- DATACLAS gives an installation-defined name of a data class to be assigned to this new data set. The data set assumes the RECORG or RECFM, LRECL, KEYLEN, KEYOFF, AVGREC, SPACE, EXPDT or RETPD, VOLUME, CISIZE, FREESPACE and SHAREOPTIONS parameters assigned to this data class by the ACS routines. This parameter is optional. If it is not used, the data set assumes the default data class assigned by the ACS routines.
- STORCLAS gives an installation-defined name of an SMS storage class to be assigned to this new data set. This storage class and the ACS routines are used to determine the volume. This parameter is optional and, if not given, the data set assumes the default storage class assigned by the ACS routines.
- MGMTCLAS is the installation-defined name of an SMS management class to be assigned to this new data set. The data set assumes the migration and backup criteria assigned to this management class by the ACS routines. This parameter is optional and, if not given, the data set assumes the default management class assigned by the ACS routines.

## Allocate a VSAM Data Set Using SMS Class Specifications: Example 2

This example uses the ALLOCATE command to allocate a new data set. Data class is not assigned, and attributes assigned through the default data class are overridden by explicitly specified parameters. By providing the SMS management class and storage class, you can take advantage of attributes already assigned through the ACS routines.

```
//ALLOC    JOB  ...
//STEP1    EXEC PGM=IDCAMS,DYNAMNBR=1
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALLOC -
           DSNAME(M166575.ALLOC.EXAMPLE) -
           NEW CATALOG -
           SPACE(10,2) -
           AVBLOCK(80) -
           AVGREC(K) -
           LRECL(80) -
           RECORG(ES) -
           STORCLAS(FAST) -
           MGMTCLAS(VSAM)
/*
```

The parameters are:

- DSNAME states that the name of the data set being allocated is M166575.ALLOC.EXAMPLE.
- NEW creates the data set.
- CATALOG retains the data set by the system in the catalog after step termination. This is mandatory for SMS-managed data sets.
- The SPACE parameter determines the amount of space to be allocated to the new data set.
  - The first amount (10) is the primary allocation. The second amount (2) is the secondary allocation.
  - Using AVGREC(K) determines that the amounts defined in the SPACE parameter represent kilobytes (K) of records. In this example, the primary allocation is 10K or 10240 records and the secondary allocation is 2K or 2048 records.

  - To determine the space allocation in bytes, multiply the number of records by 80, the record length in LRECL(80). The primary allocation is 819200 bytes. The secondary allocation is 163840 bytes.
- AVBLOCK is the average block length. This example uses an average block length of 80 bytes.
- AVGREC determines whether the quantity in the SPACE parameter represents units, thousands, or millions of records. “K” indicates that the primary and secondary space quantities are to be multiplied by 1024 (1 KB).
- LRECL says the logical record length in the data set is 80 bytes.
- RECORG shows entry-sequenced records in the new VSAM data set.
- STORCLAS gives an installation-defined name of an SMS storage class to be assigned to this new data set. This storage class and the ACS routines are used to determine the volume. This parameter is optional. If it is not used, the data set assumes the default storage class assigned by the ACS routines.
- MGMTCLAS shows an installation-defined name of an SMS management class to be assigned to this new data set. The data set assumes the migration and backup criteria assigned to this management class by the ACS routines. This parameter is optional and, if not given, the data set assumes the default management class assigned by the ACS routines.

## Allocate a New Data Set: Example 3

This example shows the ALLOCATE command being used to allocate a new data set XMP.ALLOCATE.EXAMP3.

```
//ALLOC    JOB  ...
//STEP1    EXEC PGM=IDCAMS,DYNAMNBR=1
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALLOC -
           DSNAME(XMP.ALLOCATE.EXAMP3) -
           NEW CATALOG -
           SPACE(10,5) TRACKS -
           BLKSIZE(1000) -
           LRECL(100) -
           DSORG(PS) -
           UNIT(3380) -
           VOL(338002) -
           RECFM(F,B)
/*
```

The parameters are:

- DSNAME states that the name of the data set to be allocated is XMP.ALLOCATE.EXAMP3.
- NEW creates the data set.
- CATALOG retains the data set in the catalog after step termination.
- SPACE allocates the amount of space to the new data set. In this example, TRACKS is also used so the primary space is 10 tracks with an increment of 5 tracks.
- BLKSIZE requires that the data set control block (DCB) block size is 1000.
- LRECL sets the length of a logical record in the data set to 100.
- DSORG makes the data set physical sequential (PS).
- UNIT and VOL indicate that the data set is to reside on 3380 volume 338002.
- RECFM shows fixed block records in the data set.

## Allocate a non-VSAM Data Set: Example 4

This example shows the ALLOCATE command being used to allocate a non-VSAM data set. ALLOCATE, unlike DEFINE NONVSAM, lets you give the SMS classes for a non-VSAM data set.

```
//ALLOC    JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSABEND DD   SYSOUT=A
 /SYSIN    DD   *
     ALLOC -
           DSNAME(NONVSAM.EXAMPLE) -
           NEW -
           DATACLAS(PS000000) -
           MGMTCLAS(S1P01M01) -
           STORCLAS(S1P01S01)
/*
```

The parameters are:

- DSNAME specifies that the name of the data set to be allocated is NONVSAM.EXAMPLE.
- NEW creates the data set does.is
- DATACLAS assigns an installation-defined name (PS000000) of a data class to this new data set. This parameter is optional and, if not used, the data set assumes the default data class assigned by the ACS routines.
- MGMTCLAS assigns an installation-defined name (S1P01M01) of a management class to this new data set. The data set assumes the migration and backup criteria assigned to this management class by the ACS routines. This parameter is optional and, if not used, the data set assumes the default management class assigned by the ACS routines.
- STORCLAS assigns an installation-defined name (S1P01S01) of a storage class to this new data set. This storage class and the ACS routines determine the volume. This parameter is optional and, if not used, the data set assumes the default storage class assigned by the ACS routines.

## Allocate a Partitioned Data Set Extended: Example 5

This example shows the ALLOCATE command being used with the DSNTYPE keyword to allocate a PDSE.

```
//ALLOC    EXEC  PGM=IDCAMS,DYNAMNBR=1
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
           ALLOC -
              DSNAME(XMP.ALLOCATE.EXAMPLE1) -
              NEW -
              STORCLAS(SC06) -
              MGMTCLAS(MC06) -
              DSNTYPE(LIBRARY)
/*
```

The parameters are:

- DSNAME specifies that the name of the data set to be allocated is XMP.ALLOCATE.EXAMPLE1.
- NEW creates the data set.
- STORCLAS uses the SC06 storage class definition for this data set.
- MGMTCLAS uses the SC06 management class definition for this data set.

- DSNTYPE(LIBRARY) indicates that the object being allocated is an SMS-managed PDSE.

# Chapter 5. ALTER

The ALTER command modifies the attributes of defined data sets and catalogs.

The syntax of the ALTER command is:

| | |
|---|---|
| **ALTER** | *entryname*<br>**[ACCOUNT(***account–info***)]**<br>**[ADDVOLUMES(***volser***[** *volser***...])]**<br>**[BUFFERSPACE(***size***)]**<br>**[BUFND(***number***)]**<br>**[BUFNI(***number***)]**<br>**[BWO(TYPECICS\|TYPEIMS\|NO)]**<br>**[CCSID(***value***)]**<br>**[CODE(***code***)]**<br>**[ECSHARING\|NOECSHARING]**<br>**[EMPTY\|NOEMPTY]**<br>**[ERASE\|NOERASE]**<br>**[EXCEPTIONEXIT(***entrypoint***)]**<br>**[FILE(***ddname***)]**<br>**[FILEDATA(TEXT\|BINARY)]**<br>**[FREESPACE(***CI-percent***[** *CA-percent***])]**<br>**[FRLOG(***NONE***[** *REDO***])]**<br>**[INHIBIT\|UNINHIBIT]**<br>**[KEYS(***length* *offset***)]**<br>**[LIMIT(***limit***)]**<br>**[LOCK\|UNLOCK]**<br>**[LOG(NONE\|UNDO\|ALL)]**<br>**[LOGSTREAMID(***logstream***)]**<br>**[MANAGEMENTCLASS(***class***)]**<br>**[NEWNAME(***newname***)]**<br>**[NULLIFY(**<br>**[AUTHORIZATION(MODULE\|STRING)]**<br>**[BWO]**<br>**[CODE]**<br>**[EXCEPTIONEXIT]**<br>**[LOG]**<br>**[LOGSTREAMID]**<br>**[OWNER]**<br>**[RETENTION]**<br>**[OWNER(***ownerid***)]**<br>**[RECORDSIZE(***average* *maximum***)]**<br>**[REMOVEVOLUMES(***volser***[** *volser***...])]**<br>**[REUSE\|NOREUSE]**<br>**[ROLLIN]**<br>**[SCRATCH\|NOSCRATCH]**<br>**[SHAREOPTIONS(***crossregion***[** *crosssystem***])]**<br>**[STORAGECLASS(***class***)]**<br>**[STRNO(***number***)]**<br>**[TO(***date***)\|FOR(***days***)]**<br>**[TYPE(LINEAR)]**<br>**[UNIQUEKEY\|NONUNIQUEKEY]**<br>**[UPDATE\|NOUPDATE]**<br>**[UPGRADE\|NOUPGRADE]** |

© Copyright IBM Corp. 1973, 2002

| | |
|---|---|
| | **[WRITECHECK\|NOWRITECHECK]**<br>**[CATALOG(***catname***)]** |

## Entry Types That Can Be Altered

An “X” in Figure 1 on page 55 indicates that you can alter the value or attribute for the type of catalog entry that is shown. Some attributes only apply to either the data or the index component of a cluster or alternate index entry. You can use some attributes only for the data or index component of a cluster or alternate index entry; you must then identify the entryname of the component. Use the LISTCAT command to determine the names generated for the object’s components.

You can identify a group of entries with a generic name. Entrynames that match the supplied qualifiers are altered if they have the information that is used with the ALTER command.

You cannot alter alias entries or a master catalog’s self-describing entries, nor can you change a fixed-length relative record data set to a variable-length relative record data set, or the reverse. You cannot change a linear data set (LDS) to any other VSAM data set format. Any attempt to alter a data set defined with a device type named by the user (for example, SYSDA) is unsuccessful.

When the data set characteristics being altered are for a compressed data set, the maximum record length of the control interval size is less than if compression is not done.

| Attributes that can be altered | Type of Catalog Entry | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ALTERNATE INDEX | AIX DATA | AIX INDEX | CLUSTER | CLUSTER DATA | CLUSTER INDEX | PAGESPACE | PATH | USERCAT DATA | USERCAT INDEX | NON VSAM | GDG |
| ACCOUNT | | | | | X | | | | | | X | |
| ADDVOLUME | | X | X | | X | X | | | | | X | |
| BUFFERSPACE | | X | | | X | | | | X | | | |
| BUFND | | | | | | | | | X | | | |
| BUFNI | | | | | | | | | | X | | |
| BWO | | | | X | | | | | | | | |
| CCSID | | | | X | X | | | | | | X | |
| ECSHARING | | | | X[1] | | | | | | | | |
| EMPTY | | | | | | | | | | | | X |
| ERASE | | X | | | X | | | | | | | |
| EXCEPTIONEXIT | | X | X | | X | X | | | | | | |
| FILEDATA | | | | X | X | | | | | | X | |
| FOR | X | | | X | | | | X | X | | X | X |
| FREESPACE | | X | | | X | | | | X | | | |
| INHIBIT | | X | X | | X | X | | | | | | |
| KEYS | X | X | | X | X | | | | | | | |
| LIMIT | | | | | | | | | | | | X |
| LOCK | | | | X[1] | | | | | | | | |
| LOG | | | | X | | | | | | | | |
| LOGSTREAMID | | | | X | | | | | | | | |
| MANAGEMENTCLASS | | | | X | | | | | X | | X | |
| MASTERPW | X | X | X | X | X | X | X | X | X | | | |
| NOECSHARING | | | | X[1] | | | | | | | | |
| NEWNAME | X | X | X | X | X | X | X | X | | | X | |
| NOEMPTY | | | | | | | | | | | | X |
| NOERASE | | X | | | X | | | | | | | |
| NOUNIQUEKEY | | X | | | | | | | | | | |
| NOREUSE | | | | X | | | | | | | | |
| NOSCRATCH | | | | | | | | | | | | X |
| NOUPDATE | | | | | | | | X | | | | |
| NOUPGRADE | X | | | | | | | | | | | |
| NOWRITECHECK | | X | X | | X | X | | | X | X | | |
| NULLIFY | X | X | X | X | X | X | X | X | X | | X | X |
| AUTHORIZATION | X | X | X | X | X | X | X | X | X | | | |
| BWO | | | | X | X | | | | | | | |
| CODE | X | X | X | X | X | X | X | X | X | | | |
| EXCEPTIONEXIT | | X | X | X | X | X | | | | | | |
| LOG | | | | X | X | | | | | | | |
| LOGSTREAMID | | | | X | X | | | | | | | |
| OWNER | X | X | X | X | X | X | X | X | X | | X | X |
| RETENTION | X | | | X | | | X | X | X | | X | X |

*Figure 1. ALTER Attributes That Can be Altered and Types of Catalog Entry (Part 1 of 2)*

| Attributes that can be altered | Type of Catalog Entry | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ALTERNATE INDEX | AIX DATA | AIX INDEX | CLUSTER | CLUSTER DATA | CLUSTER INDEX | PAGESPACE | PATH | USERCAT DATA | USERCAT INDEX | NON VSAM | GDG |
| OAM [2] | | | | | | | | | | | | |
| OWNER | X | X | X | X | X | X | X | X | X | | X | X |
| READPW | X | X | X | X | X | X | X | X | X | | | |
| RECORDSIZE | X | X | | X | X | | | | | | | |
| REMOVEVOLUMES | | X | X | | X | X | | | | | | |
| REUSE | | | | X | | | | | | | | |
| ROLLIN | | | | | | | | | | | X [3] | |
| SCRATCH | | | | | | | | | | | | X |
| SHAREOPTIONS | | X | X | | X | X | | | X [4] | | | |
| STAGE | | X | X | | X | X | | | | | | |
| STORAGECLASS | | | | X | | | | | X | | X | |
| STRNO | | | | | | | | | X | | | |
| TO | X | | | X | | | X | X | X | | X | X |
| TYPE | | | | X | | | | | | | | |
| UNIHIBIT | | X | X | | X | X | | | | | | |
| UNIQUEKEY | | X | | | | | | | | | | |
| UNLOCK | | | | X [1] | | | | | | | | |
| UPDATE | | | | | | | | X | | | | |
| UNDATEPW | X | X | X | X | X | X | X | X | X | | | |
| UPGRADE | X | | | | | | | | | | | |
| WRITECHECK | | X | X | | X | X | | | X | X | | |

1. LOCK and UNLOCK and ECSHARING and NOECSHARING can only be specified for an integrated catalog facility catalog.
2. Alter commands cannot be run against OAM data sets.
3. The data set must be a generation data set in either deferred rolled-in or rolled-off status.
4. When SHAREOPTIONS is specified or an integrated catalog facility catalog, the data level share option will be propagated to the index level as well.

*Figure 1. ALTER Attributes That Can be Altered and Types of Catalog Entry (Part 2 of 2)*

## ALTER Parameters

The ALTER command takes the following required and optional parameters.

## Required Parameters

*entryname*

This names the entry to be altered.

When attributes of a catalog are altered, entryname must include either the data or index components. Giving the catalog name alters attributes defined at the cluster level only. The catalog name is also the data component name.

The restricted prefix SYS1.VVDS.V or its generic form SYS1.VVDS.* or SYS1.*.V is not allowed as an entryname for the ALTER command.

If you are renaming a member of a non-VSAM partitioned data set, the entryname must given as: pdsname(membername).

See the NEWNAME parameter for information on renaming SMS-managed data sets.

You identify a general data stream (GDS) with its GDG name followed by the generation and version numbers of the data set (GDGname.GxxxxVyy). You cannot use relative generation numbers (that is, GDGname(+1)) with the entryname

## Optional Parameters

**ACCOUNT(***account–info***)**
Account is supported only for SMS-managed VSAM data sets or non-VSAM data sets.

*account–info*
Use this to change accounting information and user data for the data set. It *must* be between 1 and 32 bytes; otherwise you will receive an error message.

**Abbreviation:** ACCT

**ADDVOLUMES(***volser* **[** *volser***])**
This provides the volumes to be added to the list of candidate volumes. You can use ALTER ADDVOLUMES to add candidate volumes to non-managed VSAM data sets and SMS-managed VSAM, non-VSAM, and general data stream (GDS) data sets. Only nonspecific volumes can be added to SMS-managed, non-VSAM data sets and GDS data sets. If an ALTER ADDVOLUMES is done to a data set already opened and allocated, the data set must be closed, unallocated, reallocated, and reopened before VSAM can extend onto the newly-added candidate volume. Adding a nonexistent volume to the list can result in an error when the data set is extended. Ensure that the volume exists and is on-line before attempting to extend the data set.

**Restriction:** This does not work with NONSMS NONVSAM

SMS might not use candidate volumes for which you request specific volsers with the ADDVOLUMES parameter. Sometimes a user-specified volser for an SMS-managed data set results in an error. To avoid candidate-volume problems with SMS, you can have SMS choose the volser used for a candidate volume. To do this, you can code an * for each volser that you request with the ADDVOLUMES parameter. If, however, you request both specified and unspecified volsers in the same command, you must enter the specified volsers first in the command syntax. The system does not allocate space on candidate volumes until VSAM extends to the candidate volume. This includes SMS-managed data sets with guaranteed space.

**Abbreviation:** AVOL

**BUFFERSPACE(***size***)**
Provides the amount of space for buffers. The size you specify for the buffer space helps VSAM determine the size. We recommend that the size you give is equal to, or greater than the amount specified in the original definition. If the amount is less, VSAM attempts to get enough space to contain two data component control intervals and, if the data is key-sequenced, one index

component control interval. You can specify BUFFERSPACE only for a catalog, or for the data component of a cluster or alternate index. If you use BUFFERSPACE for a catalog, then you must usespecify the CATALOG parameter.

The BUFFERSPACE parameter is ignored for record-level sharing (RLS) access.

*size*
is the amount of space for buffers. This helps VSAM determine the size of the data component's and index component's control interval.

*Size* can be entered in decimal (n), hexadecimal (X'n'), or binary (B'n') form, but must not exceed 16,776,704. The specified size should not be less than the space needed to contain two data component control intervals and, if the data is key-sequenced, to contain one index control interval. If the given size is less than what VSAM requires, it gets it when the data set is opened.

**Abbreviations** BUFSP or BUFSPC

**BUFND(*number*)**
Gives the number of I/O buffers VSAM is to use for transmitting data between virtual and auxiliary storage. The size of the buffer area is the size of the data component control interval. Use this parameter only to alter the data component of an integrated catalog facility catalog.

The BUFND parameter is ignored for RLS access.

*number*
Is the number of data buffers you can use. The minimum number is 3, and the maximum is 255.

**Abbreviation:** BFND

**BUFNI(*number*)**
Is the number of I/O buffers VSAM uses for transmitting the contents of index entries between virtual and auxiliary storage for keyed access. The size of the buffer area is the size of the index control intervals. Use this parameter only to alter the index component of the integrated catalog facility catalog.

The BUFNI parameter is ignored for RLS access.

*number*
Is the number of index buffers you can use. The minimum number is 2, and the maximum is 255.

**Abbreviation:** BFNI

**BWO(TYPECICS|TYPEIMS|NO)**
Use this parameter if backup-while-open (BWO) is allowed for the VSAM sphere. BWO applies only to SMS data sets and cannot be used with TYPE(LINEAR). If you specify BWO, access from DFSMS/MVS 1.2 or lower is denied.

**TYPECICS**
Use TYPECICS to specify BWO in a CICS® environment. For RLS processing, this activates BWO processing for CICS. For non-RLS processing, CICS determines whether to use this specification or the specification in the CICS FCT.

**Exception:** If CICS determines that it will use the specification in the CICS FCT, the specification might override the TYPECICS or NO parameters.

**Abbreviation:** TYPEC

**TYPEIMS**
If you want to use BWO processing in an Internet Management Specification (IMS) environment, use the TYPEIMS parameter. You can use this capability only with DFSMS 1.3 or higher level DFSMS systems. If you attempt to open a cluster that has the TYPEIMS specification of DFSMS 1.2 or lower, the open will not be successful.

**Abbreviation:** TYPEI

**NO**
Use this when BWO does not apply to the cluster.

**Exception:** If CICS determines that it will use definitions in the CICS FCT, the TYPECICS or NO parameters might be overwritten.

**CATALOG(*catname*)**
Specifies the catalog containing the entry to be altered.

To assign catalog names for SMS-managed data sets, you must have access to the RACF STGADMIN.IGG.DIRCAT facility class. See “Storage Management Subsystem (SMS) Considerations” on page 2 for more information.

*catname*
Is the name of the catalog that contains the entry.

**Abbreviation:** CAT

**CCSID(*value*)**
Is the Coded Character Set Identifier attribute; it identifies:
- Encoding scheme identifier
- Character set identifier or identifiers
- Code page identifier or identifiers
- Additional coding required to uniquely identify the coded graphic used

You can use Coded Character Set Identifier (CCSID) only for system-managed data sets. If the CCSID parameter is not in the catalog at the time ALTER is called, it is created.

The *value* for CCSID can be specified in decimal (n), hexadecimal (X’n’), or binary (B’n’). The acceptable range of values is 0 (X’0’) to 65535 (X’FFFF’).

**ECSHARING|NOECSHARING**
Indicates whether sharing this catalog can be performed via the coupling facility.

**ECSHARING**
Enhanced Catalog Sharing (ECS) is allowed. ECS is a catalog sharing method that makes use of a coupling facility to increase the performance of shared catalog requests. Please read about ECS in *z/OS DFSMS: Managing Catalogs* before enabling ECS for a catalog.

**Abbreviation:** ECSHR

**NOECSHARING**
Enhanced Catalog Sharing (ECS) is not allowed. This is the default. Catalog sharing is performed, but the ECS sharing method is not be used.

**Abbreviation:** NOECSHR

**EMPTY|NOEMPTY**
Specifies what is to happen when the maximum number of generation data sets has been cataloged. If the GDG is full (the LIMIT is reached), this attribute determines whether all GDSs or just the oldest GDSs are processed.

For an SMS-managed generation data set, if the NOSCRATCH attribute is used, the GDS is uncataloged from its GDG base and is recataloged outside its GDG base as an SMS non-VSAM entry with the rolled-off status.

**EMPTY**
Specifies that, when the maximum number of generation data sets is exceeded, all the generation data sets are uncataloged or deleted.

**Abbreviation:** EMP

**NOEMPTY**
Used when the maximum number of generation data sets is exceeded, only the oldest generation data set is uncataloged or deleted.

**Abbreviation:** NEMP

**ERASE|NOERASE**
Indicates whether to erase the component when its entry in the catalog is deleted.

**ERASE**
Overwrites the component with binary zeros when its catalog entry is deleted. If the cluster or alternate index is protected by a RACF generic or discrete profile, use RACF commands to assign an ERASE attribute as part of this profile so that the data component is automatically erased upon deletion.

**Abbreviation:** ERAS

**NOERASE**
Specifies the component is not to be overwritten with binary zeros when its catalog entry is deleted. NOERASE resets only the indicator in the catalog entry that was created from a prior DEFINE or ALTER command. If the cluster or alternate index is protected by a RACF generic or discrete profile that specifies the ERASE attribute, it is erased upon deletion. Only RACF commands can be used to alter the ERASE attribute in a profile.

**Abbreviation:** NERAS

**EXCEPTIONEXIT(*entrypoint*)**
Is the name of the user-written routine that receives control if an exception (usually an I/O error) occurs while the entry's object is being processed. An exception is any condition that causes a SYNAD exit. The object's exception exit routine is processed first, then the user's SYNAD exit routine receives control.

**Abbreviation:** EEXT

**FILE(*ddname*)**
Specifies one of the following:

- The name of a DD statement that describes the volume that contains the data set to be altered.
- The name of a DD statement that identifies the volume of an entry that will be renamed. The entry must be a non-VSAM data set or the data or index component of a cluster, alternate index, or page space.

- The name of a DD statement that describes a partitioned data set when a member is to be renamed.

If you identify multiple volumes of different device types with FILE, use concatenated DD statements. If you specify ADDVOLUMES or REMOVEVOLUMES, the volume being added or removed must be identified. If FILE is not specified, an attempt is made to dynamically allocate the object's data set. Therefore, the object's volume must be mounted as permanently resident or reserved.

While the FILE parameter can preallocate a volume where the data set resides, it does not direct the ALTER request to the data set to be altered. Instead, a catalog search is done to locate the data set to be altered.

**FILEDATA(TEXT|BINARY)**
Use one of the following:

**TEXT**
Specifies that the data in the file is text. If the file is read/written across the network, the data in this file is EBCDIC on MVS and ASCII on the workstation.

**BINARY**
Specifies that data is to be processed as is.

**FREESPACE(***CI-percent***[** *CA-percent***]) Abbreviation: FSPC**
Indicates the percent of free space left after any allocation. CI-percent is a percentage of the amount of space to be preserved for adding new records and updating existing records, with an increase in the length of the record. Since a CI is split when it becomes full, the CA might also need to be split when it is filled by CIs created by a CI split. The amounts, as percentages, must be equal to, or less than, 100. If you use 100% of free space, one record is placed in each control interval and one control interval is placed in each control area (CA).

Use this parameter to alter the data component of a cluster, alternate index, or catalog.

If the FREESPACE is altered after the data set has been loaded, and sequential insert processing is used, the allocation of free space is not honored.

**FRLOG(NONE|REDO)**
Specifies if VSAM batch logging can be performed for your VSAM data set. VSAM batch logging is available with CICS VSAM Recovery V3R1.

**NONE**
Disables the VSAM batch logging function for your VSAM data set. Changes made by applications are not written to the MVS log stream indicated on the LOGSTREAMID parameter.

**REDO**
Enables the VSAM batch logging function for your VSAM data set. Changes made by applications are written to the MVS log stream indicated on the LOGSTREAMID parameter. If you specify FRLOG(REDO), you must also specify LOGSTREAMID for that data set, unless it is already defined.

**Restrictions:**

1. Use the FRLOG parameter only if you want to enable (REDO) or disable (NONE) VSAM batch logging. Do not use the FRLOG parameter for data sets that are not intended for use with VSAM batch logging.

2. If FRLOG is specified, the data set:
   - Must be SMS-managed
   - Cannot be LINEAR or a temporary data set.

**INHIBIT|UNINHIBIT**
Specifies whether the entry being altered can be accessed for any operation or only for read operations.

**INHIBIT Abbreviation: INH**
Used when the entry being altered is to be read only.

**UNINHIBIT**
Indicates that the read-only restriction set by a previous ALTER or EXPORT command is to be removed.

**Abbreviation:** UNINH

**KEYS(*length offset*)**
Specifies the length and offset of the object's key. If the altered entry defines an alternate index, offset applies to the alternate key in the data records in the base cluster.

**Restrictions:** Use KEYS if all the following are true:

- The object whose entry is being altered is an alternate index, a path, a key-sequenced cluster, or a data component of a key-sequenced cluster or alternate index.
- The object whose entry is being altered contains no data records.
- The values for KEYS in the object's catalog entry are default values. For default values, see the DEFINE command for the object.
- The new values for KEYS do not conflict with the control interval size specified when the object was defined.
- The key fits within the record whose length is specified by the RECORDSIZE parameter.
- The key fits in the first record segment of a spanned record.

*length offset*
Is the length of the key (between 1 and 255), in bytes, and its displacement from the beginning of the data record, in bytes.

If the values for KEYS in the object's catalog entry are not default values and ALTER KEYS specifies those same values, processing continues for any other parameters specified on the command, and no error message is issued.

**LOG(NONE|UNDO|ALL)**
Establishes whether the sphere to be accessed with VSAM record-level sharing (RLS) is recoverable or non-recoverable. It also indicates whether a recovery log is available for the sphere. LOG applies to all components in the VSAM sphere.

**NONE**
Indicates that neither an external-backout nor a forward-recovery capability is available for the spheres accessed in RLS mode. If you use this, RLS considers the sphere to be non-recoverable.

**UNDO**
Specifies that changes to the sphere accessed in RLS mode can be backed out using an external log. RLS considers the sphere recoverable when you use LOG(UNDO).

**ALL**
Specifies that changes to the sphere accessed in RLS mode can be backed out and forward recovered using an external log. RLS considers the sphere recoverable when you use LOG(ALL). When you specify LOG(ALL), you must also specify the LOGSTREAMID parameter, unless it is already defined.

VSAM RLS allows concurrent read/update sharing for non-recoverable spheres through commit (CICS) and non-commit protocol applications. For a recoverable sphere, an application that does not have an external log (BATCH) cannot open the sphere through RLS access.

**Restriction:** LOG cannot be used with LINEAR.

**LOGSTREAMID(*logstream*)**
Changes or adds the name of the forward recovery log stream. It applies to all components in the VSAM sphere.

*logstream*
Is the name of the forward recovery log stream. This can be a fully qualified name up to 26 characters, including separators. This parameter is required if you have specified LOG(ALL).

For information about defining log streams for CICS use, see the *CICS System Definition Guide*.

**Abbreviation:** LSID

**Restriction:** LOGSTREAMID cannot be used with LINEAR.

**LIMIT(*limit*)**
Used to modify the maximum number (between 1 and 255) of active generation data sets that might be associated with a generation data group base.

*limit*
If the limit is less than the current number of active generations, the oldest generations are rolled off until the new limit is satisfied. Any generation data sets that are rolled off by this command are listed showing their new status (recataloged, uncataloged, or deleted). For more information about limit processing of a GDS, see *z/OS DFSMS: Managing Catalogs*.

If the limit is greater than the current number of active generations, it does not cause the roll-in of existing rolled off GDSs. For this function, see the ROLLIN parameter.

**LOCK|UNLOCK**
Controls the setting of the catalog lock attribute, and therefore checks access to a catalog. Use LOCK or UNLOCK when the entry name identifies an integrated catalog facility catalog. If the LOCK|UNLOCK parameter is not specified, the status of the integrated catalog facility catalog lock attribute is not changed. Before you lock a catalog, review the information on locking catalogs in *z/OS DFSMS: Managing Catalogs*.

**LOCK**
Is used when the catalog identified by entryname is to be locked. Locking a catalog makes it inaccessible to all users without read authority to RACF facility class profile IGG.CATLOCK (including users sharing the catalog on other systems).

For protected catalogs, locking an unlocked catalog requires ALTER authority for the catalog being locked, and read authority to RACF facility

profile IGG.CATLOCK. For unprotected catalogs, locking an unlocked catalog requires read authority to RACF facility class profile IGG.CATLOCK.

**UNLOCK**
Specifies that the catalog identified by entryname is to be unlocked. For RACF and nonprotected catalogs, unlocking a locked catalog requires read authority to RACF facility class profile IGG.CATLOCK.

**MANAGEMENTCLASS(***class***)**
*For SMS-managed data sets:* Gives the name, 1-to-8 characters, of the management class for a data set. Your storage administrator defines the names of the management classes you can include. If MANAGEMENTCLASS is used for a non-SMS-managed data set, or if SMS is inactive, the ALTER command is unsuccessful.

When the storage or management class is altered for a DFSMShsm migrated data set, ALTER will not recall the data set to make the change, provided no other parameters are specified.

You must have RACF access authority to alter the management class.

**Abbreviation:** MGMTCLAS

**NEWNAME(***newname***)**
Indicates that the entry to be altered is to be given a new name.

When you rename an SMS-managed data set residing on DASD, the MGMTCLAS ACS routine is called and lets you reassign a new management class.

You can use ALTER NEWNAME to rename SMS-managed generation data sets (GDS). Table 4 on page 65 shows how NEWNAME resolves renaming a GDS under different conditions. You can successfully rename the following:
- An SMS-managed GDS to an SMS-managed non-VSAM data set
- An SMS-managed non-VSAM data set to an SMS-managed GDS
- An SMS-managed GDS to another SMS-managed GDS

**Restriction:** You cannot ALTER the data portion of a page space data set to a new name. Also, catalog names and catalog component names cannot be renamed.

You might not be able to rename a data set if you are changing the high-level qualifiers of the data set's name and those qualifiers are an alias name of a catalog. (The number of high-level qualifiers used to form an alias can be one to four, depending on the multilevel alias search level used at your installation.)

If you are changing a high-level qualifier, NEWNAME acts differently depending on whether the data set being renamed is SMS-managed or non-SMS-managed, and whether the data set has aliases or not. Table 4 on page 65 shows how NEWNAME resolves under different conditions.

*Table 4. How NEWNAME Resolves When Change of Catalog is Required*

| Data Set Type | SMS | non-SMS |
|---|---|---|
| VSAM | ALTER unsuccessful—entry not renamed | ALTER successful—entry remains in the source catalog |
| non-VSAM with no aliases | ALTER successful—entry is recataloged in target catalog. | ALTER successful—entry remains in the source catalog |
| non-VSAM with aliases | ALTER unsuccessful—entry not renamed | ALTER successful—entry remains in the source catalog |
| GDS with no aliases | ALTER successful—entry is recataloged in target catalog. | ALTER unsuccessful—entry not renamed |
| GDS with aliases | ALTER unsuccessful—entry not renamed | ALTER unsuccessful—entry not renamed |
| **Note:** The source catalog is the catalog containing the original entry. The target catalog is the catalog in which the new name would normally be cataloged according to a catalog alias search. | | |

If you want to define a data set into a particular catalog, and that catalog is not the one chosen according to the regular search, then you must have authority to RACF STGADMIN.IGG.DIRCAT facility class. For more information on this facility class see *z/OS DFSMSdfp Storage Administration Reference*.

To give an altered entry a new name:

- Unless the data set being renamed is a path, the data set's volume must be mounted because the volume table of contents (VTOC) is modified.

  You can use the FILE parameter to supply a JCL DD statement to allocate the data set. If you do not supply a DD statement, an attempt is made to dynamically allocate the data set. The volume must be mounted as either permanently resident or reserved.

  If another program has access to the data set while this is being done, the program might not be able to access the data set after it is renamed. This can result in an error.
- If you include generic names, you must define both entryname and newname as generic names.
- If you are renaming a member of a non-VSAM partitioned data set, the newname must be specified in the format: pdsname(membername).
- If you are renaming a VSAM data set that is RACF protected, the existing RACF data set profile will be renamed.
- If you are using ALTER NEWNAME, you must have one of these:
  - ALTER authority to the data set or to the catalog
  - ALTER authority to the new name (generic profile) or CREATE authority to the group
- If there is a data set profile for the new data set name prior to the ALTER command, the command ends, and the data set name and protection attributes remain unchanged.

  If the old profile is not found or cannot be altered to the new name, the NEWNAME action is not completed in the catalog, and an error message indicates why the action is not completed.

  If renaming is unsuccessful, it is possible that either the object exists with both the original name and the new name, or that the data set was not closed.

**Abbreviation:** NEWNM

**NULLIFY([AUTHORIZATION(MODULE|STRING)]**

**[BWO][CODE][EXCEPTIONEXIT]**

**[LOG][LOGSTREAMID][OWNER]**

**[RETENTION])**

Specifies that the protection attributes identified by Subparameters of NULLIFY are to be nullified. Attributes are nullified before any respecification of attributes is done.

**Abbreviation:** NULL

**AUTHORIZATION(MODULE|STRING)**

Is used when the user authorization routine or the user authorization record is to be nullified.

**Abbreviation:** AUTH

**MODULE**

Removes the module name from the catalog record, but the module itself is not to be deleted. Both the user authorization routine and the user authorization record (character string) are nullified.

**Abbreviation:** MDLE

**STRING**

Nullifies the authorization record, but the corresponding module is not nullified.

**Abbreviation:** STRG

**BWO**

Use this parameter to remove the BWO specification from the sphere.

**CODE**

Nullifies the code name used for prompting.

**EXCEPTIONEXIT**

Nullifies the entry's exception exit. The module name is removed from the catalog record, but the exception-exit routine itself is not deleted.

**Abbreviation:** EEXT

**LOG**

Nullifies the log parameter.

Access to the RLS sphere is not permitted when the log parameter is nullified.

**LOGSTREAMID**

When you use this, the name of the forward recovery log stream is nullified. NULLIFY(LOGSTREAMID) is not allowed if the data set has a value of LOG(ALL).

**Abbreviation:** LSID

**OWNER**

Nullifies that the owner identification.

**RETENTION**

Nullifies the retention period that was used in a TO or FOR parameter.

**Abbreviation:** RETN

**OWNER(***ownerid***)**
Specifies the owner identification of the entry being altered.

**RECORDSIZE(***average maximum***)**
Specifies new average and maximum lengths for data records contained in the object whose entry is being altered.

If the object whose entry is being altered is a path pointing to the alternate index, the alternate index is altered; if it is a path pointing directly to the base cluster, the base cluster is altered.

If the object whose entry is being altered is an alternate index, the length of the alternate key must be within the limit specified by maximum.

**Restrictions:** RECORDSIZE is used only if all the following are true:

- The object whose entry is being altered is an alternate index, a cluster, a path, or a data component.
- The object whose entry is being altered contains no data records.
- The maximum RECORDSIZE in the object's catalog entry is the default. For defaults, see the DEFINE command for the object.
- If NONUNIQUEKEY is used for an alternate index, the record length to be specified accounts for the increased record size; this results from the multiple prime key pointers in the alternate index data record.
- Use a maximum record length of at least seven bytes less than the control interval size, unless the record is a spanned record.
- Use a record length large enough to contain all prime and alternate keys previously defined.

If RECORDSIZE in the object's catalog entry is not the default, and ALTER RECORDSIZE specifies that same value, processing continues for any other parameters given on the command, and there is no error message.

**Abbreviation:** RECSZ

**REMOVEVOLUMES(***volser***[** *volser***])**
Specifies volumes to be removed from the list of candidate volumes associated with the entry being altered. The name of the data or index component must be specified in the ENTRYNAME parameter. If you are also adding volumes, the volumes to be removed are removed after the new volumes are added to the candidate list. Only nonspecific volumes can be removed from SMS-managed, non-VSAM data sets, and GDS data sets. For information on volume cleanup, refer to "VSAM Volume Cleanup" in *z/OS DFSMS: Managing Catalogs*.

SMS might not use candidate volumes for which you request specific volsers. Some user-specified volsers for an SMS-managed data set can result in an error. To avoid candidate volume problems with SMS, you can request that SMS choose the given volser used for a candidate volume. To do this, you can code an * for each volser that you request. If, however, you request both specified and unspecified volsers in the same command, you must enter the specified volsers first in the command syntax.

To ensure that the operation has completed correctly, the execution of ALTER REMOVEVOLUMES should be followed by a listing of the VTOC on the target volume. If ALTER REMOVEVOLUMES did not scratch any data sets allocated to job steps, it can still complete with return code zero. In the integrated catalog facility environment, both the basic catalog structure (BCS) and the VSAM volume data set (VVDS) might be allocated to another job or TSO user. If so, these entities are not scratched, and any future access method services

commands that depend on ALTER REMOVEVOLUMES completing normally might be unsuccessful. To ensure that the operation has completed correctly, follow the execution of ALTER REMOVEVOLUMES with a listing of the VTOC on the target volume.

**Exceptions:**

1. If a volume to be removed contains data that belongs to the entry being altered, the volume is not removed.
2. Volume cleanup is not supported if the volume is SMS managed.

**REUSE|NOREUSE**
Controls setting the REUSE indicator for VSAM data sets. A data set that requires the REUSE attribute be changed to ″reusable″ cannot be an alternate index nor can it have an associated alternate index. The data set also cannot be a key-sequenced data set (KSDS) with one or more key ranges.

**ROLLIN**
Indicates whether a generation data set is to be rolled-in. The generation data set must be SMS managed and in either a deferred rolled-in or rolled-off state. See “Rolling In A Data Set” in *z/OS DFSMS: Using Data Sets* for more information.

**Abbreviation:** ROL

**SCRATCH|NOSCRATCH**
Specifies whether generation data sets, when they are uncataloged, are to be removed from the VTOC of the volume where they reside.

**SCRATCH**
Removes the data set’s format-1 DSCB from the VTOC so that the data set can no longer be accessed, and, for SMS-managed data sets, the non-VSAM volume record (NVR) is removed from the VVDS.

**Abbreviation:** SCR

**NOSCRATCH**
Indicates that the data set’s format-1 DSCB is not to be removed from the VTOC and, for SMS-managed data sets, the NVR entry remains in the VVDS.

**Abbreviation:** NSCR

**SHAREOPTIONS(***crossregion***[** *crosssystem***])**
Is used when a data or index component of a cluster, alternate index, or the data component of a catalog can be shared among users. However, SMS-managed volumes, and catalogs containing SMS-managed data sets, must not be shared with non-SMS systems. (For a description of data set sharing, see *z/OS DFSMS: Using Data Sets*.)

*crossregion*
Specifies the amount of sharing allowed among regions within the same system or within multiple systems using global resource serialization (GRS). Independent job steps in an operating system, or multiple systems in a GRS ring, can access a VSAM data set concurrently. For a description of GRS, see *z/OS MVS Planning: Global Resource Serialization*. Option 3 is the only one applicable when altering a catalog. To share a data set, each user must code DISP=SHR in the data set’s DD statement. You can use the following options:

**OPT 1** The data set can be shared by any number of users for read processing, or the data set can be accessed by only one user for

read and write processing. VSAM ensures complete data integrity for the data set. This setting does not allow any non-RLS access when the data set is already open for RLS processing. An RLS open will fail with this option if the data set is already open for any processing.

**OPT 2** The data set can be accessed by any number of users for read processing, and it can also be accessed by one user for write processing. It is the user's responsibility to provide read integrity. VSAM ensures write integrity by obtaining exclusive control for a control interval while it is being updated. An RLS open is not allowed while the data set is open for non-RLS output.

If the data set has already been opened for RLS processing, a non-RLS open for input is allowed; a non-RLS open for output fails.[1] If the data set is opened for input in non-RLS mode, an RLS open is allowed.

**OPT 3** The data set can be fully shared by any number of users. The user is responsible for maintaining both read and write integrity for the data the program accesses. This setting does not allow any non-RLS access when the data set is already open for RLS processing. If the data set is opened for input in non-RLS mode, an RLS open is allowed.

This option is the only one applicable to an integrated catalog facility catalog.

**OPT 4** The data set can be fully shared by any number of users. For each request, VSAM refreshes the buffers used for direct processing. This setting does not allow any non-RLS access when the data set is already open for RLS processing. If the data set is opened for input in non-RLS mode, an RLS open is allowed.

As in SHAREOPTIONS 3, each user is responsible for maintaining both read and write integrity for the data the program accesses.

*crosssystem*

Is the amount of sharing allowed among systems. Job steps of two or more operating systems can gain access to the same VSAM data set regardless of the disposition specified in each step's DD statement for the data set. To get exclusive control of the data set's volume, a task in one system issues the RESERVE macro. The level of cross-system sharing allowed by VSAM applies only in a multiple operating system environment.

The cross-system sharing options are ignored by RLS processing. The values are:

**1** Reserved.

**2** Reserved.

**3** Specifies that the data set can be fully shared. With this option, each user is responsible for maintaining both read and write integrity for the data the program accesses. User programs that ignore write integrity guidelines can cause VSAM program checks, uncorrectable data set problems, and other unpredictable results. The RESERVE and DEQ macros are required with this option to maintain data set integrity. (For information on using RESERVE and

1. You must apply APARs OW25251 and OW25252 to allow non-RLS read access to data sets already opened for RLS processing.

DEQ, see *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN* and *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU*.)

**4** Specifies that the data set can be fully shared. For each request, VSAM refreshes the buffers used for direct processing. This option requires that you use the RESERVE and DEQ macros to maintain data integrity while sharing the data set. Improper use of the RESERVE macro can cause problems similar to those described under SHAREOPTIONS 3. (For information on using RESERVE and DEQ, see *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN* and *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU*.)

Output processing is limited to update or add processing that does not change either the high-used relative byte address (RBA) or the RBA of the high key data control interval if DISP=SHR is specified.

**Abbreviation:** SHR

**STORAGECLASS(***class***)**

*For SMS-managed data sets:* Gives the name, 1-to-8 characters, of the storage class. Your storage administrator defines the names of the storage classes you can assign. A storage class is assigned when you specify STORAGECLASS or an installation-written automatic class section (ACS) routine selects a storage class when the data set is created. Use the storage class to provide the storage service level to be used by SMS for storage of the data set. The storage class provides the storage attributes that are specified on the UNIT and VOLUME operand for non-SMS-managed data sets.

When the storage or management class is altered for a DFSMShsm migrated data set, ALTER will not recall the data set to make the change, provided no other parameters are specified.

You must have RACF access authority to alter the storage class.

If STORAGECLASS is used for a non-SMS-managed data set or if SMS is inactive, the ALTER command is unsuccessful.

**Abbreviation:** STORCLAS

**STRNO(***number***)**

Specifies the number of concurrent catalog positioning requests that VSAM should manage. Use this parameter to alter the data component of an integrated catalog facility catalog. The STRNO setting is ignored when the data set is opened for RLS.

*number*

Is the number of concurrent requests VSAM must manage. The minimum number is 2, the maximum is 255.

**TO(***date***)|FOR(***days***)**

Specifies the retention period for the entry being altered.

You cannot use these parameters for the data or index components of clusters or alternate indexes. For catalogs, you must use the data component name. The expiration date in the catalog is updated, and, for SMS-managed data sets, the expiration date in the format-1 DSCB is changed. Enter a LISTCAT command to see the correct expiration date.

The MANAGEMENTCLASS maximum retention period, if specified, limits the retention period specified by this parameter.

**TO(*date*)**
Specifies the date up to which an entry should be kept before it is allowed to be deleted. The date is specified in the form [yy]yyddd, where yyyy is a four-digit year, yy is a two-digit year, and ddd is the three-digit (001 through 366) day of the year. Two-digit years are treated as if “19” is specified as the first two digits of yyyy. The dates (19)99365 and (19)99366 are considered never-expire dates.

For expiration dates of January 1, 2000, and later, you must use the form TO(yyyyddd).

**FOR(*days*)**
Is used to choose the number of days to keep the entry. The maximum number is 9999. If the number is 0 through 9998, the entry is retained for the number of days indicated; if the number is 9999, the entry is retained indefinitely.

**TYPE(LINEAR)**
Specifies that the VSAM data set type entry-sequenced data set (ESDS) is to be changed to linear. The contents of the data set are not modified. Only an ESDS with a CI size of 4096 is eligible to be a linear data set. A linear data set’s type cannot be changed. After you have changed an ESDS set to a linear data set, the data set must remain a linear data set; you cannot change it back into an ESDS.

**LINEAR**
Changes the VSAM data type ESDS to a linear data set (LDS).

**Abbreviation:** LIN

**UNIQUEKEY|NONUNIQUEKEY**
Specifies whether the alternate key value can be found in more than one of the base cluster’s data records.

**UNIQUEKEY**
Makes each alternate key value unique. If the same alternate key value is found in more than one of the base cluster’s data records, an error results.

You can use UNIQUEKEY for an empty alternate index (that is, an alternate index that is defined but not yet built).

**Abbreviation:** UNQK

**NONUNIQUEKEY**
Allows an alternate key value to point to more than one data record in the cluster. NONUNIQUEKEY can be specified for an alternate index at any time.

If the alternate index is empty, you should also consider defining RECORDSIZE to ensure that each alternate index record is large enough to contain more than one data record pointer.

**Abbreviation:** NUNQK

**UPDATE|NOUPDATE**
Specifies whether a base cluster’s alternate index upgrade set is to be allocated when the path’s name is allocated.

The NOUPDATE setting is ignored when the data set is opened for RLS. Alternate indexes in the upgrade set are opened as if UPDATE was specified.

**UPDATE**
Allocates the cluster's alternate index upgrade set when the path's name is allocated with a DD statement.

**Abbreviation:** UPD

**NOUPDATE**
Specifies that the cluster's alternate index upgrade set is not to be allocated but the path's cluster is to be allocated. You can use NOUPDATE to open a path. If the path shares a control block structure that uses UPDATE, this indicates the upgrade set has been allocated and, in this case, the upgrade set can be updated.

**Abbreviation:** NUPD

**UPGRADE|NOUPGRADE**
Shows whether an alternate index is to be upgraded (to reflect the changed data) when its base cluster is modified.

**UPGRADE**
Indicates that the cluster's alternate index is upgraded (to reflect the changed data) when the cluster's records are added to, updated, or erased. If UPGRADE is used when the cluster is open, the upgrade attribute does not apply to the alternate index until the cluster is closed and then opened (that is, a new set of VSAM control blocks describes the cluster and its attributes).

Use UPGRADE for an empty alternate index (that is, an alternate index that is defined but not built). However, the UPGRADE attribute is not effective for the alternate index until the alternate index is built (see the BLDINDEX command).

**Abbreviation:** UPG

**NOUPGRADE**
Specifies the alternate index is not to be modified when the its base cluster is modified. NOUPGRADE can be use as an alternate index at any time.

**Abbreviation:** NUPG

**WRITECHECK|NOWRITECHECK**
Specifies whether a data or index component is to be checked by a machine action called write check when a record is written into it. This parameter can be specified to alter the data or index components of a cluster, an alternate index, or catalog.

The WRITECHECK setting is ignored when the data set is opened for RLS.

**WRITECHECK**
Writes and reads a record without data transfer, to test for the data check condition.

**Abbreviation:** WCK

**NOWRITECHECK**
Writes the record only

**Abbreviation:** NWCK

# ALTER Examples

## Alter a Cluster's Attributes Using SMS Keywords: Example 1

In this example, the ALTER command is used with the MANAGEMENTCLASS and STORAGECLASS keywords.

```
//ALTER    JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
    ALTER  -
           CLUS.ALTER.EXAMPLE -
           MANAGEMENTCLASS(VSAM) -
           STORAGECLASS(FAST)
/*
```

The ALTER command modifies some of the attributes of SMS-managed data set CLUS.ALTER.EXAMPLE. It is expected to grow and require an increase in the frequency of backup, availability and performance. The parameters are MANAGEMENTCLASS, indicating a new management class of VSAM, and STORAGECLASS, indicating a storage class of FAST.

## Roll-In a Generation Data Set: Example 2

In this example, the ALTER command is used with the ROLLIN keyword to roll-in a generation data set currently in the deferred roll-in state.

```
//ALTER    JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALTER -
           DATA.G0001V05 -
           ROLLIN
/*
```

The ALTER command rolls the SMS-managed generation data set, DATA.G0001V05, into the GDG base.

## Alter the Entry Names of Generically Named Clusters: Example 3

In this example, several clusters with similar names, GENERIC.*.BAKER (where * is any 1-to-8 character simple name), are renamed so that their entry names are GENERIC.*.ABLE. The name "GENERIC.*.BAKER" is called a generic name.

```
//ALTER2   JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     ALTER -
           GENERIC.*.BAKER -
           NEWNAME(GENERIC.*.ABLE)
/*
```

The ALTER command changes each generic entry name, GENERIC.*.BAKER, to GENERIC.*.ABLE. Its parameters are:

- GENERIC.*.BAKER identifies the objects to be modified.
- NEWNAME changes each generic entry name GENERIC.*.BAKER to GENERIC.*.ABLE.

## Alter the Attributes of a Generation Data Group: Example 4

This example modifies the attributes of a generation data group. Because the attributes are cataloged in the generation data group's base catalog entry, only this entry is modified.

```
//ALTER3   JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     ALTER -
           GDG01 -
           NOEMPTY -
           SCRATCH
/*
```

The ALTER command modifies some of the attributes of generation data group GDG01. The new attributes override any previously used for the GDG. Its parameters are:

- GDG01 identifies the object to be modified.
- NOEMPTY uncatalogs only the oldest generation data set when the maximum number of cataloged generation data sets is exceeded.
- SCRATCH removes the generation data set's DSCB from the volume VTOC when the data sets is uncataloged. If the data set is SMS-managed, the NVR is also removed.

## Alter a Data Set Expiration Date: Example 6

In this example, an ALTER command is used to modify the expiration date of data set MOD.ALTER.EXAMPLE with the keyword TO.

```
//ALTER5   JOB    ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALTER -
           MOD.ALTER.EXAMPLE -
           TO(1989123)
/*
```

The command's parameters are:

- MOD.ALTER.EXAMPLE is the name of the data set.
- TO changes the expiration date of the data set by name. The year (1989) is a four-digit number, concatenated with the day (123). You can also use two digits (89) to indicate the year. For expiration dates beyond the year 1999, a four-digit year must be specified.

## Migrate a DB2® Cluster to a Linear Data Set Cluster: Example 7

In this example, ALTER is used to alter a DB2 cluster EXAMPLE.ABC01, to a linear data set cluster.

```
//DB2TOLDS  JOB    ...
//STEP1     EXEC PGM=IDCAMS
//SYSPRINT  DD   SYSOUT=A
//SYSIN     DD   *
     ALTER  -
            EXAMPLE.ABC01 -
            TYPE(LINEAR)
/*
```

The command's parameter TYPE(LINEAR) requests ALTER change the data set type from ESDS to LDS.

## Alter a Cluster Name and the Associated Data and Index Names: Example 8

In this example, ALTER is used to rename a cluster and its associated data and index entries.

```
//EXAMPL   JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
    DEFINE CLUSTER -
           (NAME(EXAMPLE.KSDS) -
           TRK(1 1) -
           VOL (338001)) -
           DATA -
             (NAME(EXAMPLE.KSDS.DATA)) -
           INDEX -
             (NAME(EXAMPLE.KSDS.INDEX))
        ALTER -
           EXAMPLE.KSDS -
           NEWNAME(EXAMPLE.TEST)
        ALTER -
           EXAMPLE.KSDS.* -
           NEWNAME(EXAMPLE.TEST.*)
/*
```

In the first part of the example, DEFINE CLUSTER defines a cluster and its data and index components with the same high-level qualifier, with these names:

- EXAMPLE.KSDS
- EXAMPLE.KSDS.DATA
- EXAMPLE.KSDS.INDEX

In the second part of the example, ALTER renames the cluster and its components.

The first ALTER command parameters are:

- EXAMPLE.KSDS identifies the object to be modified (cluster component previously defined).
- NEWNAME changes the entry name EXAMPLE.KSDS to EXAMPLE.TEST. This alters the cluster name to:
  - EXAMPLE.TEST

The second ALTER command parameters are:

- EXAMPLE.KSDS.* identifies the objects to be modified (data and index components previously defined).
- NEWNAME changes each generic entry name EXAMPLE.KSDS.* to EXAMPLE.TEST.*. This alters the data and index names to:
  - EXAMPLE.TEST.DATA
  - EXAMPLE.TEST.INDEX

**Important:** Use the second example of the ALTER command with caution. *Any* data set with the first two qualifiers EXAMPLE.KSDS will be altered.

# Chapter 6. ALTER LIBRARYENTRY

The ALTER LIBRARYENTRY command modifies the attributes of an existing tape library entry. Use this command to recover from tape volume catalog errors.

Because access method services cannot change the library manager inventory in an automated tape library, use Interactive Storage Management Facility (ISMF) for normal tape library alter functions.

The syntax of the access method services ALTER LIBRARYENTRY command is:

| | |
|---|---|
| **ALTER** | *entryname*<br>**LIBRARYENTRY**<br>**[CONSOLENAME**(*consolename*)**]**<br>**[DESCRIPTION**(*desc*)**]**<br>**[LIBDEVTYPE(***devtype***)]**<br>**[LIBRARYID**(*libid*)**]**<br>**[LOGICALTYPE{AUTOMATED\|MANUAL}]**<br>**[NULLIFY([LIBDEVTYPE][LOGICALTYPE])]**<br>**[NUMBEREMPTYSLOTS(***numslots***)]**<br>**[NUMBERSCRATCHVOLUMES(MEDIA1(***num***) MEDIA2(***num***) MEDIA3(***num***) MEDIA4(***num***))]**<br>**[NUMBERSLOTS(***numslots***)]**<br>**[SCRATCHTHRESHOLD(MEDIA1(***num***) MEDIA2(***num***) MEDIA3(***num***) MEDIA4(***num***))]** |

## ALTER LIBRARYENTRY Parameters

The ALTER LIBRARYENTRY parameters are described in the following sections.

### Required Parameters

*entryname*
Identifies the name of the tape library entry being altered. This entry consists of the 1-to-8 character tape library name.

**LIBRARYENTRY.**
Alters a tape library entry. To alter a library entry, you must have access to RACF facility class profile STGADMIN.IGG.LIBRARY.

**Abbreviation:** LIBENTRY|LIBENT

### Optional Parameters

**CONSOLENAME(***consolename***)**
Identifies the name of the console that will receive tape library related messages.

*consolename*
Specifies a 2-to-8 character console name starting with an alphabetic character.

**Abbreviation:** CONSOLE

© Copyright IBM Corp. 1973, 2002

**DESCRIPTION(*desc*)**
Is a description for the tape library entry being altered.

*desc*
Lets you include a 1-to-120 character tape library description. If the description contains commas, semicolons, embedded blanks, parentheses, or slashes, the entire description must be enclosed in single quotation marks. The default for this parameter is blanks.

**Abbreviation:** DESC

**LIBDEVTYPE(*devtype*)**
Identifies the tape library device type.

*devtype*
Is an 8-character hardware device type. If you do not use this, LIBDEVTYPE is not established.

**Abbreviation:** LDEVT

**LIBRARYID(*libid*)**
Establishes the connection between the software-assigned tape library name and the actual tape library hardware.

*libid*
Is a 5-digit hexadecimal tape library serial number.

**Abbreviation:** LIBID

**LOGICALTYPE{AUTOMATED|MANUAL}**
Identifies the type of tape library being created. If you do not use this parameter, LOGICALTYPE is not established.

**AUTOMATED**
Indicates an automated tape library.

**MANUAL**
Is a manual tape library.

**Abbreviation:** LOGTYP

**NULLIFY([LIBDEVTYPE][LOGICALTYPE])**
Identifies the fields to be nullified. You can enter one or both; they are not mutually exclusive.

**LIBDEVTYPE**
specifies that this parameter be set to blanks, indicating that the parameter is not established.

**Abbreviation:** LDEVT

**LOGICALTYPE**
Specifies that the value of this parameter be set to blanks, which implies that this parameter is not established.

**Abbreviation:** LOGTYP

**NUMBEREMPTYSLOTS(*numslots*)**
Identifies the total number of empty slots in the given tape library. You can use it only when LOGICALTYPE is AUTOMATED.

*numslots*
Is the number of tape cartridges you can add to the tape library. Use a number from 0 to 999999. The default is 0.

**Abbreviation:** NUMESLT

**NUMBERSCRATCHVOLUMES(MEDIA1(*num*) MEDIA2(*num*) MEDIA3(*num*) MEDIA4(*num*))**
Identifies the total number of MEDIA1, MEDIA2, MEDIA3, and MEDIA4 scratch volumes currently available in the given tape library.

**MEDIA1(*num*)**
Is the number of Cartridge System Tape scratch volumes available. Use a number from 0 to 999999. The default is 0.

**MEDIA2(*num*)**
Specifies the number of Enhanced Capacity Cartridge System Tape scratch volumes available. Use a number from 0 to 999999. The default is 0.

**MEDIA3(*num*)**
Is the number of High Performance Cartridge Tape scratch volumes available. Use a number from 0 to 999999. The default is 0.

**MEDIA4(*num*)**
Specifies the number of IBM Extended High Performance Cartridge Tape scratch volumes available. Use a number from 0 to 999999. The default is 0.

**Abbreviation:** NUMSCRV

**NUMBERSLOTS(*numslots*)**
Is the total number of slots in the given tape library. You can use this parameter only when LOGICALTYPE is **AUTOMATED**.

*numslots*
Is the total number of tape cartridges that can be contained in the tape library. Use a number from 0 to 999999. The default is 0.

**Abbreviation:** NUMSLT

**SCRATCHTHRESHOLD(MEDIA1(*num*) MEDIA2(*num*) MEDIA3(*num*) MEDIA4(*num*))**
Identifies the scratch volume message threshold. When the number of scratch volumes in the tape library falls below the scratch threshold, an operator action message, requesting that scratch volumes be entered into the tape library, is issued to the library's console. When the number of scratch volumes exceeds twice the scratch threshold, the message is removed from the console.

**MEDIA1(*num*)**
Specifies the threshold number of Cartridge System Tape scratch volumes. Use a number from 0 to 999999. The default is 0.

**MEDIA2(*num*)**
Is the threshold number of Enhanced Capacity System Tape scratch volumes. Use a number from 0 to 999999. The default is 0.

**MEDIA3(*num*)**
Specifies the threshold number of High Performance Cartridge Tape scratch volumes. Use a number from 0 to 999999. The default is 0.

**MEDIA4(*num*)**
Is the threshold number of IBM Extended High Performance Cartridge Tape scratch volumes. Use a number from 0 to 999999. The default is 0.

**Abbreviation:** SCRTHR

# ALTER LIBRARYENTRY Examples

## Altering a Tape Library Entry: Example 1

This example alters the entry for the tape library ATLLIB1.

```
//ALTERLIB JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALTER ATLLIB1 -
           LIBRARYENTRY -
           NUMBEREMPTYSLOTS(2574) -
           NUMBERSCRATCHVOLUMES(MEDIA1(500) MEDIA2(400)) -
           SCRATCHTHRESHOLD(MEDIA1(200) MEDIA2(100))

/*
```

This command's parameters are:

- ATLLIB1 is the name of the entry being altered.
- LIBRARYENTRY alters a tape library entry.
- NUMBEREMPTYSLOTS sets the number of empty slots to 2574.
- NUMBERSCRATCHVOLUMES sets the current number of scratch volumes available for MEDIA1 to 500 and for MEDIA2 to 400.
- SCRATCHTHRESHOLD sets the threshold number of scratch volumes for MEDIA1 to 200 and for MEDIA2 to 100.

## Altering a LIBRARY Entry: Example 2

This example alters the entry that describes the LIBRARY ATLLIB1.

```
//ALTERLIB JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALTER ATLLIB1 -
           LIBRARYENTRY -
           NUMBEREMPTYSLOTS(2574) -
           NUMBERSCRATCHVOLUMES(MEDIA3(1272)) -
           SCRATCHTHRESHOLD(MEDIA3(125))
```

This command's parameters are:

- ATLLIB1 specifies the name of the entry being altered.
- LIBRARYENTRY indicates that a LIBRARY entry is being altered.
- NUMBEREMPTYSLOTS specifies that the number of empty slots available be set to 2574.
- NUMBERSCRATCHVOLUMES specifies that the current number of scratch volumes available for MEDIA3 be set to 1272.
- SCRATCHTHRESHOLD specifies that the threshold number of scratch volumes for MEDIA3 be set to 125.

# Chapter 7. ALTER VOLUMEENTRY

The ALTER VOLUMEENTRY command modifies the attributes of an existing tape volume entry. Use this command only to recover from tape volume catalog errors.

Because access method services cannot change the library manager inventory in an automated tape library, Interactive Storage Management Facility should be used for normal tape library alter functions.

The syntax of the access method services ALTER VOLUMEENTRY command is:

| | |
|---|---|
| **ALTER** | **(***entryname***)**<br>**VOLUMEENTRY**<br>**[CHECKPOINT\|NOCHECKPOINT]**<br>**[COMPACTION{YES\|IDRC\|NO\|NONE\|UNKNOWN}]**<br>**[ENTEREJECTDATE(***eedate***)]**<br>**[EXPIRATIONDATE(***expdate***)]**<br>**[LIBRARYNAME(***libname***)]**<br>**[LOCATION{LIBRARY\|SHELF}]**<br>**[MEDIATYPE{MEDIA1\|MEDIA2\|MEDIA3\|MEDIA4}]**<br>**[MOUNTDATE(***mountdate***)]**<br>**[NULLIFY(ERRORSTATUS)]**<br>**[OWNERINFORMATION(***ownerinfo***)]**<br>**[RECORDING{18TRACK\|36TRACK\|128TRACK\|256TRACK\|UNKNOWN}]**<br>**[SHELFLOCATION(***shelf***)]**<br>**[SPECIALATTRIBUTE{READCOMPATIBLE\|NONE}]**<br>**[STORAGEGROUP(***groupname***)]**<br>**[USEATTRIBUTE{SCRATCH\|PRIVATE}]**<br>**[WRITEDATE(***wrtdate***)]**<br>**[WRITEPROTECT\|NOWRITEPROTECT]** |

## ALTER VOLUMEENTRY Parameters

### Required Parameters

*entryname*
Names the tape volume entry being altered. This name consists of a V concatenated with the 1-to-6 character volser. See “Tape Volume Names” on page 9 for tape volume volser naming conventions.

**VOLUMEENTRY**
Alters a tape volume entry. To alter a tape volume entry, you must have access to RACF facility class profile STGADMIN.IGG.LIBRARY.

**Abbreviation:** VOLENTRY|VOLENT

### Optional Parameters

**CHECKPOINT|NOCHECKPOINT**
Checks whether the tape volume is a secure checkpoint volume. If you do not use this, the checkpoint status is unknown.

**CHECKPOINT**
Indicates that the tape volume is a secure checkpoint volume.

© Copyright IBM Corp. 1973, 2002

**Abbreviation:** CHKPT

**NOCHECKPOINT**
Indicates that the volume is not a secure checkpoint volume.

**Abbreviation:** NOCHKPT

**COMPACTION{|YES|IDRC|NO|NONE|UNKNOWN}**
Identifies whether the data on the volume is compacted. The YES and IDRC parameter variables are synonymous. The NO and NONE parameter variables are synonymous. Use this parameter only for private tape volumes. If you use it for scratch tape volumes, a default of NONE is forced.

**YES**
Specifies that data is compacted in the manner appropriate for the type of media.

**IDRC**
Specifies that improved data recording capability (IDRC) compaction was used.

**NO**
Specifies that no compaction was used.

**NONE**
Specifies that no compaction was used.

**UNKNOWN**
Specifies that it is unknown if compaction was used.

**Abbreviation:** COMP

**ENTEREJECTDATE(*eedate*)**
Identifies the date that a tape volume was last ejected from a tape library or last entered into a tape library.

*eedate*
The date, as YYYY-MM-DD. See “Tape Library Date Formats” on page 9 for valid dates. The default is blank.

**Abbreviation: EEDATE**

**EXPIRATIONDATE(*expdate*)**
Identifies the date on which the tape volume expires. If there is more than one data set on the volume, the expiration date is the latest expiration date among the data sets on the volume.

*expdate*
Enter a date as YYYY-MM-DD. The expiration date is set to blanks when the USEATTRIBUTE is SCRATCH.

**Abbreviation:** EXDATE

**LIBRARYNAME(*libname*)**
Identifies the name of the tape library in which this tape volume resides. If you use this parameter, the parameter LOCATION must equal LIBRARY. If LOCATION equals SHELF, the library name is set to SHELF.

*libname*
A 1-to-8 character library name.

**Abbreviation:LIBNAME**

**LOCATION{LIBRARY|SHELF}**
Specifies either that the tape volume resides in a tape library or that it resides on a shelf outside the tape library.
- If you use LIBRARY, you must also use the LIBRARYNAME parameter.
- If you use SHELF, the library name defaults to SHELF.

**Abbreviation:** LOC

**MEDIATYPE{ MTYPEMEDIA1|MEDIA2|MEDIA3|MEDIA4}**
Identifies the media type of the tape volume. If you do not use this, MEDIATYPE defaults to MEDIA2.

**MEDIA1**
Specifies that the tape volume is Cartridge System Tape.

**MEDIA2**
Specifies that the tape volume is Enhanced Capacity System Tape. You cannot use this parameter when SPECIALATTRIBUTE is READCOMPATIBLE, or RECORDING is set to 18TRACK.

**MEDIA3**
Specifies that the tape volume is High Performance Cartridge Tape.

**MEDIA4**
Specifies that the tape volume is IBM Extended High Performance Cartridge Tape.

**Abbreviation:**MTYPE

**MOUNTDATE(*mountdate*)**
The date on which the tape volume was last mounted onto a tape drive and successfully opened.

*mountdate*
The date, YYYY-MM-DD. See “Tape Library Date Formats” on page 9 for a description of valid date values. The default is blank.

**Abbreviation:** MDATE

**NULLIFY(ERRORSTATUS)**
Gives the fields to be nullified.

**ERRORSTATUS**
If you use this, the error status is set to 0.

**Abbreviation: ERRSTAT|**ER

**OWNERINFORMATION(*ownerinfo*)**
Provides information about the tape volume’s owner.

*ownerinfo*
A 1-to-64 character owner information field. If you use commas, semicolons, embedded blanks, parentheses, or slashes, enclose the entire description in single quotation marks. The default is blanks.

**Abbreviation:** OWNINFO

**RECORDING{18TRACK|36TRACK|128TRACK|256TRACK|UNKNOWN}**
Identifies the recording technique for creating the tape. This parameter can only be used for private tape volumes. Scratch tape volumes default to UNKNOWN.

**18TRACK**
Tape was written and must be read on an 18-track device.

**36TRACK**
Tape was written and must be read on a 36-track device.

**128TRACK**
Tape was written and must be read on a 128-track device.

**256TRACK**
Tape was written and must be read on a 256-track device. This parameter is only valid with MEDIATYPE(MEDIA3) or MEDIATYPE(MEDIA4).

**UNKNOWN**
Tape recording technique is unknown.

**Abbreviation:** REC

**SHELFLOCATION(*shelf*)**
Gives the shelf location for a tape volume that resides outside a tape library. This parameter can be included for a library resident volume.

*shelf*
The 1-to-32 character shelf location information field. If your description contains commas, semicolons, embedded blanks, parentheses, or slashes, enclose the entire description in single quotation marks. The default is blank.

**Abbreviation:**SHELFLOC

**SPECIALATTRIBUTE{READCOMPATIBLE|NONE}**
Shows special attributes of the tape volume. Use this parameter only for private tape volumes. Scratch tape volumes default to NONE.

**READCOMPATIBLE**
On subsequent allocations, read compatible devices for allocation of this tape volume are used.

**Abbreviation:** RDCOMPAT

**NONE**
There are no special tape attributes.

**Abbreviation:** SATTR

**STORAGEGROUP(*groupname*)**
Identifies the storage group name.

*groupname*
The 1-to-8 character name of the storage group in which this tape volume is defined. The default is blanks. If the USEATTRIBUTE parameter is SCRATCH, however, the storage group name defaults to *SCRTCH*.

**Abbreviation:** STORGRP

**USEATTRIBUTE{SCRATCH|PRIVATE}**
Identifies the use attribute of a tape volume. You can use SCRATCH for scratch volumes, or PRIVATE for private volumes (tape volumes with unexpired data sets on them). The default is PRIVATE. If you use SCRATCH, the storage group name is set to *SCRTCH*. and the expiration date is set to blanks.

**Abbreviation:**UATTR

**WRITEDATE(*wrtdate*)**
Identifies the last date that a data set on the tape volume was opened for writing.

*wrtdate*
A date, YYYY-MM-DD. The default is blank.

**Abbreviation:**WDATE

**WRITEPROTECT|NOWRITEPROTECT**
Identifies whether the tape volume is write-protected or not. If you do not use this, write-protect status is unknown.

**WRITEPROTECT**
Indicates that the volume is write-protected.

Setting **WRITEPROTECT** in the tape volume entry does not automatically write protect your volume. It is an informational setting that is recorded when software detects that the volume is write protected and is not used by software when determining whether the volume is protected. For a volume to be write protected, you will still need to set the write protect tab available on the physical tape volume. Because the availability to write protect a volume does not exist for logical volumes in a VTS, (no tab available either physically or logically), you can alternately use RACF or PROTECT=ALL in your JCL to protect the volume during usage.

**Abbreviation:**WPRT

**NOWRITEPROTECT**
Iindicates that the volume is not write-protected.

**Abbreviation:**NWPRT

# ALTER VOLUMEENTRY Examples

## Altering a Volume Entry: Example 1

This example alters the tape library volume entry with volser AL0001.

```
//ALTERVOL JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALTER VAL0001 -
           VOLUMEENTRY -
           LIBRARYNAME(ATLLIB1) -
           LOCATION(LIBRARY) -
           USEATTRIBUTE(SCRATCH) -
           EXPIRATIONDATE(2000-12-31)

/*
```

This command's parameters are:

- VAL0001: Specifies the name of the tape volume entry being altered. The volume's volser is AL0001.
- VOLUMEENTRY: Indicates that an entry describing a single tape volume (that is a cartridge) in a tape library is being altered.
- LIBRARYNAME: Specifies that this tape volume record is associated with a tape library named ATLLIB1.
- LOCATION: Specifies that the tape volume will now reside in a tape library slot.
- USEATTRIBUTE: Specifies that the tape volume is a scratch volume.

- EXPIRATIONDATE: Specifies an expiration date of 2000-12-31. On that date the data set on the tape volume will expire; however, because USEATTRIBUTE is specified as SCRATCH, the expiration date is set to blanks.

## Altering a VOLUME Entry: Example 2

This example alters the entry that describes the VOLUME AL0001.

```
//ALTERVOL JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALTER VAL0001 -
           VOLUMEENTRY -
           LIBRARYNAME(ATLLIB1) -
           USEATTRIBUTE(PRIVATE) -
           MEDIATYPE(MEDIA3) -
           RECORDING(128TRACK))
```

This command's parameters are:

- VOLUMEENTRY: Indicates that an entry describing a single volume (i.e. a cartridge) in a LIBRARY is being altered.
- VAL0001: Specifies the name of the VOLUMEENTRY entry being altered and the volser AL0001.
- LIBRARYNAME: Specifies that this VOLUME record is associated with LIBRARY ATLLIB1.
- USEATTRIBUTE: Specifies that the volume will be a PRIVATE volume.
- MEDIATYPE: Specifies the media type of MEDIA3.
- RECORDING: Specifies the recording technique 128TRACK.

# Chapter 8. BLDINDEX

The BLDINDEX command builds alternate indexes for existing data sets. The syntax of the BLDINDEX command is:

| | |
|---|---|
| **BLDINDEX** | **{INFILE(***ddname***)\|**<br>**INDATASET(***entryname***)}**<br>**{OUTFILE(***ddname*<br>**[** *ddname***...])\|**<br>**OUTDATASET(***entryname*<br>**[** *entryname***...])}**<br>**[{EXTERNALSORT\|INTERNALSORT}]**<br>**[{SORTCALL\|NOSORTCALL}]**<br>**[SORTDEVICETYPE***(device type)***]**<br>**[SORTFILENUMBER***(number)***]**<br>**[SORTMESSAGEDD***(ddname)***]**<br>**[SORTMESSAGELEVEL({ALL\|CRITICAL\|NONE})]**<br>**[WORKFILES(***ddname* *ddname***)]**<br>**[CATALOG(***catname***)]** |

BLDINDEX can be abbreviated: BIX

**Requirement:** If you use BLDINDEX and intend to use the default sort product (DFSORT™ or equivalent), you must ensure that IDCAMS is called in problem state.

## BLDINDEX Parameters

### Required Parameters

**INFILE(***ddname***)|INDATASET(***entryname***)**
names the DD statement or data set that identifies the base cluster or a path that points to the base cluster.

**INFILE(***ddname***)**
is the DD statement that identifies the base cluster or a path that points to the base cluster. You must define the base cluster in the same catalog as the alternate index, and it must contain at least one data record.

**Abbreviation:** IFILE

**INDATASET(***entryname***)**
names the data set that identifies the base cluster or a path that points to the base cluster. You must define the base cluster in the same catalog as the alternate index, and it must contain at least one data record.

When you use INDATASET to dynamically allocate the base-cluster volume, make sure the base-cluster volume is mounted as permanently resident or reserved.

**Abbreviation:** IDS

**OUTFILE(***ddname***)|OUTDATASET(***entryname***)**
names the DD statement or data set that identifies the alternate index or a path that points to the alternate index. If the data set has previously been deleted and redefined in this same invocation of IDCAMS and the FILE parameter was

© Copyright IBM Corp. 1973, 2002

specified on the delete, you must specify the OUTDATASET keyword instead of OUTFILE to avoid picking up incorrect volume information from the original DD statement. Alternately, you may issue the BLDINDEX in a different step than the step that did the delete and define. You can build more than one alternate index for the same base cluster by using more than one ddname or data set name with the OUTFILE or OUTDATASET parameter.

**OUTFILE(***ddname***[** *ddname***...])**
indicates the DD statement that identifies the alternate index, or a path that points to the alternate index. You must define the alternate index in the same catalog as the base cluster, and it must be empty (that is, its high-used relative byte address equals zero) or defined with the REUSE attribute.

The alternate index must be related to the base cluster identified with INDATASET or INFILE.

**Abbreviation:** OFILE

**OUTDATASET(***entryname***[** *entryname***...])**
specifies the data set that identifies the alternate index or a path that points to the alternate index. When you define the alternate index in the same catalog as the base cluster, it must be empty (that is, its high-used RBA equals zero) or must be defined with the REUSE attribute.

The alternate index must be related to the base cluster identified with INDATASET or INFILE.

When you use OUTDATASET, to dynamically allocate the alternate index's volume, make sure the volume is mounted as permanently resident or reserved.

**Abbreviation:** ODS

## Optional Parameters

**CATALOG(***catname***)**
names the catalog in which the work files are to be defined. The work files are defined and used by the BLDINDEX routine. When all alternate indexes are built and the BLDINDEX routine no longer needs the work files, they are deleted. See "Catalog Selection Order for BLDINDEX" on page 10 for more information.

To use catalog names for SMS-managed data sets, you must have access to the RACF STGADMIN.IGG.DIRCAT facility class. See "Storage Management Subsystem (SMS) Considerations" on page 2 for more information.

**Abbreviation:** CAT

**EXTERNALSORT|INTERNALSORT**
decides whether the key-pointer pairs are to be sorted entirely within virtual storage.

**EXTERNALSORT**
specifies that two external-sort work files are defined and built as entry-sequenced clusters. You must provide two DD statements that describe the external-sort work files to be defined by BLDINDEX. You can name the DD statements IDCUT1 and IDCUT2. When you choose other names for the work file DD statements, you must identify those DD statements with the WORKFILES parameter.

**Abbreviation:** ESORT

**INTERNALSORT**
requires access method services to sort the key-pointer pairs entirely within the user-provided virtual storage, if possible. If you do not have enough virtual storage available when you use INTERNALSORT, two external-sort work files are built and the key-pointer pairs are sorted externally. You must provide DD statements as for EXTERNALSORT. If the minimum amount of virtual storage is not provided the BLDINDEX processing ends with an error message. See *z/OS DFSMS: Using Data Sets* for more information.

**Abbreviation:** ISORT

**{SORTCALL|NOSORTCALL}**
use this parameter to choose whether or not to call DFSORT to sort the alternate index.

**SORTCALL**
specifies that you want DFSORT to sort the alternate index. EXTERNALSORT, INTERNALSORT, WORKFILES, CATALOG, IDCUT1, and IDCUT2 are ignored when DFSORT is called. If DFSORT is not available, BLDINDEX uses the IDCAMS internal sort. SORTCALL is the default.

**NOSORTCALL**
tells BLDINDEX to use the IDCAMS internal sort (or external sort if specified) instead of DFSORT to sort the alternate index. When the IDCAMS internal or external sort is used, SORTMESSAGELEVEL, SORTDEVICETYPE, SORTMESSAGEDD and SORTFILENUMBER specifications are prohibited.

**SORTDEVICETYPE***(device type)*
specifies the DASD device type passed to DFSORT in the DYNALLOC parameter in the OPTION control statement. Use this parameter only if you wish to override the default device type for DFSORT work data sets. See *DFSORT Application Programming Guide Release 14* for further details on the DYNALLOC parameter. This parameter is not allowed if you use NOSORTCALL.

**Abbreviation:** SORTDVT SDVT

**SORTFILENUMBER***(number)*
the maximum number of work data sets passed to DFSORT in the DYNALLOC parameter in the OPTION control statement. Use this parameter to override the number of work data sets that BLDINDEX determines are needed. See *DFSORT Application Programming Guide Release 14* for further details on the DYNALLOC parameter. This parameter is not allowed if you use NOSORTCALL.

**Abbreviation:** SORTFN SFN

**SORTMESSAGEDD***(ddname)*
is the ddname that describes the DFSORT message data set. If there is no DD statement for this ddname, a message data set with this ddname is allocated dynamically as a SYSOUT=* data set. SYSOUT is the default for ddname. Do not use any ddname reserved for use by IDCAMS (SYSPRINT or SYSIN) or DFSORT. See *DFSORT Application Programming Guide Release 14* for a list of reserved ddnames. This parameter is not allowed if you use NOSORTCALL or SORTMESSAGELEVEL (NONE).

**Abbreviation:** SORTMDD SMDD

**SORTMESSAGELEVEL({ALL|CRITICAL|NONE})**
is the level of DFSORT messages to print to the DFSORT message data set. You cannot use this parameter with NOSORTCALL.

**Abbreviation:** SORTML SML

**ALL**
Requires that all DFSORT messages and control statements are printed to the message data set.

**CRITICAL**
Allows only critical DFSORT messages to print to the message data set. No DFSORT control statements are printed. Critical is the default.

**NONE**
Allows no DFSORT messages or control statements to print to the message data set.

**WORKFILES(***ddname ddname***)**
names the DD statements that describe the name and placement of the work files you want BLDINDEX to define if you require an external sort of the key-pointer pairs. See the CATALOG parameter for further description of where the work files are defined. You can use DD statements to describe two work files that are defined and opened before the BLDINDEX routine begins processing the base-cluster's data records.

**Exception:** Do not use tape data sets as work data sets.

If one of the data sets is SMS-managed, the other must either be SMS-managed or a non-SMS-managed data set cataloged in the catalog determined by the catalog search order.

When you code the DD statements that describe the work files and identify them with the standard ddnames IDCUT1 and IDCUT2, you do not need to use the WORKFILES parameter.

**Abbreviation:** WFILE

## Calculating Virtual Storage Space for an Alternate Index

When BLDINDEX builds an alternate index, access method services opens the base cluster to sequentially read the data records, sorts the information obtained from the data records, and builds the alternate index records:

1. The base cluster is opened for read-only processing. To prevent other users from updating the base cluster's records during BLDINDEX processing, include the DISP=OLD parameter in the base cluster's DD statement. If INDATASET is specified, access method services dynamically allocates the base cluster with DISP=OLD.
2. The base cluster's data records are read and information is extracted to form the key-pointer pair:
   - When the base cluster is entry-sequenced, the alternate key value and the data record's RBA form the key-pointer pair.
   - When the base cluster is key-sequenced, the alternate key value and the data record's prime key value form the key-pointer pair.

   If the base cluster's data records can span control intervals the alternate key must be in the record's first control interval.

3. The key-pointer pairs are sorted in ascending alternate key order. If your program provides enough virtual storage, access method services does an internal sort. (The sorting of key-pointer pairs takes place entirely within virtual storage.)

   Use the following process to determine the amount of virtual storage required to sort the records internally:

   a. Sort record length = alternate key length + (prime key length (for a key-sequenced data set) or 4 (for an entry-sequenced data set)).
   b. Record sort area size = either the sort record length times the number of records in the base cluster rounded up to the next integer multiple of 2048 (the next 2K boundary), or a minimum of 32768, whichever is greater.
   c. Sort table size = (record sort area size/sort record length) x 4.
   d. The sum of b + c = required amount of virtual storage for an internal sort. (The amount for an internal sort is in addition to the normal storage requirements for processing an access method services command.)

   If you do not provide enough virtual storage for an internal sort, or if you specify the EXTERNALSORT parameter, access method services defines and uses two sort work files and sorts the key-pointer pairs externally. Access method services uses the sort work files to contain most of the key-pointer pairs while it sorts some of them in virtual storage. An external sort work file is a VSAM entry-sequenced cluster, marked reusable. The minimum amount of virtual storage you need for an external sort is:

   32768 + ((32768/sort record length) x 4)

   The amount of space that access method services requests when defining each sort work file is calculated as follows:

   a. Sort records per block = 2041/sort record length
   b. Primary space allocation in records = (number of records in base cluster/sort records per block) + 10
   c. Secondary space allocation in records = (primary space allocation x 0.10) + 10

   Both primary and secondary space allocation are requested in records with a fixed-length record size of 2041 bytes. The control interval size is 2048 bytes.

   There must be enough space on a single DASD volume to satisfy the primary allocation request; if there is not, the request fails. To correct the problem, specify the volume serial of a device that has sufficient space (see “DD Statements That Describe the Sort Work Files” on page 92).

4. When the key-pointer pairs are sorted into ascending alternate key order, access method services builds an alternate index record for each key-pointer pair. If the NONUNIQUEKEY attribute is used and more than one key-pointer pair has the same alternate key values, the alternate index record contains the alternate key value, followed by the pointer values in ascending order. If the UNIQUEKEY attribute is used, each alternate key value must be unique.

   When the record is built, it is written into the alternate index as though it is a data record loaded into a key-sequenced data set. The record’s attributes and values, specified when the alternate index is defined, include:

   BUFFERSPACE
   CONTROLINTERVALSIZE
   DATACLASS
   FREESPACE

RECORDSIZE
RECOVERY
SPEED
WRITECHECK

5. When all alternate index records are built and loaded into the alternate index, the alternate index and its base cluster are closed. Steps 1 through 4 are repeated for each alternate index that is specified with the OUTFILE and OUTDATASET parameter. When all alternate indexes are built, any defined external sort work files are deleted. Access method services finishes processing and issues messages that show the results of the processing.

## DD Statements That Describe the Sort Work Files

VSAM data set space available for the sort routine can be identified by specifying two ddnames with the WORKFILES parameter and supplying two DD statements that describe the work files to be defined. Each work file DD statement should be coded:

```
//ddname DD  DSNAME=dsname,VOL=SER=volser,
//           UNIT=devtype,DISP=OLD,AMP='AMORG'
```

**Exception:** WORKFILES is ignored when DFSORT is available to do the sorting of the alternate index and you have not overridden the default by specifying NOSORTCALL.

ddname
as specified in the WORKFILES parameter. If you do not specify the WORKFILES parameter and you intend to provide VSAM data set space for external sort work files, identify the work file DD statements with the names IDCUT1 and IDCUT2.

dsname
a data set name. The scheduler generates a data set name for the work file if none is provided. The data set will be cataloged into the master catalog unless a STEPCAT/JOBCAT is specified and deleted when BLDINDEX processing has completed. A data set name must be specified if the user, or the user's group, is defined to RACF with the automatic data set protection (ADSP) attribute. The first qualifier of the data set name must be a valid user or group name.

JOBCAT and STEPCAT DD statements are not supported for system-managed data sets. The JOBCAT or STEPCAT DD statement will fail if it refers to a system-managed catalog, or if the data set being searched is system-managed. Also, all referenced integrated catalog facility catalogs must be connected to the system master catalog.

**VOL=SER=**volser
Required. Identifies the volume owned by the STEPCAT, JOBCAT, or master catalog where the work file is cataloged. The work file's space is allocated from the volume's space. You can specify a maximum of five volumes for each work file. For a description of how to calculate the amount of space to be allocated for each sort work file, see "Chapter 8. BLDINDEX" on page 87. If your BLDINDEX job requires external sort work files, this space must be available on the volume(s) identified by *volser* or your job will fail.

**UNIT=**devtype
type of direct access device on which the volume is mounted. You can specify a generic device type (for example, 3380) or a device number (for example 121). You cannot specify SYSDA.

**DISP=OLD**
Required.

**AMP='AMORG'**
Required.

If BLDINDEX is used interactively in a TSO environment, these sort work file DD statements must be in the logon procedure.

## BLDINDEX Examples

The BLDINDEX command can be used to perform the functions shown in the following examples.

### Build an Alternate-Index over a Key-Sequenced Data Set (KSDS): Example 1

This example builds an alternate index over a previously defined base cluster, EXAMPLE.KSDS2. Data records are already loaded into the base cluster. The alternate index, its path, and its base cluster are all defined in the same catalog, USERCAT.

```
//BUILDAIX JOB  ...
//BASEDD   DD   DSNAME=EXAMPLE.KSDS2,DISP=OLD
//AIXDD    DD   DSNAME=EXAMPLE.AIX,DISP=OLD
//IDCUT1   DD   DSNAME=SORT.WORK.ONE,DISP=OLD,AMP='AMORG',
//     VOL=SER=VSER01,UNIT=DISK
//IDCUT2   DD   DSNAME=SORT.WORK.TWO,DISP=OLD,AMP='AMORG',
//     VOL=SER=VSER01,UNIT=DISK
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
  BLDINDEX INFILE(BASEDD) -
           OUTFILE(AIXDD) -
           NOSORTCALL -
           CATALOG(USERCAT)
/*
```

Job control language statements:

- BASEDD DD describes the base cluster.
- AIXDD DD describes the alternate index.
- IDCUT1 and IDCUT2 DD describe volumes available as sort work data sets if an external sort is done. They are not used by BLDINDEX if enough virtual storage is available for an internal sort. If there are multiple volumes, a maximum of five volumes for each work file can be specified.

The BLDINDEX command builds an alternate index. If there is not enough virtual storage for an internal sort, DD statements with the default ddnames of IDCUT1 and IDCUT2 are given for two external-sort work data sets.

The parameters are:

- INFILE names the base cluster. The ddname of the DD statement for this object must be identical to this name.
- OUTFILE names the alternate index. The ddname of the DD statement for this object must be identical to this name.
- CATALOG identifies the user catalog.

## Build an Alternate-Index over a Key-Sequenced Data Set (KSDS) Using DFSORT: Example 2

This example, using DFSORT, builds an alternate index over a previously defined base cluster, EXAMPLE.KSDS2. Data records are already loaded into the base cluster. The alternate index, its path, and its base cluster are all defined in the same catalog, USERCAT.

```
//BUILDAIX JOB  ...
//BASEDD   DD   DSNAME=EXAMPLE.KSDS2,DISP=OLD
//AIXDD    DD   DSNAME=EXAMPLE.AIX,DISP=OLD
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
  BLDINDEX INFILE(BASEDD) -
           OUTFILE(AIXDD/AIXUPPW) -
           SORTCALL -
           SORTMESSAGELEVEL(ALL)
/*
```

Job control language statements:
- BASEDD DD describes the base cluster.
- AIXDD DD describes the alternate index.

The BLDINDEX command builds an alternate index. BLDINDEX calls DFSORT to sort the alternate index records. If DFSORT is not available, BLDINDEX uses its own sort routines.

The parameters are:
- INFILE names the base cluster. The ddname of the DD statement for this object must be identical to this name.
- OUTFILE names the alternate index. The ddname of the DD statement for this object must be identical to this name.
- SORTCALL tells BLDINDEX to call DFSORT to sort the alternate index records. This parameter is the default.
- SORTMESSAGELEVEL(ALL) requires that all DFSORT messages and control statements are returned in the DFSORT message data set.

# Chapter 9. CREATE LIBRARYENTRY

The CREATE LIBRARYENTRY command creates a tape library entry. Use it only to recover from tape volume catalog errors.

Because access method services cannot change the library manager inventory in an automated tape library, ISMF should be used for normal tape library create functions.

The syntax for the CREATE LIBRARY command is:

| | |
|---|---|
| **CREATE** | **LIBRARYENTRY**<br>**(NAME(***entryname***)**<br>**LIBRARYID**(*libid*)<br>**[CONSOLENAME**(*consolename*)**]**<br>**[DESCRIPTION**(*desc*)**]**<br>**[LIBDEVTYPE(***devtype***)]**<br>**[LOGICALTYPE{AUTOMATED\|MANUAL}]**<br>**[NUMBEREMPTYSLOTS(***numslots***)]:**<br>**[NUMBERSCRATCHVOLUMES(MEDIA1(***num***) MEDIA2(***num***)**<br>**MEDIA3(***num***) MEDIA4(***num***))]**<br>**[NUMBERSLOTS**(*numslots*)**]**<br>**[SCRATCHTHRESHOLD(MEDIA1(***num***) MEDIA2(***num***) MEDIA3(***num***)**<br>**MEDIA4(***num***))]** |

## Required Parameters

**LIBRARYENTRY**
is the name of the tape library entry being created. To create a library entry, you must have authorization to RACF facility class profile STGADMIN.IGG.LIBRARY.

**Abbreviation:** LIBENTRY|LIBENT

**NAME(***entryname***)**
is the name of the tape library entry being created.

*entryname*
Consists of a 1-to-8 character tape library name. The characters can include alphanumerics, $, @, and #. The first character cannot be numeric.

To avoid conflicts with volume names, library names cannot begin with the letter V.

**LIBRARYID(***libid***)**
this number connects the software-assigned tape library name and the actual tape library hardware.

*libid*
is a 5-digit hexadecimal tape library serial number.

**Abbreviation:** LIBID

© Copyright IBM Corp. 1973, 2002

## Optional Parameters

**CONSOLENAME(*consolename*)**
Names the console that receives messages related to the tape library.

*consolename*
Is a 2-to-8 character console name, starting with an alphabetic character.

**Abbreviation:** CONSOLE

**DESCRIPTION(*desc*)**
Identifies a description for the tape library being created.

*desc*
Is a 1-to-120 character tape library description. If you use commas, semicolons, embedded blanks, parentheses, or slashes, enclose the entire description in single quotation marks. The default is blanks.

**Abbreviation:** DESC

**LIBDEVTYPE(*devtype*)**
identifies the tape library device type.

*devtype*
is an 8 character hardware device type. If you do not use this, LIBDEVTYPE is not established.

**Abbreviation:** LDEVT

**LOGICALTYPE{AUTOMATED|MANUAL}**
identifies the type of tape library being created. If you do not use this, LOGICALTYPE is not established.

**AUTOMATED**
is an automated tape library.

**MANUAL**
is a manual tape library.

**Abbreviation:** LOGTYP

**NUMBEREMPTYSLOTS(*numslots*)**
identifies the total number of empty slots in the specified tape library. This parameter can only be specified when LOGICALTYPE is specified as AUTOMATED.

*numslots*
is the number from 0 to 999999, of tape cartridges that can be added to the tape library. The default is 0.

**Abbreviation:** NUMESLT

**NUMBERSCRATCHVOLUMES(MEDIA1(*num*) MEDIA2(*num*) MEDIA3(*num*) MEDIA4(*num*))**
is the total number of MEDIA1, MEDIA2, MEDIA3, and MEDIA4 scratch volumes currently available in the given tape library.

When creating a library entry for an automated tape library dataserver, NUMBERSCRATCHVOLUMES can be specified for MEDIA1, MEDIA2, MEDIA3, and MEDIA4. When creating a library entry for a manual tape library dataserver, NUMBERSCRATCHVOLUMES can only be specified for MEDIA1 and MEDIA2.

**MEDIA1(*num*)**
is the number from 0 to 999999, of Cartridge System Tape scratch volumes available. The default is 0.

**MEDIA2(*num*)**
is the number from 0 to 999999, of Enhanced Capacity Cartridge System Tape scratch volumes available. The default is 0.

**MEDIA3(*num*)**
is the number from 0 to 999999, of High Performance Cartridge Tape scratch volumes available. The default is 0.

**MEDIA4(*num*)**
is the number from 0 to 999999, of MEDIA4 scratch volumes available. MEDIA4 is IBM Extended High Performance Cartridge Tape. The default is 0.

**Abbreviation:** NUMSCRV

**NUMBERSLOTS(*numslots*)**
is the total number of slots in the given tape library. Use this only when LOGICALTYPE is specified as AUTOMATED.

*numslots*
is the total number, from 0 to 999999, of tape cartridges that can be contained in the tape library. The default is 0.

**Abbreviation:** NUMSLT

**SCRATCHTHRESHOLD(MEDIA1(*num*) MEDIA2(*num*) MEDIA3(*num*) MEDIA4(*num*))**
identifies the scratch volume message threshold. When the number of scratch volumes in the tape library falls below the scratch threshold, an operator action message requesting that scratch volumes be entered into the tape library is issued to the library's specified console. When the number of scratch volumes exceeds twice the scratch threshold, the message is removed from the console.

When creating a library entry for an automated tape library dataserver, SCRATCHTHRESHOLD can be specified for MEDIA1, MEDIA2, MEDIA3, and MEDIA4. When creating a library entry for a manual tape library dataserver, SCRATCHTHRESHOLD can only be specified for MEDIA1 and MEDIA2.

**MEDIA1(*num*)**
is the threshold number, from 0 to 999999, of Cartridge System Tape scratch volumes. The default is 0.

**MEDIA2(*num*)**
is the threshold number, from 0 to 999999, of Enhanced Capacity System Tape scratch volumes. The default is 0.

**MEDIA3(*num*)**
is the threshold number, from 0 to 999999, of High Performance Cartridge Tape scratch volumes. The default is 0.

**MEDIA4(*num*)**
is the threshold number, from 0 to 999999, of MEDIA4 scratch volumes. MEDIA4 is IBM Extended High Performance Cartridge Tape. The default is 0.

**Abbreviation:** SCRTHR

## CREATE LIBRARYENTRY Examples

The CREATE LIBRARYENTRY command can be used to perform functions shown in the following examples.

### Creating a Tape Library Entry: Example 1

This example creates an entry for a tape library named ATLLIB1.

```
//CREATLIB JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     CREATE LIBRARYENTRY -
            (NAME(ATLLIB1) -
            LIBRARYID(12345) -
            LIBDEVTYPE(3495-L50) -
            LOGICALTYPE(AUTOMATED) -
            NUMBERSLOTS(15000) -
            NUMBEREMPTYSLOTS(1000) -
            NUMBERSCRATCHVOLUMES(MEDIA1(500) MEDIA2(400)) -
            SCRATCHTHRESHOLD(MEDIA1(200) MEDIA2(100)) -
            DESCRIPTION('TEST LIBRARY ATLLIB1') -
            CONSOLENAME(TESTCON)

/*
```

The parameters are:

- LIBRARYENTRY creates a new entry for a tape library.
- NAME names the tape library ATLLIB1.
- LIBRARYID is the tape library's five-digit hexadecimal serial number, 12345.
- LIBDEVTYPE indicates that the tape library device type is 3495-L50.
- LOGICALTYPE specifies that the new tape library is automated.
- NUMBERSLOTS is the total number of slots available in this tape library, 15000.
- NUMBEREMPTYSLOTS is the total number of empty slots currently available, 1000.
- NUMBERSCRATCHVOLUMES is the total number of MEDIA1 scratch volumes (500) and MEDIA2 scratch volumes (400).
- SCRATCHTHRESHOLD is the scratch volume threshold for MEDIA1 tape volumes (200) and MEDIA2 tape volumes is (100). When the number of available scratch volumes decreases to these values, an operator action message is issued to the console.
- DESCRIPTION is the description of the tape library.
- CONSOLENAME shows TESTCON is the console name.

### Creating a LIBRARY Record: Example 2

This example creates a record for LIBRARY ATLLIB1.

```
//CREATLIB JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     CREATE LIBRARYENTRY -
            (NAME(ATLLIB1) -
            LIBRARYID(12345) -
            LOGICALTYPE(AUTOMATED) -
            NUMBERSLOTS(14800) -
            NUMBEREMPTYSLOTS(1000) -
            NUMBERSCRATCHVOLUMES(MEDIA3(500)) -
```

```
SCRATCHTHRESHOLD(MEDIA3(200)) -
DESCRIPTION(TEST LIBRARY ATLLIB1) -
CONSOLENAME(TESTCON)
```

The parameters are:

- LIBRARYENTRY indicates that an entry describing an entire LIBRARY is being created.
- NAME specifies that the name of the LIBRARYENTRY being created is ATLLIB1.
- NUMBERSCRATCHVOLUMES specifies the total number of volumes available as scratch volumes for MEDIA3 to be 500.
- SCRATCHTHRESHOLD specifies that when the number of scratch volumes available for MEDIA3 falls below 200, an operator action message will be issued.

# Chapter 10. CREATE VOLUMEENTRY

The CREATE VOLUMEENTRY command creates tape volume entries. Use this command only to recover from tape volume catalog errors.

Because access method services cannot change the library manager inventory in an automated tape library, ISMF should be used for normal tape library create functions.

The syntax of the CREATE VOLUMEENTRY command is:

| | |
|---|---|
| **CREATE** | **VOLUMEENTRY**<br>**(NAME(***entryname***)**<br>**[CHECKPOINT\|NOCHECKPOINT]**<br>**[COMPACTION{YES\|IDRC\|NO\|NONE\|UNKNOWN}]**<br>**[ENTEREJECTDATE(***eedate***)]**<br>**[EXPIRATIONDATE(***expdate***)]**<br>**[LIBRARYNAME(***libname***)]**<br>**[LOCATION{LIBRARY\|SHELF}]**<br>**[MEDIATYPE{MEDIA1\|MEDIA2\|MEDIA3\|MEDIA4}]**<br>**[MOUNTDATE(***mountdate***)]**<br>**[OWNERINFORMATION(***ownerinfo***)]**<br>**[RECORDING{18TRACK\|36TRACK\|128TRACK\|256TRACK\|UNKNOWN}]**<br>**[SHELFLOCATION(***shelf***)]**<br>**[SPECIALATTRIBUTE{READCOMPATIBLE\|NONE}]**<br>**[STORAGEGROUP(***groupname***)]**<br>**[USEATTRIBUTE{SCRATCH\|PRIVATE}]**<br>**[WRITEDATE(***wrtdate***)]**<br>**[WRITEPROTECT\|NOWRITEPROTECT]**) |

## Required Parameters

**VOLUMEENTRY**
creates a tape volume entry. To create a tape volume entry, you must have access to RACF facility class profile STGADMIN.IGG.LIBRARY.

**Abbreviation:** VOLENTRY|VOLENT

**NAME(***entryname***)**
is the name of the volume entry being created.

*entryname*
consists of a V concatenated with the 1-to-6 character volser. The volser can include only uppercase alphabetics A–Z and numerics 0–9.

## Optional Parameters

**CHECKPOINT|NOCHECKPOINT**
identifies whether the tape volume is a secure checkpoint volume. If you do not use this, the checkpoint status is unknown.

**CHECKPOINT**
indicates a secure checkpoint volume.

**Abbreviation:** CHKPT

© Copyright IBM Corp. 1973, 2002

**NOCHECKPOINT**
indicates a non-secure checkpoint volume.

**Abbreviation:** NOCHKPT

**COMPACTION{YES |IDRC|NO|NONE|UNKNOWN}**
identifies whether the data on the volume is compacted. The YES and IDRC parameter variables are synonymous. The NO and NONE parameter variables are synonymous.

**YES**
specifies that data is compacted in the manner appropriate for the type of media.

**IDRC**
specifies that improved data recording capability (IDRC) compaction was used.

**NO**
specifies that no compaction was used.

**NONE**
specifies that no compaction was used.

**UNKNOWN**
specifies that it is not known if compaction was used.

**Abbreviation:** COMP

**ENTEREJECTDATE(*eedate*)**
is the date that a tape volume was last ejected from a tape library or last entered into a tape library.

*eedate*
is a date, YYYY-MM-DD. See “Tape Library Date Formats” on page 9 for a description of valid date values. The default is blanks.

**Abbreviation:** EEDATE

**EXPIRATIONDATE(*expdate*)**
is the date the tape volume expires. If there is more than one data set on the volume, the expiration date is the latest expiration date. The expiration date is set to blanks when the USEATTRIBUTE parameter is specified as SCRATCH.

*expdate*
is a date, YYYY-MM-DD. See “Tape Library Date Formats” on page 9 for a description of valid date values.

**Abbreviation:** EXDATE

**LIBRARYNAME(*libname*)**
is the name of the tape library where the tape volume resides. If you use this, set LOCATION=LIBRARY. If LOCATION=SHELF, the library name becomes SHELF.

*libname*
is a 1-to-8 character name of a tape library.

**Abbreviation:** LIBNAME

**LOCATION{LIBRARY|SHELF}**
Either the tape volume resides in a tape library, or it resides on a shelf outside the tape library.

- If you set it to LIBRARY, you must also enter a LIBRARYNAME.
- If you set it to SHELF, the library name defaults to SHELF.

**Abbreviation:** LOC

**MEDIATYPE{MEDIA1|MEDIA2|MEDIA3|MEDIA4}**
identifies the media type of the tape volume.

**MEDIA1**
specifies that the tape volume is Cartridge System Tape.

**MEDIA2**
specifies that the tape volume is Enhanced Capacity System Tape. You cannot use this parameter when SPECIALATTRIBUTE is set to READCOMPATIBLE, or RECORDING is set to 18TRACK. MEDIATYPE defaults to MEDIA2.

**MEDIA3**
specifies that the tape volume is High Performance Cartridge Tape.

**MEDIA4**
specifies that the tape volume is IBM Extended High Performance Cartridge Tape.

**Abbreviation:** MTYPE

**MOUNTDATE(*mountdate*)**
identifies the date on which the tape volume was last mounted onto a tape drive and successfully opened.

*mountdate*
is a date, YYYY-MM-DD. See “Tape Library Date Formats” on page 9 for a description of valid date values. The default for this parameter is blanks.

**Abbreviation:** MDATE

**OWNERINFORMATION(*ownerinfo*)**
provides information about the tape volume’s owner.

*ownerinfo*
specifies a 1-to-64 character owner information field. If you use commas, semicolons, embedded blanks, parentheses, or slashes, place the entire description in single quotation marks. The default is blanks.

**Abbreviation:** OWNINFO

**RECORDING{18TRACK|36TRACK|128TRACK|256TRACK|UNKNOWN}**
identifies the recording technique for creating the tape. You can only use this for private tape volumes. Scratch tape volumes default to 36TRACK for MEDIA1 and MEDIA2. Scratch tape volumes default to 128TRACK for MEDIA3 and MEDIA4.

**18TRACK**
Tape was written and must be read on an 18-track device. This parameter is only valid with MEDIATYPE(MEDIA1).

**36TRACK**
Tape was written and must be read on a 36-track device. This parameter is only valid with MEDIATYPE(MEDIA1) or MEDIATYPE(MEDIA2). This parameter cannot be specified with SPECIALATTRIBUTE(READCOMPATIBLE).

**128TRACK**
Tape was written and must be read on a 128-track device. This parameter is only valid with MEDIATYPE(MEDIA3) or MEDIATYPE(MEDIA4). This parameter cannot be specified with SPECIALATTRIBUTE(READCOMPATIBLE).

**256TRACK**
Tape was written and must be read on a 256-track device. This parameter is only valid with MEDIATYPE(MEDIA3) or MEDIATYPE(MEDIA4). If MEDIATYPE(MEDIA3) or MEDIATYPE(MEDIA4) is specified and RECORDING is not specified, then RECORDING will default to 128TRACK as it does now.

**UNKNOWN**
Tape recording technique is unknown.

**Abbreviation:** REC

**SHELFLOCATION(*shelf*)**
identifies the shelf location for a tape volume that resides outside a tape library. This parameter can be included for a library-resident tape volume.

*shelf*
a 1-to-32 character shelf location information field. If you use commas, semicolons, embedded blanks, parentheses, or slashes, enclose the entire description in single quotation marks. The default is blanks.

**Abbreviation:** SHELFLOC

**SPECIALATTRIBUTE{READCOMPATIBLE|NONE}**
shows special attributes of the tape volume. Use this only for private tape volumes. Scratch tape volumes default to NONE.

**READCOMPATIBLE**
On subsequent allocations, the system uses read compatible devices for allocation of this tape volume.

**Abbreviation:** RDCOMPAT

**NONE**
requires no special tape attributes.

**Abbreviation:** SATTR

**STORAGEGROUP(*groupname*)**
Identifies the storage group name.

*groupname*
is the 1-to-8 character name of the storage group in which this tape volume is defined. The default is blanks. If the USEATTRIBUTE=SCRATCH, however, the storage group name defaults to *SCRTCH*.

**Abbreviation:** STORGRP

**USEATTRIBUTE{SCRATCH|PRIVATE}**
can be SCRATCH for scratch volumes or PRIVATE for private volumes. If you use SCRATCH, the storage group name is set to *SCRTCH*, and the expiration date is set to blanks.

**Abbreviation:** UATTR

**WRITEDATE(*wrtdate*)**
identifies the date that a data set on a tape volume was last opened for writing.

*wrtdate*
is a date, YYYY-MM-DD. See “Tape Library Date Formats” on page 9 for a description of valid date values. The default for this parameter is blanks.

**Abbreviation:** WDATE

**WRITEPROTECT|NOWRITEPROTECT**
identifies whether the tape volume is write protected or not. If you do not use this, write protect status is unknown.

**WRITEPROTECT**
indicates that the tape volume is write protected.

**Abbreviation:** WPRT

Setting **WRITEPROTECT** in the tape volume entry does not automatically write protect your volume. It is an informational setting that is recorded when software detects that the volume is write protected and is not used by software when determining whether the volume is protected. For a volume to be write protected, you will still need to set the write protect tab available on the physical tape volume. Since the availability to write protect a volume does not exist for logical volumes in a VTS, (no tab available either physically or logically), you can alternately use RACF or PROTECT=ALL in your JCL to protect the volume during usage.

**NOWRITEPROTECT**
indicates that the tape volume is not write protected.

**Abbreviation:** NWPRT

# CREATE VOLUMEENTRY Examples

## Creating a Tape Volume Entry: Example 1

This example creates a tape library entry for a tape volume with volser AL0001.

```
//CREATVOL JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     CREATE VOLUMEENTRY -
            (NAME(VAL0001) -
            LIBRARYNAME(ATLLIB1) -
            STORAGEGROUP(*SCRTCH*) -
            USEATTRIBUTE(SCRATCH) -
            NOWRITEPROTECT -
            LOCATION(LIBRARY) -
            SHELFLOCATION(10098SHELF) -
            OWNERINFORMATION('JOHN SMITH,RMKD222') -
            ENTEREJECTDATE(1990-03-18) -
            EXPIRATIONDATE(2000-12-31) -
            WRITEDATE(1991-01-02) -
            MOUNTDATE(1991-01-02))
/*
```

The parameters are:

- VOLUMEENTRY creates a tape volume entry in a tape library.
- NAME names the tape volume entry, VAL0001 (‘V’ concatenated with volser AL0001).

- LIBRARYNAME adds this tape volume to the tape library named ATLLIB1.
- STORAGEGROUP names the storage group *SCRTCH* (default name when USEATTRIBUTE=SCRATCH).
- USEATTRIBUTE specifies the tape volume as SCRATCH.
- NOWRITEPROTECT identifies the tape volume as not write protected.
- LOCATION specifies that the tape volume will reside in the tape library.
- SHELFLOCATION gives 10098SHELF as the shelf location.
- OWNERINFORMATION gives JOHN SMITH,RMKD222 for owner information.
- ENTEREJECTDATE is the date on which the tape volume was last entered into, or ejected from, the tape library named ATLLIB1.
- EXPIRATIONDATE is the date on which the tape volume expires.
- WRITEDATE is the date when the tape volume was last written to.
- MOUNTDATE is the date when the tape volume was last mounted onto a tape drive.

## Creating a VOLUME Entry: Example 2

This example creates an entry for volume AL0001.

```
//CREATVOL JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     CREATE VOLUMEENTRY -
            (NAME(VAL0001) -
            LIBRARYNAME(ATLLIB1) -
            USEATTRIBUTE(PRIVATE) -
            MEDIATYPE(MEDIA3) -
            RECORDING(128TRACK))
```

The parameters are:

- VOLUMEENTRY indicates that an entry describing a single volume in a LIBRARY is being created.
- NAME specifies that the name of the VOLUMEENTRY entry being created is VAL0001 and the volser is AL0001.
- LIBRARYNAME specifies that the name of the LIBRARY this VOLUME record is associated with is ATLLIB1.
- USEATTRIBUTE identifies the volume as being a PRIVATE tape.
- MEDIATYPE specifies the media type MEDIA3.
- RECORDING specifies the recording technique 128TRACK.

# Chapter 11. DCOLLECT

The DFSMS Data Collection Facility (DCOLLECT) is a function of access method services. DCOLLECT collects data in a sequential file you can use as input to other programs or applications.

DCOLLECT obtains data on:

- **Active Data Sets**

  DCOLLECT provides data about space use and data set attributes and indicators on the selected volumes and storage groups.
- **VSAM Data Set Information**

  DCOLLECT provides specific information relating to VSAM data sets residing on the selected volumes and storage groups.
- **Volumes**

  DCOLLECT provides statistics and information on volumes that are selected for collection.
- **Inactive Data**

  DCOLLECT produces output for DFSMShsm-managed data, (inactive data management), which includes both migrated and backed up data sets.
  - **Migrated Data Sets**: DCOLLECT provides information on space utilization and data set attributes for data sets migrated by DFSMShsm.
  - **Backed Up Data Sets**: DCOLLECT provides information on space utilization and data set attributes for every version of a data set backed up by DFSMShsm.
- **Capacity Planning**

  Capacity planning for DFSMShsm-managed data (inactive data management) includes the collection of both DASD and tape capacity planning.
  - **DASD Capacity Planning**: DCOLLECT provides information and statistics for volumes managed by DFSMShsm (ML0 and ML1).
  - **Tape Capacity Planning**: DCOLLECT provides statistics for tapes managed by DFSMShsm.
- **SMS Configuration Information**

  DCOLLECT provides information about the SMS configurations. The information can be from either an active control data set (ACDS) or a source control data set (SCDS), or the active configuration.

  DCOLLECT provides attributes that are in the selected configuration for the following:
  - Data Class Constructs
  - Storage Class Constructs
  - Management Class Constructs
  - Storage Group Constructs
  - SMS Volume Information
  - SMS Base Configuration Information
  - Aggregate Group Construct Information
  - Optical Drive Information
  - Optical Library Information
  - Cache Names
  - Accounting Information for the ACS

© Copyright IBM Corp. 1973, 2002

For information on calling DCOLLECT from ISMF, see *z/OS DFSMSdfp Storage Administration Reference* . For information on using DCOLLECT to monitor space usage, see *MVS/ESA SML: Managing Data* .

The syntax of the DCOLLECT command is:

| | |
|---|---|
| **DCOLLECT** | **{OUTFILE(***ddname***)\|**<br>**OUTDATASET(***entryname***)}**<br>**{[VOLUMES(***volser***[** *volser***...])]**<br>**[BACKUPDATA]**<br>**[CAPPLANDATA]**<br>**[EXCLUDEVOLUMES(***volser***[** *volser***...])]**<br>**[MIGRATEDATA]**<br>**[SMSDATA(SCDSNAME(***entryname***)\|ACTIVE)]**<br>**[STORAGEGROUP(***sgname***[** *sgname***...])]}**<br>**[ERRORLIMIT(***value***)]**<br>**[EXITNAME(***entrypoint***)]**<br>**[MIGRSNAPALL\|MIGRSNAPERR]**<br>**[NODATAINFO]**<br>**[NOVOLUMEINFO]**<br>**[REPLACE\|APPEND]** |

DCOLLECT can be abbreviated: DCOL

**Exception:** Although BACKUPDATA, CAPPLANDATA, MIGRATEDATA, STORAGEGROUP, VOLUMES, and SMSDATA are designated as optional parameters, you must use at least one of them. You can use any combination of these parameters; use at least one.

**DCOLLECT User Exit:**
For a description of the DCOLLECT User Exit, refer to “Appendix F. DCOLLECT User Exit” on page 405.

**DCOLLECT Output**
For an explanation of how to interpret DCOLLECT output, see “Appendix G. Interpreting DCOLLECT Output” on page 411.

**DCOLLECT Security Considerations**
*APF Authorization*: For information on a program calling DCOLLECT see “Appendix D. Invoking Access Method Services from Your Program” on page 393.

**DCOLLECT HFS Considerations**
The information in DCOLLECT type ’D’ records for HFS files are not all contained in the VTOC, VVDS or catalog. You must define an appropriate OMVS segment in order to gather all statistics.

*RACF Authorization*: To control access to the DCOLLECT function, a RACF check for authorization is made for a facility class profile of STGADMIN.IDC.DCOLLECT. If this profile exists, then read authority is necessary. The command will not be successful if the user is not authorized.

## DCOLLECT Parameters

The DCOLLECT command uses the following parameters.

## Required Parameters

**OUTDATASET(*entryname*)**
identifies the target data set. You must use a physical sequential data set with a record format of V or VB. Use an LRECL that is *at least* the size of the longest DCOLLECT record to be collected. Changes to the JCL are not necessary if you use an LRECL larger than the longest record to be collected. The LRECL should be at least as large as the longest record DCOLLECT generates but not larger than 32756. A mid-range value is appropriate.

If you use OUTDATASET, the entryname is dynamically allocated with a status of either OLD or MOD, as required by the REPLACE parameter.

**Abbreviation:** ODS

**OUTFILE(*ddname*)**
enter the name of a DD statement that identifies the target data set.

**Abbreviation:** OFILE

## Optional Parameters

**BACKUPDATA**
requires that information on backed up data sets is collected from the given backup control data set (BCDS).

The desired BCDS must be allocated to the ddname BCDS.

**Abbreviation:** BACD

**CAPPLANDATA**
includes capacity planning information in the output data set. Allocate the MCDS to the ddname MCDS and the BCDS to the ddname BCDS.

**Abbreviation:** CAPD

**ERRORLIMIT(*value*)**
is the maximum number of errors for which detailed DCOLLECT error messages can print during program run. ERRORLIMIT prevents runaway message output. The default for ERRORLIMIT is 2,147,483,647 errors, but any number between 1 and 2,147,483,647 can be given. Processing continues even though the error limit has been reached.

**Abbreviation:** ELIMIT

**EXCLUDEVOLUMES(*volser*[ *volser*...])**
allows you to exclude information on a selected volume or group of volumes. One or more volumes selected by using the STORAGEGROUP and VOLUMES keywords can be excluded with this keyword. Options for EXCLUDEVOLUMES are:

- a fully specified volume serial number, containing 1-to-6 characters
- a partially specified volume serial number using a single trailing asterisk as a placeholder for all remaining characters, or
- any combination of the above.

**Abbreviation:** EXV

**EXITNAME(*entrypoint*)**
is the 1-to-8 character entrypoint name for an external DCOLLECT user exit module. Load it to an APF-authorized library for access at the time of DCOLLECT invocation. If you do not use it, the default DCOLLECT user exit, IDCDCX1, is used.

**Abbreviation:** EXIT

**MIGRATEDATA**
requires collection of information on migrated data sets from the specified MCDS (Migration Control Data Set). The desired MCDS must be allocated to the ddname MCDS.

**Abbreviation:** MIGD

**MIGRSNAPALL**
asks ARCUTIL to do SNAP processing, and is used for diagnostic reasons only. See *z/OS DFSMShsm Implementation and Customization Guide* for more information on SNAP ALL processing. Do not use it with MIGRSNAPERR. It is ignored if you do not use MIGRATEDATA, BACKUPDATA, or CAPPLANDATA.

**Abbreviation:** MSALL

**MIGRSNAPERR**
requires ARCUTIL to run SNAP processing when an error occurs during ARCUTIL processing. Use it for diagnostic purposes only. See *z/OS DFSMShsm Implementation and Customization Guide* for more information on SNAP ALL processing. Do not use it with MIGRSNAPALL. It is ignored if you do not use MIGRATEDATA, BACKUPDATA, or CAPPLANDATA.

**Abbreviation:** MSERR

**NODATAINFO**
says that no data set information records are generated or written to the output data set. Use this parameter if you want only volume information generated for the given volumes or storage groups.

**Abbreviation:** NOD

**NOVOLUMEINFO**
says that no volume information records are generated or written to the output data set. Use this parameter if you want only data set information generated for the given volumes or storage groups.

**Abbreviation:** NOV

**REPLACE|APPEND**
specifies whether the output data is to replace existing data or whether the output data is to be added to the end of the existing data set. The REPLACE/APPEND applies when OUTDATASET is used. If you use OUTFILE, data set processing is controlled by the JCL DISP parameter: OLD replaces the current contents of the data set, and MOD appends new records to the end of the data set.

**REPLACE**
asks that the contents of the output data set are overwritten with new data. All existing data in the output data set is lost when this parameter is selected.

**Abbreviation:** REPL

**APPEND**
writes new records starting at the end of the existing data, if any exists. All existing data is preserved when this parameter is selected.

**Abbreviation:** APP

**SMSDATA(SCDSNAME(*entryname*)|ACTIVE)**
includes SMS configuration data in the DCOLLECT output data set. This parameter can include either an SCDS name or the keyword ACTIVE.

One or more of the following record types is created when you use **SMSDATA**:

| Type | Description |
|---|---|
| **DC** | Data Class construct information |
| **SC** | Storage Class construct information |
| **MC** | Management Class construct information |
| **BC** | Base Configuration information |
| **SG** | Storage Group construct information |
| **VL** | Storage Group volume information |
| **AG** | Aggregate Group information |
| **DR** | OAM Drive Record information |
| **LB** | OAM Library Record information |
| **CN** | Cache Names from the Base Configuration Information |
| **AI** | Accounting Information for the ACS |

**Abbreviation: SMS**

The subparameters of **SMSDATA** are:

**SCDSNAME(***entryname***)**
is the source of the SMS control data that is to be collected.

*entryname*
is used to specify the name of an existing cataloged SCDS. An enqueue with a major name of IGDCDS is issued to serialize access to the control data set. The enqueue is held for the duration of **SMSDATA** processing.

**Abbreviation: SCDS**

**ACTIVE**
takes the SMS information from the configuration that is currently active on the system.

**STORAGEGROUP(***sgname***[** *sgname***...])**
lists the storage groups from which information is to be collected. For each storage group listed, a list of online volume serials is generated. Information is collected for all data sets residing on those volumes unless you use NODATAINFO. Volume information is collected unless you use NOVOLUMEINFO. A maximum of 255 storage groups can be selected. Although several storage groups can be specified, and the volume list might have duplicates, each volume's information is only processed once.

**Abbreviation:** STOG

**VOLUMES(***volser***[** *volser***...])**
lists the volumes from which information is to be collected. For each online volume serial listed (or resolved from generic specifications), information is collected for all data sets residing on those volumes unless you use NODATAINFO. Volume information is collected unless you use NOVOLUMEINFO. You can use a maximum of 255 volume serials.

Options are:

- A fully specified volume serial number, containing 1-to-6 characters.
- A partially specified volume serial number using a single asterisk as a place holder for all remaining characters.
- Six asterisks to indicate the system residence volume (SYSRES).
- Any combination of the above.

For example, you might use one of these for the volume serial number:

**SYS001** This collects data from volume SYS001 only.
**SYS*** This collects data from all online volumes beginning with SYS.

This collects data from the system residence volume (SYSRES).

This collects data from all online volumes.

Although the same volumes can be specified several times, each volume's information is only processed once.

**Abbreviation:** VOL

# DCOLLECT in a Batch Environment

The following JCL examples illustrate how to use the DCOLLECT function in a batch environment.

## Generic Volume Data Collection: Example 1

In this example, a partially specified volume serial number is provided which causes data collection to occur from all on line volumes beginning with that generic name.

```
//COLLECT1 JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//OUTDS    DD     DSN=USER.DCOLLECT.OUTPUT,
//       STORCLAS=LARGE,
//       DSORG=PS,
//       DCB=(RECFM=VB,LRECL=644,BLKSIZE=0),
//       SPACE=(1,(100,100)),AVGREC=K,
//       DISP=(NEW,CATLG,KEEP)
//SYSIN    DD     *
     DCOLLECT -
           OFILE(OUTDS) -
           VOLUME(SYS1*)
/*
```

Job control language statements:

- The DD statement OUTDS describes the sequential output data set where records from data collection is written.

Parameters are:

- OFILE identifies the output data set (USER.DCOLLECT.OUTPUT) by ddname.
- VOLUME names the volumes for which data is to be collected. In this example the generic specification collects data for all on line volumes that begin with the characters SYS1.

## Storage Group Data Collection: Example 2

In this example a storage a group name is specified which causes data to be collected from all on line volumes belonging to that storage group.

```
//COLLECT2 JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//OUTDS    DD     DSN=USER.DCOLLECT.OUTPUT,
//       STORCLAS=LARGE,
//       DSORG=PS,
//       DCB=(RECFM=VB,LRECL=644,BLKSIZE=0),
//       SPACE=(1,(100,100)),AVGREC=K,
//       DISP=(NEW,CATLG,KEEP)
```

```
//SYSIN    DD     *
     DCOLLECT -
           OFILE(OUTDS) -
           STORAGEGROUP(STGGP001) -
           NODATAINFO
/*
```

Job control language statements:

- OUTDS describes the sequential output data set where records from data collection is written.

The DCOLLECT command defines which data is to be collected.

Parameters are:

- OFILE identifies the output data set (USER.DCOLLECT.OUTPUT) by ddname.
- STORAGEGROUP names the storage group from which data is to be collected. Data is collected from all on line volumes that reside in storage group STGGP001.
- NODATAINFO says that only volume information records are created and written to the output data set. No data set information is collected and written to the output data set.

## Migrated and Backup Data Set Data Collection: Example 3

This example shows data collection for all migrated and backup data sets that reside on the system.

```
//COLLECT3 JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//MCDS     DD     DSN=HSM.MCDS,DISP=SHR
//BCDS     DD     DSN=HSM.BCDS,DISP=SHR
//SYSIN    DD     *
     DCOLLECT -
           OUTDATASET(USER.DCOLLECT.OUTPUT) -
           MIGRATEDATA -
           BACKUPDATA
/*
```

Job control language statements:

- MCDS identifies the Migration Control Data Set. This data set must be identified by the ddname MCDS. When using a multicluster CDS, each cluster must be identified on a separate DD statement. The ddnames are MCDS, MCDS2, MCDS3, and MCDS4.
- BCDS identifies the Backup Control Data Set. This data set must be identified by the ddname BCDS. When using a multicluster CDS, each cluster must be identified on a separate DD statement. The ddnames are BCDS, BCDS2, BCDS3, and BCDS4.

The DCOLLECT command defines which data is to be collected.

The parameters are:

- OUTDATASET names the output data set USER.DCOLLECT.OUTPUT. Which must exist before the job is run. All new data records are appended to the end of the data set.
- MIGRATEDATA creates data records for all migrated data sets that reside on this system.
- BACKUPDATA creates data records for all backed up data sets on this system.

## Combination of Options: Example 4

In this example, four different volume serial numbers and four different storage group names are used. Information is collected from migrated data sets and capacity planning information is retrieved.

```
//COLLECT4 JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//MCDS     DD     DSN=HSM.MCDS,DISP=SHR
//BCDS     DD     DSN=HSM.BCDS,DISP=SHR
//OUTDS    DD     DSN=USER.DCOLLECT.OUTPUT,
//       STORCLAS=LARGE,
//       DSORG=PS,
//       DCB=(RECFM=VB,LRECL=644,BLKSIZE=0),
//       SPACE=(1,(10,10)),AVGREC=M,
//       DISP=(NEW,CATLG,KEEP)
//SYSIN    DD     *
     DCOL -
           OFILE(OUTDS) -
           VOL(SYS100, SYS101, SYS200, SYS201) -
           STOG(STGGP100, STGGP101, STGGP200, STGGP201) -
           MIGD -
           CAPD
/*
```

Job control language statement:

- The DD statement OUTDS describes the sequential output data set where records from data collection is written.

The DCOLLECT command defines which data is to be collected.

Parameters are:

- OFILE identifies the output data set (USER.DCOLLECT.OUTPUT) by ddname.
- VOL names the volume from which data is to be collected. In this example, VOL is used to collect data for the on line volumes SYS100, SYS101, SYS200 and SYS201.
- STOG names the storage group from which data is to be collected. In this example, STOG is used to collected data from all online volumes that reside in storage groups STGGP100, STGGP101, STGGP200 and STGGP201.
- MIGD creates data records for all migrated data sets that reside on this system.
- CAPD includes capacity planning information in the output data set.

## Collection of SMS Construct Information: Example 5

This example uses the SMSDATA keyword to extract the construct definitions from a named SCDS.

```
//COLLECT5 JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//OUTDS    DD     DSN=USER.DCOLLECT.OUTPUT,
//       STORCLAS=LARGE,
//       DSORG=PS,
//       DCB=(RECFM=VB,LRECL=32756,BLKSIZE=0),
//       SPACE=(1,(10,10)),AVGREC=K,
//       DISP=(NEW,CATLG,KEEP)
//SYSIN    DD     *
     DCOL -
           OFILE(OUTDS) -
           SMSDATA(SCDSNAME(SYSPROG.SCDS.SYSTEMA))
/*
```

Job control language statement:

- OUTDS describes the sequential output data set where records from data collection are written. The LRECL is set to 32756, which is the largest record size that can be handled by DCOLLECT. You do not need to change the JCL each time a DCOLLECT record is extended.

The DCOLLECT command defines which data is to be collected.

Parameters are:

- OFILE identifies the output data set (’USER.DCOLLECT.OUTPUT’) by ddname.
- SMSDATA collects construct data from the named SCDS. In this example, the SCDS is named SYSPROG.SCDS.SYSTEMA.

# Chapter 12. DEFINE ALIAS

The DEFINE ALIAS command defines an alternate name for a non-VSAM data set or a user catalog. The syntax of the DEFINE ALIAS command is:

| | |
|---|---|
| **DEFINE** | **ALIAS**<br>**(NAME(***aliasname***)**<br>**RELATE(***entryname***))**<br>**SYMBOLICRELATE(***entryname***))**<br>**[CATALOG(***catname***)]** |

DEFINE can be abbreviated: DEF

## DEFINE ALIAS Parameters

The DEFINE ALIAS command uses the following parameters.

## Required Parameters

**ALIAS**

Defines an alias for a user catalog or non-VSAM data set.

If the *entryname* in the RELATE parameter is non-VSAM, choose an *aliasname* in the NAME parameter. This is done to ensure the multilevel alias facility selects the catalog that has the *entryname*.

The multilevel alias facility and the system-generated name format requires special attention:

- When you DEFINE a VSAM data set, point the data/index name to the same catalog as the cluster; otherwise, an error occurs.
- During the DEFINE of a VSAM cluster or a generation data group (GDG), if the name of the cluster or GDG matches an existing alias or user catalog, the DEFINE request is denied with a duplicate name error. This prevents the data/index component or a generation data set (GDS) from becoming inaccessible.
- When you add an alias to the catalog, ensure it does not cause existing data sets to become inaccessible.

For more details, see *z/OS DFSMS: Managing Catalogs*.

**NAME(***aliasname***)**

Is the alias (the alternate entryname) for a user catalog or non-VSAM data set. An alias must be unique within a catalog.

**RELATE(***entryname***)**

Is the name of the entry (the user catalog entryname or the non-VSAM data set name) for which the alias is defined.

**Abbreviation:** REL

**Restriction:** The RELATE and SYMBOLICRELATE parameters are mutually exclusive and cannot be specified at the same time. The resolved '*entryname*' must be a catalog entry that is located in the same catalog that contains '*aliasname*'.

© Copyright IBM Corp. 1973, 2002

**SYMBOLICRELATE(*entryname*)**
Allows the specification of the base data set name using system symbols. For more details, see ″Extended Alias Support″ in *z/OS DFSMS: Managing Catalogs*.

**Abbreviation:** SYM

## Optional Parameters

**CATALOG(*catname*)**
Identifies the catalog in which the alias is defined. If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved. See "Catalog Selection Order for DEFINE" on page 10 for the order in which a catalog is selected when the catalog's name is not given.

*catname*
Names of catalog. When the alias is for a user catalog connector, *catname* is the name of the master catalog.

**Abbreviation:** CAT

# DEFINE ALIAS Examples

## Define Alias for a non-VSAM non-SMS-Managed Data Set: Example 1

This example defines an alias for a non-VSAM data set:

```
//DEFALS   JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE ALIAS -
          (NAME(EXAMPLE.NONVSAM1) -
          RELATE(EXAMPLE.NONVSAM) ) -
          CATALOG(USERCAT4)
/*
```

The DEFINE ALIAS command defines an alias, EXAMPLE.NONVSAM1, for the non-VSAM data set EXAMPLE.NONVSAM.

The parameters are:

- NAME—the alias (alternate entryname), EXAMPLE.NONVSAM1.
- RELATE—the entryname, EXAMPLE.NONVSAM, for which the alias is an alternate entryname.
- CATALOG—the name of the user catalog.

## Define an Alias for a User Catalog: Example 2

In this example, an alias is defined for a user catalog. The alias is defined in the master catalog.

```
//DEFUCALS  JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
     DEFINE ALIAS -
           (NAME(RST) -
            RELATE(VWXUCAT1)) -
            CATALOG(AMAST1)
/*
```

The DEFINE ALIAS command defines an alias, RST, for the user catalog, VWXUCAT1. VSAM locates any data set defined with a first-level qualifier of RST in user catalog VWXUCAT1 when an access method services command or user program references the data set.

The parameters are:

- NAME—the alias, RST.
- RELATE—the name of the user catalog, VWXUCAT1, for which RST is an alternate entryname.
- CATALOG—the name of the master catalog.

# Chapter 13. DEFINE ALTERNATEINDEX

The DEFINE ALTERNATEINDEX command defines an alternate index. Use it to show attributes for the alternate index as a whole and for the components of the alternate index. The syntax of the DEFINE ALTERNATIVEINDEX command is:

**DEFINE ALTERNATEINDEX** (parameters) -
[**DATA**(parameters)] -
[**INDEX**(parameters)] -
[**CATALOG**(subparameters)]

| | |
|---|---|
| **DEFINE** | **ALTERNATEINDEX**<br>**(NAME(***entryname***)**<br>**RELATE(***entryname***)**<br>**{CYLINDERS(***primary***[** *secondary***])\|**<br>**KILOBYTES(***primary***[** *secondary***])\|**<br>**MEGABYTES(***primary***[** *secondary***])\|**<br>**RECORDS(***primary***[** *secondary***])\|**<br>**TRACKS(***primary***[** *secondary***])}**<br>**VOLUMES(***volser***[** *volser***...])**<br>**[BUFFERSPACE(***size***)]**<br>**[CONTROLINTERVALSIZE(***size***)]**<br>**[DATACLASS(***class***)]**<br>**[ERASE\|NOERASE]**<br>**[EXCEPTIONEXIT(***entrypoint***)]**<br>**[FILE(***ddname***)]**<br>**[FREESPACE(***CI-percent***[** *CA-percent***]\| 0 0)]**<br>**[KEYS(***length* *offset***\|64 0)]**<br>**[MODEL(***entryname***[** *catname***])]**<br>**[OWNER(***ownerid***)]**<br>**[RECATALOG\|NORECATALOG]**<br>**[RECORDSIZE(***average* *maximum***\| 4086 32600)]**<br>**[REUSE\|NOREUSE]**<br>**[SHAREOPTIONS(***crossregion***[** *crosssystem***]\| 1 3)]**<br>**[SPEED\|RECOVERY]**<br>**[TO(***date***)\|FOR(***days***)]**<br>**[UNIQUEKEY\|NONUNIQUEKEY]**<br>**[UPGRADE\|NOUPGRADE]**<br>**[WRITECHECK\|NOWRITECHECK])**<br>**[DATA** (<br>**{CYLINDERS(***primary***[** *secondary***])\|**<br>**KILOBYTES(***primary***[** *secondary***])\|**<br>**MEGABYTES(***primary***[** *secondary***])\|**<br>**RECORDS(***primary***[** *secondary***])\|**<br>**TRACKS(***primary***[** *secondary***])}**<br>**[VOLUMES(***volser***[** *volser***...])]**<br>**[ATTEMPTS(***number***)]**<br>**[AUTHORIZATION(***entrypoint***[** *string***])]**<br>**[BUFFERSPACE(***size***)]**<br>**[CODE(***code***)]**<br>**[CONTROLINTERVALSIZE(***size***)]**<br>**[ERASE\|NOERASE]**<br>**[EXCEPTIONEXIT(***entrypoint***)]**<br>**[FILE(***ddname***)]**<br>**[FREESPACE(***CI-percent***[** *CA-percent***])]** |

© Copyright IBM Corp. 1973, 2002

```
  [KEYS(length  offset)]
  [MODEL(entryname [ catname ])]
  [NAME(entryname)]
  [OWNER(ownerid)]
  [RECORDSIZE(average  maximum)]
  [REUSE|NOREUSE]
  [SHAREOPTIONS(crossregion[ crosssystem])]
  [SPEED|RECOVERY]
  [UNIQUEKEY|NONUNIQUEKEY]
  [WRITECHECK|NOWRITECHECK])]
[INDEX (
  {CYLINDERS(primary[ secondary])|
   KILOBYTES(primary[ secondary])|
   MEGABYTES(primary[ secondary])|
   RECORDS(primary[ secondary])|
   TRACKS(primary[ secondary])}
  [VOLUMES(volser[ volser...])]
  [ATTEMPTS(number)]
  [AUTHORIZATION(entrypoint[ string])]
  [CODE(code)]
  [CONTROLINTERVALSIZE(size)]
  [EXCEPTIONEXIT(entrypoint)]
  [FILE(ddname)]
  [MODEL(entryname[ catname ])]
  [NAME(entryname)]
  [OWNER(ownerid)]
  [REUSE|NOREUSE]
  [SHAREOPTIONS(crossregion[ crosssystem])]
  [WRITECHECK|NOWRITECHECK])]
[CATALOG(catname)]
```

DEFINE can be abbreviated: DEF

**Restriction**: If IMBED, KEYRANGE, ORDERED, or REPLICATE is specified, it is ignored.

# DEFINE ALTERNATEINDEX Parameters

## Required Parameters

**ALTERNATEINDEX**
defines an alternate index or recatalogs an alternate index entry.

The ALTERNATEINDEX keyword is followed by the parameters for the alternate index as a whole. These parameters are enclosed in parentheses and, optionally, are followed by parameters given separately for the DATA and INDEX components.

**Abbreviation:** AIX

**NAME(*entryname*)**
The alternate index's entryname or the name of each of its components. The entry name specified for the alternate index as a whole is not propagated to the alternate index's components.

You can define a separate entry name for the alternate index, its data component, and its index component. If you do not give a name for the data or index component, one is generated. For more information about the system-generated name format, see *z/OS DFSMS: Managing Catalogs* .

When the alternate index, data component, and index component are individually named, each can be addressed.

**RELATE(***entryname***)**
Names the alternate index base cluster. The base cluster is an entry-sequenced cluster or a key-sequenced cluster to which the alternate index is to be related. You cannot relate an alternate index to a reusable cluster, to a relative record cluster, to an extended addressability ESDS, or to a VVDS (data set name 'SYS1.VVDS.Vvolser'). An SMS-managed alternate index has the same management class and storage class as its base cluster.

Select the *entryname* so that the multilevel alias facility selects the same catalog as the one containing the related data set name.

**Abbreviation:** REL

**CYLINDERS(***primary***[** *secondary***])|**

**KILOBYTES(***primary***[** *secondary***])|**

**MEGABYTES(***primary***[** *secondary***])|**

**RECORDS(***primary***[** *secondary***])|**

**TRACKS(***primary***[** *secondary***])**
The amount of space in cylinders, kilobytes, megabytes, records, or tracks allocated to the alternate index from the volume's available space. A kilobyte and megabyte allocation resolves to either tracks or cylinders; records are allocated to the nearest track boundary.

**Exception:** If allocation resolves to tracks, the space is contiguous. See "Optimizing Control Area Size" in *z/OS DFSMS: Using Data Sets* for more information.

Requests for space are directed to DADSM and result in a format-1 DSCB for the data and index component entries.

If you do not use the MODEL parameter or the RECATALOG parameter, you must include one, and only one, of these parameters: CYLINDERS, KILOBYTES, MEGABYTES, RECORDS, or TRACKS.

The space parameter is optional if the cluster is SMS-managed, but if you do not use it space can be modeled or defaulted by SMS. If it is not determined, the DEFINE is unsuccessful.

To maintain device independence, do not use the TRACKS or CYLINDERS parameters. If you do not use TRACKS or CYLINDERS for an SMS-managed alternate index, space is allocated on the volume selected by SMS.

When you do not divide the data component into key ranges, and more than one volume is given, the primary amount of space is allocated only on the first volume when the component is defined. When the component increases to extend to additional volumes, the first allocation on each overflow volume is the primary amount.

Secondary amounts can be allocated on all volumes available to contain parts of the alternate index, regardless of the key ranges when the alternate index is extended.

You can include the amount of space as a parameter of ALTERNATEINDEX, as a parameter of DATA, or as a parameter of both DATA and INDEX.

If the space is specified as a parameter of:

- ALTERNATEINDEX, the amount specified is divided between the data and index components. The division algorithm is a function of control interval size, record size, device type, and other data set attributes.

  If the division results in an allocation for the data component that is not an integral multiple of the required control area size, the data component's allocation is rounded up to the next higher control area multiple. This rounding can result in a larger total allocation for your alternate index than what you specified.
- DATA, the entire amount given is allocated to the data component. An additional amount of space, depending on control interval size, record size, device type, and other data set attributes, is allocated to the index component.

To determine the exact amount of space allocated to each component, list the alternate index's catalog entry, using the LISTCAT command.

The primary and each secondary allocation must be able to be satisfied within five extents; otherwise, your DEFINE or data set extension is unsuccessful.

You can use these keywords for both SMS and non-SMS-managed data sets.

*primary*
: allocates the initial amount of space to the alternate index.

*secondary*
: Allocates the amount of space each time the alternate index extends, as a secondary extent. If the secondary space allocation is greater than 4.0 gigabytes, it is reduced to an amount as close to 4.0 GB as possible, without going over. This is not true for extended addressability datasets, which have no such space limitation. When you use secondary, space for the alternate index's data and index components can be expanded to a maximum of 123 extents.

**Abbreviations:** CYL, KB, MB, REC, and TRK

**VOLUMES(***volser***[** *volser***...])**
: specifies the volumes on which an alternate index's components are to have space. This parameter is not required if the cluster is modeled or if the cluster is SMS-managed. You can specify VOLUMES for SMS-managed data sets; however, the volumes specified might not be used and, in some cases, can result in an error.

  For SMS-managed data sets, you can use up to 59 volumes. If the combined number of volumes for a cluster and its associated alternate indexes exceeds 59, unpredictable results can occur.

  You can let SMS choose the volumes for SMS-managed data sets by coding an * for the volser with the VOLUMES parameter. If both user-specified and SMS-specified volumes are requested, the user-specified volser must be input first in the command syntax. The default is one volume.

If you do not use the MODEL parameter, VOLUMES must be placed as a parameter of ALTERNATEINDEX, or as a parameter of both DATA and INDEX.

If the data and index components are to reside on different device types, you must include VOLUMES as a parameter of both DATA and INDEX. If more than one volume is listed with a single VOLUMES parameter, the volumes must be the same device type.

You can repeat a volume serial number in the list only if you use the KEYRANGE parameter. This can place more than one key range on the same volume. However, repetition is valid only if all duplicate occurrences are used for the primary allocation of some key range.

The VOLUMES parameter interacts with other DEFINE ALTERNATEINDEX parameters. Ensure that the volumes you define for the alternate index are consistent with the alternate index's other attributes:

- CYLINDERS, RECORDS, TRACKS: The volumes contain enough available space to satisfy the component's primary space requirement.
- FILE: To define an alternate index, the volume information supplied with the DD statement pointed to by FILE must be consistent with the information listed for the alternate index and its components.

**Abbreviation:** VOL

## Optional Parameters

The Optional command uses the following parameters.

**BUFFERSPACE(*size*)**
provides the minimum space for buffers. VSAM determines the data component's and index component's control interval size. If you do not use BUFFERSPACE, VSAM provides enough space to contain two data component control intervals and, if the data is key-sequenced, one index component control interval.

*size*
is the buffer of space. You can use decimal (n), hexadecimal (X'n'), or binary (B'n') not to exceed 16,776,704. The size cannot be less than enough space to contain two data component control intervals and, if the data is key sequenced, one index control interval.

If the buffer size is less than VSAM requires to run your job, it will be treated as though the parameter was not specified and the buffer size will be set to the default value.

**Exception:** When you use RLS access, BUFFERSPACE is ignored.

**Abbreviations:** BUFSP or BUFSPC

**CATALOG(*catname*)**
Identifies the catalog in which the alternate index is defined. The catalog also contains the base cluster's entry (see the RELATE parameter above). See "Catalog Selection Order for DEFINE" on page 10 for the order in which a catalog is selected if the catalog's name is not specified.

To assign catalog names for SMS-managed data sets, you must have access to the RACF STGADMIN.IGG.DIRCAT facility class. See "Storage Management Subsystem (SMS) Considerations" on page 2 for more information.

*catname*
Names the catalog.

**Abbreviation:** CAT

If the catalog's volume is physically mounted, it is dynamically allocated. Mount the volume as permanently resident or reserved.

**CONTROLINTERVALSIZE(*size*)**
Defines the size of the alternate index's control intervals. This depends on the maximum size of data records, and on the amount of buffer space given.

LSR/GSR buffering technique users can ensure buffer pool selection by explicitly defining data and index control interval sizes.

When you do not specify the control interval size, VSAM determines the control interval size. If you have not specified BUFFERSPACE and the size of your records permits, VSAM selects the optimum size for the data control interval size and 512 bytes for the index control interval size.

*size*
The size of the alternate index's data and index components.

Because an alternate index always has the spanned attribute, the control interval size can be less than the maximum record length. You can define a size from 512, to 8K in increments of 512 or from 8K to 32K in increments of 2K (where K is 1024 in decimal notation). If you use a size that is not a multiple of 512 or 2048, VSAM chooses the next higher multiple.

The index control interval should be large enough to accommodate all of the compressed keys in a data control area. If the index control interval size is too small, unnecessary control area splits can occur. After the first define (DEFINE), a catalog listing ( LISTC ) shows the number of control intervals in a control area and the key length of the data set. To make a general estimate of the index control interval size needed, multiply one-half of the key length (KEYLEN) by the number of data control intervals per control area (DATA CI/CA):

(KEYLEN/2 ) * DATA CI/CA ≤ INDEX CISIZE

Refer to "Optimizing VSAM Performance" in *z/OS DFSMS: Using Data Sets* for the relationship between control interval size and physical block size. It also includes restrictions that apply to control interval size and physical block size.

**Abbreviations:** CISZ or CNVSZ

**DATACLASS(*class*)**
The 1-to-8 character name of the data class for the data set. It provides the allocation attributes for new data sets. Your storage administrator defines the data class. However, you can override the parameters defined for DATACLASS by explicitly defining other attributes. See "Specifying Attribute Selection Order" on page 14 for the order of precedence (filtering) the system uses to select which attribute to assign. The record organization attribute of DATACLASS is not used for DEFINE ALTERNATEINDEX.

DATACLASS parameters apply to both SMS- and non-SMS-managed data sets. If DATACLASS is used and SMS is inactive, the DEFINE is unsuccessful.

You cannot use DATACLASS as a subparameter of DATA or INDEX.

**Abbreviation:** DATACLAS

**ERASE|NOERASE**
indicates if the records of the alternate index components are erased when the alternate index is deleted.

**ERASE**
Requires the records of the alternate index components are overwritten with binary zeros when the alternate index is deleted. If the base cluster of the alternate index is protected by a RACF generic or discrete profile and the base cluster is cataloged in an integrated catalog facility catalog, you can use RACF commands to specify an ERASE attribute as part of this profile so that the component is automatically erased upon deletion.

**Abbreviation:** ERAS

**NOERASE**
Specifies that the records of the alternate index components are not to be overwritten with binary zeros. NOERASE prevents the component from being erased if the base cluster of the alternate index is protected by a RACF generic or discrete profile that specifies the ERASE attribute and if the base cluster is cataloged in an integrated catalog facility catalog. You can use RACF commands to alter the ERASE attribute in a profile.

**Abbreviation:** NERAS

**EXCEPTIONEXIT(***entrypoint***)**
The name of your exception exit routine, that receives control when an exceptional I/O error condition occurs during the transfer of data between your program's address space and the alternate index's direct access storage space. (An exception is any condition that causes a SYNAD exit to be taken.) The component's exception exit routine is processed first; then SYNAD exit routine receives control. If an exception exit routine is loaded from an unauthorized library during access method services processing, an abnormal termination occurs.

**Abbreviation:** EEXT

**FILE(***ddname***)**
Names the DD statement that identifies the direct access devices and volumes on which to allocate space to the alternate index. If more than one volume is specified in a volume list, all volumes must be the same device type.

When the data component and index component are to reside on different devices, you can create a separate FILE parameter as a parameter of DATA and INDEX to point to different DD statements.

If the FILE parameter is not used, an attempt is made to dynamically allocate the required volumes. The volumes must be mounted as permanently resident or reserved.

The DD statement you specify must be:

```
//ddname DD UNIT=(devtype[,unitcount]),
//  VOL=SER=(volser1,volser2,volser3,...),DISP=OLD
```

**Restriction:** When FILE refers to more than one volume of the same device type, the DD statement that describes the volumes cannot be a concatenated DD statement.

**FREESPACE(***CI-percent***[** *CA-percent***]|0 0)**
Designates the amount of empty space left after any primary or secondary allocation and any split of control intervals *(CI-percent)* and control areas *(CA-percent)* when the alternate index is built (see "Chapter 8. BLDINDEX" on page 87

page 87). The empty space in the control interval and control area is available for data records that are updated and inserted after the alternate index is initially built. The amounts are specified as percentages. *CI-percent* translates into a number of bytes that is either equal to, or slightly less than, the percentage value of *CI-percent*. *CA-percent* translates into a number of control intervals that is either equal to, or less than, the percentage of *CA-percent*.

The percentages must be equal to, or less than, 100. When you use 100% of free space, one data record is placed in the first control interval of each control area when the alternate index is built.

**Abbreviation:** FSPC

**IMBED|NOIMBED**

**IMBED|NOIMBED**
IMBED|NOIMBED is no longer supported; if it is specified, it will be ignored and no message will be issued.

**KEYRANGES((***lowkey highkey***)[(***lowkey highkey***)...])**
KEYRANGE is no longer supported; if it is specified, it will be ignored and no message will be issued.

**KEYS(***length offset***|64 0)**
Describes the alternate-key field in the base cluster's data record.

The key field of an alternate index is called an alternate key. The data record's alternate key can overlap or be contained entirely within another (alternate or prime) key field.

The length plus offset cannot be greater than the length of the base cluster's data record.

When the base cluster's data record spans control intervals, the record's alternate-key field is within the record's first segment (that is, in the first control interval).

*length offset*
Gives the length of the alternate key (between 1 and 255), in bytes, and its displacement from the beginning of the base cluster's data record, in bytes.

**MODEL(***entryname***[** *catname***])**
Uses existing entry as a model for the entry being defined or recataloged.

DATACLASS, MANAGEMENTCLASS, and STORAGECLASS cannot be modeled. See "Specifying Attribute Selection Order" on page 14 for information on how the system selects modeled attributes.

You can use an existing alternate index's entry as a model for the attributes of the alternate index being defined. For details about how a model is used, see *z/OS DFSMS: Managing Catalogs*.

You can use some attributes of the model and override others by defining them in the cluster or component. If you do not want to add or change any attributes, use only the entry type (alternate index, data, or index) of the model and the name of the entry to be defined.

When you use an alternate index entry as a model for an alternate index, the model entry's data and index components are used as models for the to-be-defined entry's data and index components, unless another entry is specified with the MODEL parameter as a subparameter of DATA or INDEX.

*entryname*
Names the entry to be used as a model.

*catname*
Names the model entry's catalog. You must identify the catalog that contains the model entry when:

- The model entry's catalog is not identified with a JOBCAT or STEPCAT DD statement, and is not the master catalog.

If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved. See "Catalog Selection Order for DEFINE" on page 10 for information about the order in which a catalog is selected when the catalog's name is not specified.

**ORDERED|UNORDERED**
ORDERED|UNORDERED is no longer supported; if it is specified, it will be ignored and no message will be issued.

**OWNER(***ownerid***)**
Gives the identification of the alternate index's owner.

For TSO users, if the owner is not identified with the OWNER parameter, the TSO user's userid becomes the *ownerid*.

**RECATALOG|NORECATALOG**
Specifies whether the catalog entries for the alternate index components are re-created from information in the VVDS.

**RECATALOG**
Recreates the catalog entries if valid VVDS entries are found on the primary VVDS volume. If not, the command ends.

Use of RECATALOG requires that the NAME, RELATE, and VOLUMES parameters be specified as they were when the alternate index was originally defined. If you use RECATALOG, you are not required to include CYLINDERS, RECORDS, or TRACKS.

If ATTEMPTS, AUTHORIZATION, CATALOG, CODE, FOR, MODEL, NOUPGRADE, OWNER, or TO parameters were used during the original define, they must be entered again with RECATALOG to restore their original values; otherwise, their default values are used.

**Abbreviation:** RCTLG

**NORECATALOG**
Specifies that the catalog entries are not to be re-created from VVDS entries. Catalog entries are created for the first time.

**Abbreviation:** NRCTLG

**RECORDSIZE(***average maximum***|4086 32600)**
The average and maximum length, in bytes, of an alternate index record.

An alternate index record can span control intervals, so RECORDSIZE can be larger than CONTROLINTERVALSIZE. The formula for the maximum record size of spanned records as calculated by VSAM is:

```
MAXLRECL = CI/CA * (CISZ - 10)
```

where:
- MAXLRECL is the maximum spanned record size
- CI/CA represents the number of control intervals per control area
- CA is the number of control areas
- CISZ is the quantity control interval size

You can use these formulas to determine the size of the alternate index record when the alternate index supports:

- a key-sequenced base cluster

  ```
  RECSZ = 5 + AIXKL + (n x BCKL)
  ```

- an entry-sequenced base cluster

  ```
  RECSZ = 5 + AIXKL + (n x 4)
  ```

where:

- RECSZ is the average record size.
- AIXKL is the alternate-key length (see the KEYS parameter).
- BCKL is the base cluster's prime-key length (you can enter the LISTCAT command to determine the base cluster's prime-key length).

**REPLICATE|NOREPLICATE**

REPLICATE|NOREPLICATE is no longer supported; if it is specified, it will be ignored and no message will be issued.

**REUSE|NOREUSE**

Indicate whether or not the alternate index can be used again as a new alternate index.

**REUSE**

Indicates that the alternate index can be used over again as a new alternate index. When a reusable alternate index is opened, its high-used RBA can be set to zero. Open it with an access control block using the RESET attribute.

When you use BLDINDEX to build a reusable alternate index, the high-used RBA is always reset to zero when the alternate index is opened for BLDINDEX processing.

Reusable alternate indexes can be multivolumed and might have up to 123 physical extents.

**Exception:** If you use the keyword **UNIQUE** with **REUSE**, the DEFINE command is unsuccessful.

**Abbreviation:** RUS

**NOREUSE**

Specifies that the alternate index cannot be used again as a new alternate index.

**Abbreviation:** NRUS

**SHAREOPTIONS(***crossregion***[** *crosssystem***]|1 3)**

Specifies how an alternate index's data or index component can be shared among users. However, SMS-managed volumes, and catalogs containing SMS-managed data sets, must not be shared with non-SMS systems. For data integrity, ensure that share options defined for data and index components are the same. For a description of data set sharing, see *z/OS DFSMS: Using Data Sets*.

*crossregion*

Indicates the amount of sharing allowed among regions within the same system or within multiple systems using global resource serialization (GRS). Independent job steps in an operating system, or multiple systems in a GRS ring, can access a VSAM data set concurrently. For more information

about GRS, see *z/OS MVS Planning: Global Resource Serialization*. To share a data set, each user must include DISP=SHR in the data set's DD statement. You can use the following options:

**OPT 1** The data set can be shared by any number of users for read processing, or the data set can be accessed by only one user for read and write processing. This setting does not allow any non-RLS access when the data set is already open for RLS processing. An RLS open will fail with this option if the data set is already open for any processing.

**OPT 2** The data set can be accessed by any number of users for read processing, and it can also be accessed by one user for write processing. It is the user's responsibility to provide read integrity. VSAM ensures write integrity by obtaining exclusive control for a control interval while it is being updated. An RLS open is not allowed while the data set is open for non-RLS output.

If the data set has already been opened for RLS processing, a non-RLS open for input is allowed; a non-RLS open for output fails.[2] If the data set is opened for input in non-RLS mode, an RLS open is allowed.

**OPT 3** The data set can be fully shared by any number of users. The user is responsible for maintaining both read and write integrity for the data the program accesses. This setting does not allow any non-RLS access when the data set is already open for RLS processing. If the data set is opened for input in non-RLS mode, an RLS open is allowed.

This option is the only one applicable to an integrated catalog facility catalog.

**OPT 4** The data set can be fully shared by any number of users. For each request, VSAM refreshes the buffers used for direct processing. This setting does not allow any non-RLS access when the data set is already open for RLS processing. If the data set is opened for input in non-RLS mode, an RLS open is allowed.

As in SHAREOPTIONS 3, each user is responsible for maintaining both read and write integrity for the data the program accesses.

*crosssystem*

The amount of sharing allowed among systems. Job steps of two or more operating systems can gain access to the same VSAM data set regardless of the disposition specified in each step's DD statement for the data set. However, if you are using GRS across systems or JES3, the data set might not be shared depending on the disposition of the system.

To get exclusive control of the data set's volume, a task in one system issues the RESERVE macro. The level of cross-system sharing allowed by VSAM applies only in a multiple operating system environment.

The cross-system sharing options are ignored by RLS processing. The values are:

**1** Reserved.

**2** Reserved.

2. You must apply APARs OW25251 and OW25252 to allow non-RLS read access to data sets already opened for RLS processing.

**3** Specifies that the data set can be fully shared. Each user is responsible for maintaining both read and write integrity for the data that user's program accesses. User programs that ignore write integrity guidelines can result in:
- VSAM program checks
- Uncorrectable data set errors
- Unpredictable results

The RESERVE and DEQ macros are required with this option to maintain data set integrity. (See *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN* and *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU* for information on using RESERVE and DEQ.) If the sphere is accessed using RLS protocols, VSAM RLS maintains the required integrity.

**4** Specifies that the data set can be fully shared. For each request, VSAM refreshes the buffers used for direct processing. This option requires that you use the RESERVE and DEQ macros to maintain data integrity while sharing the data set. Improper use of the RESERVE macro can cause problems similar to those described under SHAREOPTIONS 3. (See *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN* and *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU* for information on using RESERVE and DEQ.) Output processing is limited to update, or add processing, or both that does not change either the high-used RBA or the RBA of the high key data control interval if DISP=SHR is used.

To ensure data integrity in a shared environment, VSAM provides users of SHAREOPTIONS 4 (cross-region and cross-system) with the following assistance:
- Each PUT writes the appropriate buffer immediately into the VSAM object's DASD. VSAM writes out the buffer in the user's address space that contains the new or updated data record.
- Each GET refreshes the user's input buffers. The contents of each data and index buffer used by the user's program is retrieved from the VSAM object's DASD.

**Exception:** If you use RLS, SHAREOPTIONS is assumed to be (3,3). If you do not use RLS, the SHAREOPTIONS specification is respected.

**Abbreviation:** SHR

**SPEED|RECOVERY**

Specify whether or not the data component's control areas are preformatted before alternate index records are loaded into them.

This parameter is only considered during the actual loading (creation) of a data set, when the data set is opened and the high-used RBA is equal to zero. After normal CLOSE processing at the completion of the load operation, the physical structure of the data set and the content of the data set extents are exactly the same. Any processing of the data set following that successful load operation is the same and the specification of this parameter is not considered.

If you use RECOVERY, the initial load takes longer because the control areas are first written with either empty or software end-of-file intervals. These

preformatted CIs are then updated, using update writes, with the alternate index records. SPEED is recommended, because the initial load is quicker.

**SPEED**
Does not preformat the data component's space.

If the initial load is unsuccessful, you must load the alternate index records again from the beginning, because VSAM cannot determine the location of your last correctly written record. VSAM cannot find a valid end-of-file indicator when it searches your alternate index records.

**RECOVERY**
Specifies that the data component's control areas are written with records that indicate end-of-file. When an alternate index record is written into a control interval, it is always followed by a record that identifies the record just written as the last record in the alternate index.

RECOVERY is a way to verify storage used on the device for each CA before the data is actually written.

**Abbreviation:** RCVY

**TO(*date*)|FOR(*days*)**
The retention period for the alternate index. The alternate index is not automatically deleted when the expiration date is reached. When you do not provide a retention period, the alternate index can be deleted at any time. The MANAGEMENTCLASS maximum retention period, if used, limits the retention period named by this parameter.

For non-SMS-managed data sets, the correct retention period is reflected in the catalog entry. The VTOC entry might not have the correct retention period. Enter a LISTCAT command to see the correct expiration date.

For SMS-managed data sets, the expiration date in the catalog is updated and the expiration date in the format-1 DSCB is changed. Should the expiration date in the catalog not agree with the expiration date in the VTOC, the VTOC entry overrides the catalog entry. In this case, enter a LISTVTOC command to see the correct expiration date.

**TO(*date*)**
Specifies the earliest date that alternate index can be deleted. The date is specified in the form [yy]yyddd, where yyyy is a four-digit year, yy is a two-digit year, and ddd is the three-digit (001 through 366) day of the year. Two-digit years are treated as if "19" is specified as the first two digits of yyyy. The dates (19)99365 and (19)99366 are considered never-expire dates.

For expiration dates of January 1, 2000, and later, you must use the form TO(yyyyddd). the first two digits of yyyy.

The maximum yyyy is a four-digit year (through 2155) and the maximum ddd is a three digit day number from 000 through 365 for non leap years. For leap years, the maximum ddd is 366.

**FOR(*days*)**
Is the number of days to keep the alternate index before it is deleted. The maximum number is 9999. If the number is 0 through 9998, the alternate index is retained for that number of days; if the number is 9999, the alternate index is retained indefinitely.

**UNIQUEKEY|NONUNIQUEKEY**

Shows whether more than one data record (in the base cluster) can contain the same key value for the alternate index.

**UNIQUEKEY**

Points each alternate index key to only one data record. When the alternate index is built (see “Chapter 8. BLDINDEX” on page 87) and more than one data record contains the same key value for the alternate index, the BLDINDEX processing ends with an error message.

**Abbreviation:** UNQK

**NONUNIQUEKEY**

points a key value for the alternate index to more than one data record in the base cluster. The alternate index’s key record points to a maximum of 32768 records with non-unique keys.

When you include NONUNIQUEKEY, the maximum record size should be large enough to allow for alternate index records that point to more than one data record.

**Abbreviations:** NUNQK

**UPGRADE|NOUPGRADE**

Specify whether or not the alternate index is to be upgraded (that is, kept up to date) when its base cluster is modified.

**UPGRADE**

Upgrades the cluster’s alternate index to reflect changed data when the base cluster’s records are added to, updated, or erased.

When UPGRADE is specified, the alternate index’s name is cataloged with the names of other alternate indexes for the base cluster. The group of alternate index names identifies the upgrade set that includes all the base cluster’s alternate indexes that are opened when the base cluster is opened for write operations.

The UPGRADE attribute is not effective for the alternate index until the alternate index is built (see “Chapter 8. BLDINDEX” on page 87). If the alternate index is defined when the base cluster is open, the UPGRADE attribute takes effect the next time the base cluster is opened.

**Abbreviation:** UPG

**NOUPGRADE**

The alternate index does not upgrade when its base cluster is modified.

**Abbreviation:** NUPG

**WRITECHECK|NOWRITECHECK**

Determines whether an alternate index or component is checked by a machine action called write-check when a record is written into it.

**WRITECHECK**

Indicates that a record is written and then read, without data transfer, to test for the data check condition.

**Exception:** When you use access RLS, the WRITECHECK parameter is ignored.

**Abbreviation:** WCK

**NOWRITECHECK**
Does not write-check the alternate index or component. checked by a write check.

**Abbreviation:** NWCK

## Data and Index Components of an Alternate Index

Attributes can be specified separately for the alternate index's data and index components. There is a list of the DATA and INDEX parameters at the beginning of this section. These are described in detail as parameters of the alternate index as a whole. Restrictions are noted with each description.

## DEFINE ALTERNATEINDEX Examples

### Define an Alternate Index Using SMS Data Class Specification: Example 1

In this example, an SMS-managed alternate index is defined. Because a data class is specified and no overriding attributes are explicitly specified, this define will be unsuccessful if SMS is inactive.

```
//DEFAIX   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
      DEFINE ALTERNATEINDEX -
           (NAME(EXMP1.AIX) -
           RELATE(EXAMPLE.SMS1) -
           DATACLAS(VSALLOC) -
           NONUNIQUEKEY -
           UPGRADE)
/*
```

The DEFINE ALTERNATEINDEX command creates an alternate index entry, a data entry, and an index entry to define the alternate index EXMP1.AIX. The parameters are:

- NAME indicates that the alternate index's name is EXMP1.AIX.
- RELATE identifies the alternate index base cluster, EXAMPLE.SMS1. Because an SMS-managed alternate index is being defined, the base cluster must also be SMS-managed.
- DATACLAS is an installation-defined name of an SMS data class. The data set assumes the RECORG or RECFM, LRECL, KEYLEN, KEYOFF, AVGREC, SPACE, EXPDT or RETPD, VOLUME, CISIZE, FREESPACE and SHAREOPTIONS parameters assigned to this data class by the ACS routines. This parameter is optional. If it is not used, the data set will assume the data class default assigned by the ACS routines.
- NONUNIQUEKEY specifies that the alternate key value might be the same for two or more data records in the base cluster.
- UPGRADE specifies that the alternate index is to be opened by VSAM and upgraded each time the base cluster is opened for processing.

### Define an SMS-Managed Alternate Index: Example 2

In this example, an SMS-managed alternate index is defined. Data class is not used, and explicitly defined attributes override any attributes in the default data class.

```
//DEFAIX   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
      DEFINE ALTERNATEINDEX -
           (NAME(EXMP2.AIX) -
           RELATE(EXAMPLE.SMS2) -
           KEYS(3 0) -
           RECORDSIZE(40 50) -
           KILOBYTES(1600 200) -
           NONUNIQUEKEY -
           UPGRADE)
/*
```

The DEFINE ALTERNATEINDEX command creates an alternate index entry, a data entry, and an index entry to define the alternate index EXMP2.AIX. The command's parameters are:

- NAME indicates that the alternate index's name is EXMP2.AIX.
- RELATE identifies the alternate index base cluster, EXAMPLE.SMS2. Because an SMS-managed alternate index is being defined, the base cluster must also be SMS-managed.
- KEYS specifies the length and location of the alternate key field in each of the base cluster's data records. The alternate key field is the first three bytes of each data record.
- RECORDSIZE specifies that the alternate index's records are variable length, with an average size of 40 bytes and a maximum size of 50 bytes.
- KILOBYTES allocates the minimum number of tracks required to contain 1600 kilobytes for the alternate index's space. When the alternate index is extended, it is to be extended by the minimum number of tracks required to contain 200 kilobytes.
- NONUNIQUEKEY means the alternate key value might be the same for two or more data records in the base cluster.
- UPGRADE opens the alternate index by VSAM and upgrades it each time the base cluster is opened for processing.

## Define an Alternate Index: Example 3

In this example, an alternate index is defined. An example for DEFINE CLUSTER illustrates the definition of the alternate index base cluster, EXAMPLE.KSDS2. A subsequent example illustrates the definition of a path, EXAMPLE.PATH, that lets you process the base cluster's data records using the alternate key to locate them. The alternate index, path, and base cluster are defined in the same catalog, USERCAT.

```
//DEFAIX1  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DEFINE ALTERNATEINDEX -
           (NAME(EXAMPLE.AIX) -
           RELATE(EXAMPLE.KSDS2) -
           KEYS(3 0) -
           RECORDSIZE(40 50) -
           VOLUMES(VSER01) -
           CYLINDERS(3 1) -
           NONUNIQUEKEY -
           UPGRADE) -
         CATALOG(USERCAT)
/*
```

The DEFINE ALTERNATEINDEX command creates an alternate index entry, a data entry, and an index entry to define the alternate index EXAMPLE.AIX. The DEFINE ALTERNATEINDEX command also obtains space for the alternate index from one of the VSAM data spaces on volume VSER01, and allocates three cylinders for the alternate index's use. The parameters are:

- NAME indicates that the alternate index's name is EXAMPLE.AIX.
- RELATE identifies the alternate index base cluster, EXAMPLE.KSDS2.
- KEYS identifies the length and location of the alternate key field in each of the base cluster's data records. The alternate key field is the first three bytes of each data record.
- RECORDSIZE specifies that the alternate index's records are variable length, with an average size of 40 bytes and a maximum size of 50 bytes.
- VOLUMES indicates that the alternate index is to reside on volume VSER01. This example assumes that the volume is already cataloged in the user catalog, USERCAT.
- CYLINDERS allocates three cylinders for the alternate index's space. The alternate index is extended in increments of one cylinder.
- NONUNIQUEKEY specifies that the alternate key value might be the same for two or more data records in the base cluster.
- UPGRADE specifies that the alternate index is opened by VSAM and upgraded each time the base cluster is opened for processing.
- CATALOG defines the alternate index in the user catalog, USERCAT.

## Define an Alternate Index with RECATALOG: Example 4

In this example, an alternate index is redefined into an integrated catalog facility catalog.

```
//DEFAIXR  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DEFINE ALTERNATEINDEX -
           (NAME(DEFAIXR.AIX01) -
           RELATE(DEFKSDS.KSDS03) -
           CYLINDERS(2 1) -
           VOLUMES(333001) -
           RECATALOG) -
         CATALOG(USERCAT4)
/*
```

This DEFINE ALTERNATEINDEX command recatalogs an alternate index entry, a data entry, and an index entry to redefine the alternate index, DEFAIXR.AIX01. The VSAM volume record (VVR) entry and the corresponding VTOC entry for the alternate index must exist. Only the catalog entry is recataloged, so no space is allocated. The command's parameters are:

- NAME indicates the alternate index's name, DEFAIXR.AIX01.
- RELATE identifies the alternate index base cluster, DEFKSDS.KSDS03.
- CYLINDERS allocates two cylinders for the alternate index's space. The alternate index is extended in increments of one cylinder.
- VOLUMES places the alternate index on volume 333001. This example assumes that a VTOC entry already exists for this object.
- RECATALOG recatalogs the alternate index and uses the existing VVR entry and VTOC entry.
- CATALOG defines the alternate index in the user catalog, USERCAT4.

# Chapter 14. DEFINE CLUSTER

Using Access Method Services, you can set up jobs to execute a sequence of commands with a single invocation of IDCAMS. Modal command execution is based on the success or failure of prior commands.

Use DEFINE CLUSTER to define attributes for the cluster as a whole and for the components of the cluster. The general syntax of the DEFINE CLUSTER command is:

**DEFINE CLUSTER** (parameters) -
[**DATA**(parameters)] -
[**INDEX**(parameters)] -
[**CATALOG**(subparameters)]

| | |
|---|---|
| **DEFINE** | **CLUSTER**<br>**(NAME(***entryname***)**<br>**{CYLINDERS(***primary***[** *secondary***])\|**<br>**KILOBYTES(***primary***[** *secondary***])\|**<br>**MEGABYTES(***primary***[** *secondary***])\|**<br>**RECORDS(***primary***[** *secondary***])\|**<br>**TRACKS(***primary***[** *secondary***])}**<br>**VOLUMES(***volser***[** *volser***...])**<br>[**ACCOUNT(***account–info***)**]<br>**[BUFFERSPACE(***size***)]**<br>**[BWO(TYPECICS\|TYPEIMS\|NO)]**<br>**[CONTROLINTERVALSIZE(***size***)]**<br>**[DATACLASS(***class***)]**<br>**[ERASE\|NOERASE]**<br>**[EXCEPTIONEXIT(***entrypoint***)]**<br>**[FILE(***ddname***)]**<br>**[FREESPACE(***CI-percent***[** *CA-percent***]\|0 0)]**<br>**[FRLOG(***NONE***[\|** *REDO***])]**<br>**[INDEXED\|LINEAR\|NONINDEXED\|NUMBERED]**<br>**[KEYS(***length offset***\| 64 0]**<br>**[LOG(NONE\|UNDO\|ALL)]**<br>**[LOGSTREAMID(***logstream***)]**<br>**[MANAGEMENTCLASS(***class***)]**<br>**[MODEL(***entryname***[** *catname***])]**<br>**[OWNER(***ownerid***)]**<br>**[RECATALOG\|NORECATALOG]**<br>**[RECORDSIZE(***average maximum***)]**<br>**[REUSE\|NOREUSE]**<br>**[SHAREOPTIONS(***crossregion***[** *crosssystem***]\|1 3)]**<br>**[SPANNED\|NONSPANNED]**<br>**[SPEED\|RECOVERY]**<br>**[STORAGECLASS(***class***)]**<br>**[TO(***date***)\|FOR(***days***)]**<br>**[WRITECHECK\|NOWRITECHECK])**<br>**[DATA** (<br>**{CYLINDERS(***primary***[** *secondary***])\|**<br>**KILOBYTES(***primary***[** *secondary***])\|**<br>**MEGABYTES(***primary***[** *secondary***])\|**<br>**RECORDS(***primary***[** *secondary***])\|**<br>**TRACKS(***primary***[** *secondary***])}** |

© Copyright IBM Corp. 1973, 2002

```
  [VOLUMES(volser[ volser...])]
  [BUFFERSPACE(size)]
  [CONTROLINTERVALSIZE(size)]
  [ERASE|NOERASE]
  [EXCEPTIONEXIT(entrypoint)]
  [FILE(ddname)]
  [FREESPACE(CI-percent[ CA-percent])]
  [KEYS(length  offset)]
  [MODEL(entryname[ catname])]
  [NAME(entryname)]
  [OWNER(ownerid)]
  [RECORDSIZE(average  maximum)]
  [REUSE|NOREUSE]
  [SHAREOPTIONS(crossregion[ crosssystem])]
  [SPANNED|NONSPANNED]
  [SPEED|RECOVERY]
  [WRITECHECK|NOWRITECHECK])]
[INDEX (
  {CYLINDERS(primary[ secondary])|
   KILOBYTES(primary[ secondary])|
   MEGABYTES(primary[ secondary])|
   RECORDS(primary[ secondary])|
   TRACKS(primary[ secondary])}
  [VOLUMES(volser[ volser...])]
  [CONTROLINTERVALSIZE(size)]
  [EXCEPTIONEXIT(entrypoint)]
  [FILE(ddname)]
  [MODEL(entryname [ catname])]
  [NAME(entryname)]
  [OWNER(ownerid)]
  [REUSE|NOREUSE]
  [SHAREOPTIONS(crossregion[ crosssystem])]
  [WRITECHECK|NOWRITECHECK])]
[CATALOG(catname)]
```

**DEFINE Abbreviation: DEF**

A sequence of commands commonly used in a single job step includes DELETE—DEFINE—REPRO or DELETE—DEFINE—BLDINDEX. You can specify either a DD name or a data set name with these commands. When you refer to a DD name, however, allocation occurs at job step initiation. This could result in a job failure if a command such as REPRO follows a DELETE—DEFINE sequence that changes the location (volser) of the data set. A failure can occur with either SMS-managed data sets.

To avoid potential failures with a modal command sequence in your IDCAMS job:

- Specify the data set name instead of the DD name; or
- Use a separate job step to perform any sequence of commands (for example, REPRO, IMPORT, BLDINDEX, PRINT, or EXAMINE) that follow a DEFINE command.

**Recommendation**:DB2 uses Access Method Services DEFINE CLUSTER for STOGROUP defined data sets. This can result in performance problems for partitioned table spaces if multiple partitions are defined on the same volume. DB2

uses software striping on partitioned table spaces to improve performance of sequential queries. The throughput is then gated by the data delivery capability of each volume. Since each partition is a separate data set, this problem can be avoided by allocating all the partitions in a single JCL step in an IEFBR14 (not IDCAMS) job. See *z/OS DFSMS: Using Data Sets* for details. Allocating all the partitions in this manner works if there are adequate number of volumes available with the requested space quantity in a single SMS storage group to satisfy all the partitions.

**Restriction**: If you specify IMBED, KEYRANGE, ORDERED, or REPLICATE it will be ignored.

## DEFINE CLUSTER Parameters

The DEFINE CLUSTER command uses the following parameters.

## Required Parameters

**CLUSTER**
CLUSTER defines or recatalogs a cluster or cluster entry.

The CLUSTER keyword is followed by the parameters specified for the cluster as a whole. These parameters are enclosed in parentheses and, optionally, are followed by parameters given separately for the DATA and INDEX components.

**Abbreviation:**CL

**NAME(***entryname***)**
Defines the cluster's entryname or the name of each of its components. The entryname used for the cluster as a whole is not propagated to the cluster's components.

For SMS and non-SMS-managed clusters, the component names must resolve to the same catalog as the data set's cluster name.

You can define a separate entryname for the cluster, its data component, and its index component. If no name is specified for the data and index component, a name is generated. When the cluster, data component, and index component are individually named, each can be addressed. For information on system generated names, see *z/OS DFSMS: Using Data Sets*.

When defining a VSAM volume data set (VVDS), the entryname for the cluster or the data component must be in the form SYS1.VVDS.Vvolser, where volume serial number is the volume serial number specified by the VOLUMES parameter. The default primary and secondary allocation is 10 tracks. For information on defining a VVDS see *z/OS DFSMS: Managing Catalogs*.

**CYLINDERS(***primary***[** *secondary***])**

**KILOBYTES(***primary***[** *secondary***])|**

**MEGABYTES(***primary***[** *secondary***])|**

**RECORDS(***primary***[** *secondary***])|**

**TRACKS(***primary***[** *secondary***])|**
The amount of space in cylinders, kilobytes, megabytes, records, or tracks allocated to the cluster from the volume's available space. A kilobyte or megabyte allocation resolves to either tracks or cylinders; record allocation resolves to tracks.

**Exception:** If allocation resolves to tracks, the space is contiguous. See "Optimizing Control Area Size" in *z/OS DFSMS: Using Data Sets* for more information.

Requests for space are directed to DADSM and result in a format-1 DSCB for all entries.

If the cluster is not SMS-managed, you must use the amount of space allocated, either through this parameter, or through the DATACLASS, MODEL, or RECATALOG parameters. This parameter is optional if the cluster is managed by SMS. If it is used, it overrides the DATACLASS space specification. If it is not used, it can be modeled or defaulted by SMS. If it cannot be determined, the DEFINE is unsuccessful.

If you select KILOBYTES or MEGABYTES, the amount of space allocated is the minimum number of tracks or cylinders required to contain the specified number of kilobytes or megabytes.

If you select RECORDS, the amount of space allocated is the minimum number of tracks that are required to contain the given number of records. The maximum number of records is 16,777,215. If RECORDS is specified for a linear data set, space is allocated with the number of control intervals equal to the number of records.

To maintain device independence, do not use the TRACKS or CYLINDERS parameters. If you use them for an SMS-managed data set, space is allocated on the volumes selected by SMS in units equivalent to the device default geometry. If there is an allocation failure due to lack of space, SMS retries allocation with a reduced space quantity. However, any retry, including reduced space quantity, is only attempted if Space Constraint Relief ⇒ Y is specified. SMS also removes other limitations if the data class allows space constraint relief.

Regardless of the allocation type, the calculation of the CA (control area) size is based on the smaller of the two allocation quantities (primary or secondary) in the DEFINE command. A CA is never greater than a single cylinder, it might be less (that is, some number of tracks), depending on the allocation amount and type used. When tracks or records are used, the space allocation unit (the CA size) can be adjusted to one cylinder. This adjustment is made if the calculated CA size contains more tracks than exist in a single cylinder of the device being used. The CA area size assigned by VSAM is the smallest of:

- One cylinder
- The primary space quantity
- The secondary space quantity

If the CA size assigned is not evenly divisible into either the primary or secondary space quantity, VSAM increases that space to a value evenly divisible by the CA size. If you are defining an extended format data set, you should review ″Defining an Extended Format Key-Sequenced Data Set″ in z/OS DFSMS: Using Data Sets for information about additional space requirements.

DEFINE RECORDS allocates sufficient space to the specified number of records, but factors unknown at define time (such as key compression or method of loading records) can result in inefficient use of the space allocated.

This might prevent every data CA from being completely used, and you might be unable to load the specified number of records without requiring secondary allocation.

When multiple volumes are used for a data set, these rules and conditions apply:

- The first volume is defined as the prime volume. The initial allocation of a data set is on the prime volume. The remaining volumes are defined as candidate volumes.
- A data set's primary space allocation (defined for each data set) is the amount of space initially allocated on both the prime volume and on any candidate volumes the data set extends to.
- A data set's secondary space allocation (if it is defined) is the space allocated when the primary space is filled and the data set needs additional space on the same volume.
- If a data set extends to a candidate volume, the amount of space initially allocated on the candidate volume is the primary space allocation. If the data set extends beyond the primary allocation on the candidate volume, then the amount of space allocated is the secondary space allocation.
- With a DEFINE request, the primary space allocation must be fulfilled in five DASD extents unless the Space Constraint Relief option is specified in the associated SMS data class.

  However, the request is not successful if you do not fulfill each secondary space allocation in five DASD extents. A DASD extent is the allocation of one available area of contiguous space on a volume. For example, if a data set's primary space allocation is 100 cylinders, you must allocate a maximum of five DASD extents that add up to 100 cylinders.

Secondary amounts can be allocated on all volumes available to contain parts of the cluster regardless of the key ranges.

You can specify the amount of space as a parameter of CLUSTER, as a parameter of DATA, or as a parameter of both. When a key-sequenced cluster is being defined, and the space is a parameter of:

- CLUSTER, the amount is divided between the data and index components. The division algorithm is a function of control interval size, record size, device type, and other data set attributes.

  If the division results in an allocation for the data component that is not an integral multiple of the required control area size, the data component's allocation is rounded up to the next higher control area multiple. This rounding can result in a larger total allocation for your cluster.
- DATA, the entire amount specified is allocated to the data component. An additional amount of space, depending on control interval size, record size, device type, and other data set attributes, is allocated to the index component.

To determine the exact amount of space allocated to each component, list the cluster's catalog entry, using the LISTCAT command.

The primary and each secondary allocation must be able to be satisfied in five DASD extents; otherwise, your DEFINE or data set extension is unsuccessful.

*primary*
Allocates the initial amount of space to the cluster.

*secondary*
Allocates an amount of space each time the cluster extends, as a secondary extent. You can use this secondary allocation to add space for the data or index components of the cluster. A VSAM data set can be expanded to 123 extents per volume. If this is a multi-volume VSAM data set, then the VSAM component can be extended to a maximum of 255 extents combined over all volumes.

**VOLUMES(***volser***[** *volser***...])**
Specifies the volumes on which a cluster's components are to have space. If you do not use the MODEL parameter, or if the cluster is not SMS-managed, VOLUMES must be used either as a parameter of CLUSTER, or as a parameter of both DATA and INDEX.

VOLUMES can be specified or modeled for a data set that is to be SMS-managed; know that the volumes specified might not be used and result in an error. See *z/OS DFSMSdfp Storage Administration Reference* for information about SMS volume selection.

Volumes are always allocated in the order specified. If there is not enough space on the volume, the allocation is not successful. For non-SMS-managed data sets, the primary space is allocated on the first volume in the list. When you extend the data set because the first allocation is full, the volumes are used in the order in which they appeared in the DEFINE command.

Letting SMS select the volume from the storage group reduces the chances of allocation errors caused by insufficient space. If the data set is SMS-managed with guaranteed space, SMS places the primary quantity on all the volumes with sufficient space for later extensions. If the SMS-managed data set does not have guaranteed space or is a key range data set, primary space is allocated only on the first volume. For SMS-managed VSAM data sets, the primary space might be allocated on a different volume from the one you specified.

You can let SMS choose the volumes for SMS-managed data sets by coding an * for the volser with the VOLUMES parameter. If both user-specified and SMS-specified volumes are requested, the user-specified volser must be input first in the command syntax. The default is one volume.

For SMS-managed and non-SMS-managed data sets, you can specify up to 59 volume serial numbers. If the combined number of volumes for a cluster and its associated alternate indexes exceeds 59, unpredictable results can occur.

If the data and index components are to reside on different device types, you must specify VOLUMES as a parameter of both DATA and INDEX. If more than one volume is listed with a single VOLUMES parameter, the volumes must be of the same device type.

For SMS-managed data sets, if you want the data and index components to be on separate volumes for non-guaranteed space storage class requests, code two different dummy names in the VOLUME parameter for each component. If there are not enough volumes in the storage group to satisfy this requirement, the allocation will fail.

If a guaranteed space storage class is assigned to the data sets (cluster) and volume serial numbers are used, space is allocated on all specified volumes if the following conditions are met:

- All defined volumes are in the same storage group.
- The storage group to which these volumes belong is in the list of storage groups selected by the ACS routines for this allocation.
- The data set is not a key range data set.

The volume serial number is repeated in the list only if the KEYRANGE parameter is used. You can use this to have more than one key range on the same volume. Repetition is valid when duplicate occurrences are used for the primary allocation of some key range.

If a VVDS is being defined, only one volume can be specified and that volume serial number must be reflected in the name indicated in the NAME parameter.

The VOLUMES parameter interacts with other DEFINE CLUSTER parameters. Ensure that the volume you give for the cluster is consistent with the cluster's other attributes:

- CYLINDERS, KILOBYTES, MEGABYTES, RECORDS, TRACKS: The volume must contain enough unallocated space to satisfy the component's primary space requirement.
- FILE: The volume information supplied with the DD statement pointed to by FILE must be consistent with the information specified for the cluster and its components.

**Abbreviation**CYL, KB, MB, REC, TRK

**Abbreviation**VOL

## Optional Parameters

The DEFINE CLUSTER command can execute the following optional parameters.

**ACCOUNT(***account–info***)**

Defines up to 32 bytes of accounting information and user data for the data set. It *must* be between 1 and 32 bytes, otherwise you will receive an error message.

*account–info*

Is only supported for SMS-managed VSAM and non–VSAM data sets. It is only used for the data set level (not member level) of PDSE/PDS.

**Abbreviation: ACCT**

**BUFFERSPACE(***size***)**

Specifies the minimum space for buffers. The buffer space size helps VSAM determine the data component's and index component's control interval size. If BUFFERSPACE is not coded, VSAM attempts to get enough space to contain two data component control intervals and, if the data is key-sequenced, one index component control interval.

If the data set being defined is a KSDS, and the BUFFERSPACE specified is not large enough to contain two data and one index CI's, VSAM increases the specified buffer space and completes the define. VSAM also increases index CISIZE and, if necessary, increases the buffer space to accommodate the larger index CISIZE.

*size*

The space for buffers. *Size* can be given in decimal (n), hexadecimal (X'n'), or binary (B'n') form, but must not exceed 16,776,704.

The size cannot be less than enough space to contain two data component control intervals and, if the data is key-sequenced, one index control interval. If size is too small for VSAM buffers, VSAM ends your DEFINE and provides an appropriate error message.

The BUFFERSPACE setting is ignored when the data set is opened for RLS.

**Abbreviations:**BUFSP or BUFSPC

**BWO(TYPECICS|TYPEIMS|NO)**
Use this parameter if backup-while-open (BWO) is allowed for the VSAM sphere. BWO applies only to SMS data sets and cannot be used with TYPE(LINEAR). If BWO, LOG, OR LOGSTREAMID are specified (or an RLS cell exists for the data set), access from DFSMS/MVS 1.2 or lower level systems is denied.

If BWO is specified in the SMS data class, the value defined is used as the data set definition, unless it has been previously defined with an explicitly specified or modeled DEFINE attribute.

**TYPECICS**
Use TYPECICS to specify BWO in a CICS environment. For RLS processing, this activates BWO processing for CICS. For non-RLS processing, CICS determines whether to use this specification or the specification in the CICS FCT. See the *CICS System Definition Guide*.

**Exception:** If CICS determines that it will use the specification in the CICS FCT, the specification might override the TYPECICS or NO parameters.

**Abbreviation:**TYPEC

**TYPEIMS**
Use to enable BWO processing for IMS data sets. This can only be used with DFSMS 1.3 or higher level DFSMS systems. If you attempt to open a cluster that has the TYPEIMS specification of a DFSMS 1.2 (or lower) system, the open will not be successful.

**Abbreviation:**TYPEI

**NO**
Use this when BWO does not apply to the cluster.

**Exception:** If CICS determines that it will use definitions in the CICS FCT, the TYPECICS or NO parameters might be overwritten.

**CATALOG(*catname*)**
Identifies the catalog in which the cluster is to be defined. See “Catalog Selection Order for DEFINE” on page 10 for the order in which catalogs are selected.

To specify catalog names for SMS-managed data sets, you must have authority to the RACF STGADMIN.IGG.DIRCAT facility class. See “Storage Management Subsystem (SMS) Considerations” on page 2 for more information.

*catname*
The name of the catalog in which the entry is to be defined.

If the catalog’s volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved.

**Abbreviation:** CAT

**CONTROLINTERVALSIZE(*size*)**
The size of the control interval for the cluster or component.

For linear data sets, the specified value in bytes is rounded up to a 4K multiple, up to a maximum of 32K. If the size is not specified, the value specified in the data class that is assigned to the data set is used. Otherwise a default value of 4K is used.

If CONTROLINTERVALSIZE is given on the cluster level, it propagates to the component level at which no CONTROLINTERVALSIZE has been specified.

The size of the control interval depends on the maximum size of the data records and the amount of buffer space you provide.

LSR/GSR buffering technique users can ensure buffer pool selection by explicitly defining data and index control interval sizes.

If CONTROLINTERVALSIZE is not coded, VSAM determines the size of control intervals. VSAM selects a control interval size for the data component that will optimize direct access storage usage. It will then select an index control interval size based on the number of data control intervals in the data control area.

*size*

Indicates a cluster's data and index component size..

If SPANNED is not used, the size of a data control interval must be at least 7 bytes larger than the maximum record length.

If the control interval specified is less than maximum record length plus a 7-byte overhead, VSAM increases the data control interval size to contain the maximum record length plus the needed overhead.

If SPANNED is specified, the control interval size can be less than the maximum record length. You can select a size from 512 to 8K in increments of 512, or from 8K to 32K in increments of 2K. When you choose a size that is not a multiple of 512 or 2048, VSAM chooses the next higher multiple. For a linear data set, the size specified is rounded up to 4096 if specified as 4096 or less. It is rounded to the next higher multiple of 4096 if specified as greater than 4096.

The size of the index control interval is the number of data control intervals in a data control area that need indexing at the sequence set level of the index component. The size of each entry depends on an average compression value for a user key. The keys will compress to 1/3 of the length of the actual key value. In some cases, the general compressed key length on which the algorithm is based will be affected by the actual values and ordering of the user key. The result is that each entry can occupy more space in the index record than that provided. This may result in additional control area splits and in all cases, wasted space in the data set. If after loading the data sets, this condition exists; noted by more than anticipated space to store the data set on the direct access device. You should increase the index control interval size. The size can be increased incrementally until it is felt that this condition no longer exists. The guideline formula documented in the past is as follows:

(KEYLEN/2 ) * DATA CI/CA less than or equal to INDEX CISIZE.

You should be aware that this is only a guideline and does not take into account the actual algorithm for determining the index control interval size requirement. However, the 2:1 compression of key length in the above formula provides some additional overhead over the actual 3:1 formula used during the actual algorithm. Using the above formula can result in an index control interval

size that is too large. This may increase I/O transfer time for each index component record, or it may be too small to address the condition described above.

Refer to “Optimizing VSAM Performance” in *z/OS DFSMS: Using Data Sets* for a discussion of control interval size and physical block size.

**Abbreviations:**CISZ or CNVSZ

**DATACLASS(***class***)**
Identifies the name, 1-to-8 characters, of the data class for the data set. It provides the allocation attributes for new data sets. Your storage administrator defines the data class. However, you can override the parameters defined for DATACLASS by explicitly using other attributes. See “Specifying Attribute Selection Order” on page 14 for the order of precedence (filtering) the system uses to select which attribute to assign.

DATACLASS parameters apply to both SMS- and non-SMS-managed data sets. If DATACLASS is specified and SMS is inactive, DEFINE is unsuccessful.

DATACLASS cannot be used as a subparameter of DATA or INDEX.

**Abbreviation:**DATACLAS

**ERASE|NOERASE**
Specifies whether the cluster’s components are to be erased when its entry in the catalog is deleted.

**ERASE**
Overwrites each component of the cluster with binary zeros when its catalog entry is deleted. If the cluster is protected by a RACF generic or discrete profile and is cataloged in an integrated catalog facility catalog, you can use RACF commands to specify an ERASE attribute. If you do this, the data component is automatically erased upon deletion.

**Abbreviation:**ERAS

**NOERASE**
Secifies that each component of the cluster is not to be overwritten with binary zeros. NOERASE will not prevent erasure if the cluster is protected by a RACF generic or discrete profile that specifies the ERASE attribute and if the cluster is cataloged in an integrated catalog facility catalog. Use RACF commands to alter the ERASE attribute in a profile.

**Abbreviation: NERAS**

**EXCEPTIONEXIT(***entrypoint***)**
Specifies the name of a user-written exception-exit routine, that receives control when an exceptional I/O error condition occurs during the transfer of data between your program’s address space and the cluster’s DASD space. An exception is any condition that causes a SYNAD exit to be taken. The component’s exception-exit routine is processed first, then the user’s SYNAD exit routine receives control. If an exception-exit routine is loaded from an unauthorized library during access method services processing, an abnormal termination occurs. See *z/OS DFSMS: Using Data Sets*.

**Abbreviation:**EEXT

**FILE(***ddname***)**
Names the DD statement that identifies and allocates the DASD and volumes

that must be available for space allocation on the volumes specified by the VOLUMES keyword. If more than one volume is specified, all volumes must be the same device type.

If data and index components are to reside on separate devices, you can specify a separate FILE parameter as a parameter of DATA and INDEX to point to different DD statements.

If the FILE parameter is not specified, an attempt is made to dynamically allocate the required volumes. The volume must be mounted as permanently resident or reserved. When the FILE parameter is used, the specified volumes are directly allocated before access method services gets control.

An example DD statement is:

```
//ddname DD UNIT=(devtype[,unitcount]),
//    VOL=SER=(volser1,volser2,volser,...),DISP=OLD
```

**Restriction:** When FILE refers to more than one volume of the same device type, the DD statement that describes the volumes cannot be a concatenated DD statement.

**FREESPACE(***CI-percent***[** *CA-percent***]|0 0)**

Specifies the percentage of each control interval and control area to be set aside as free space when the cluster is initially loaded or when a mass insert is done. CI-percent is a percentage of the amount of space to be preserved for adding new records and updating existing records with an increase in the length of the record. Since a CI is split when it becomes full, the CA might also need to be split when it is filled by CIs created by a CI split. The empty space in the control interval and control area is available for data records that are updated and inserted after the cluster is initially loaded. This parameter applies only to key-sequenced clusters, and variable-length relative records with variable-length records. *CI-percent* is the number of bytes that is equal to, or slightly less than, the percentage value of *CI-percent*. *CA-percent* is the number of control intervals equal to, or less than, the percentage of *CA-percent*.

*CI-percent* and *CA-percent* must be equal to, or less than, 100. When you use FREESPACE(100 100), one data record is placed in each control interval used for data. One control interval in each control area is used for data (that is, one data record is stored in each control area when the data set is loaded). If you do not use FREESPACE, the default reserves no free space when the data set is loaded.

When you define the cluster using the RECORDS parameter, the amount of free space specified is not considered in the calculations to determine primary allocation.

**Abbreviation:**FSPC

**FRLOG(NONE|REDO)**

Specifies if VSAM batch logging can be performed for your VSAM data set. VSAM batch logging is available with CICS VSAM Recovery V3R1.

There is no default value for FRLOG. If FRLOG is left out, the data set cannot be used for VSAM batch logging. See the ALTER command for enabling VSAM batch logging after a data set is created.

**NONE**

Indicates that the data set can be used for VSAM batch logging. However, the function should be disabled. The LOGSTREAMID parameter indicates changes that are made by applications that are written to the MVS log

stream. Specifying FRLOG(NONE) implies that you may use the data set for RLS processing; omitting it indicates that RLS processing will not occur.

**REDO**
Enables the VSAM batch logging function for your VSAM data set. The LOGSTREAMID parameter indicates changes that are made by applications that are written to the MVS log stream. When specifying FRLOG(REDO), you must also specify LOGSTREAMID.

**Restrictions:**

1. If you do not want VSAM batch logging for your data set, do not specify the FRLOG parameter. If you specify FRLOG(NONE), the data set must support VSAM batch logging, but logging is not in effect.
2. If FRLOG is specified, the data set:
   - Must be SMS-managed
   - Cannot be LINEAR or a temporary data set

**INDEXED|LINEAR|NONINDEXED|NUMBERED**
Shows the type of data organization for the cluster.

If you want a data organization other than INDEXED (the default), you must explicitly use it with this parameter.

When a cluster is defined, you indicate whether the data is to be indexed (key-sequenced), nonindexed (entry-sequenced), numbered (relative record), or linear.

Certain parameters apply only to key-sequenced clusters, as noted in the description of each of these parameters.

Linear data set clusters are treated as ESDS clusters that must be processed using control interval access.

If you do not choose either the data organization or the MODEL parameter, your cluster defaults to key-sequenced (indexed).

If you want to define an entry-sequenced or a relative record cluster, you must specify the NONINDEXED, the NUMBERED, or the MODEL parameter.

The data organization you select must be consistent with other parameters you specify.

**INDEXED**
Shows that the cluster being defined is for key-sequenced data. If INDEXED is specified, an index component is automatically defined and cataloged. The data records can be accessed by key or by relative-byte address (RBA).

**Abbreviation:**IXD

**LINEAR**
Specifies that the cluster being defined is for linear data. Because linear data set clusters are treated as ESDS clusters that must be processed using control interval access, you can use most of the commands and parameters you use to manipulate ESDS clusters. There are two exceptions:

- Parameters that refer to logical records are not allowed (except RECORDS).
- Use partial printing by specifying the RBA syntax.

Space is allocated for a linear data set with the number of control intervals equal to the number of records.Linear data sets cannot be accessed for RLS processing. The LOG, LOGSTREAMID, and BWO parameters do not apply to linear data sets.

**Abbreviation:**LIN

**NONINDEXED**
Indicates that the cluster being defined is for entry-sequenced data. The data records can be accessed sequentially or by relative-byte address (RBA).

**Abbreviation:**NIXD

**NUMBERED**
Specifies that the cluster's data organization is for relative record data. A relative record cluster, which is similar to an entry-sequenced cluster, has fixed-length records or variable-length records that are stored in slots. The RECORDSIZE parameter determines if the records are fixed-length or variable-length. Empty slots hold space for records to be added later. The data records are accessed by relative record number (slot number).

**Abbreviation:**NUMD

**KEYS(***length offset***|64 0)**
KEYS gives information about the prime key field of a key-sequence data set's data records.

This parameter overrides any KEYS specification on the DATACLASS parameter.

This parameter applies only to key-sequenced clusters. The default is a key field of 64 bytes, beginning at the first byte (byte 0) of each data record.

The key field of the cluster's index is called the prime key to distinguish it from other keys, called alternate keys. See "Chapter 13. DEFINE ALTERNATEINDEX" on page 121 for more details on how to choose alternate indexes for a cluster.

When the data record spans control intervals, the record's key field must be within the part of the record that is in the first control interval.

*length offset*
specifies the length of the key and its displacement (in bytes) from the beginning of the record. The sum of length plus offset cannot exceed the length of the shortest record. The length of the key can be 1 to 255 bytes.

**LOG(NONE|UNDO|ALL)**
Establishes whether the sphere to be accessed with VSAM record-level sharing (RLS) is recoverable or non-recoverable. It also indicates whether a recovery log is available for the sphere. LOG applies to all components in the VSAM sphere.

If BWO, LOG, OR LOGSTREAMID are specified (or an RLS cell exists for the data set), access from DFSMS/MVS 1.2 or lower level systems is denied.

If LOG is specified in the SMS data class, the value defined is used as the data set definition, unless it has been previously defined with an explicitly specified or modeled DEFINE attribute.

LOG cannot be used with LINEAR.

LOGSTREAMID cannot be used with LINEAR.

**NONE**
Indicates that neither an external backout nor a forward recovery capability is available for the sphere accessed in RLS mode. If you use this, RLS considers the sphere to be non-recoverable.

**UNDO**
Specifies that changes to the sphere accessed in RLS mode can be backed out using an external log. RLS considers the sphere recoverable when you use LOG(UNDO).

**ALL**
Specifies that changes to the sphere accessed in RLS mode can be backed out and forward recovered using an external log. RLS considers the sphere recoverable when you use LOG(ALL). When you specify LOG(ALL), you must also specify the LOGSTREAMID parameter.

VSAM RLS allows concurrent read/update sharing for nonrecoverable spheres through commit (CICS) and non-commit protocol applications. For a recoverable sphere, an application that does not have an external log (BATCH) cannot open the sphere through RLS access.

**LOGSTREAMID(***logstream***)**
Gives the name of the forward recovery log stream. It applies to all components in the VSAM sphere.

If BWO, LOG, OR LOGSTREAMID are specified (or an RLS cell exists for the data set), access from DFSMS/MVS 1.2 or lower level systems is denied.

If LOGSTREAMID is specified in the SMS data class, the value defined is used as the data set definition, unless it has been previously defined with an explicitly specified or modeled DEFINE attribute.

*logstream*
The name of the forward recovery log stream. This can be a fully qualified name up to 26 characters, including separators. If LOG(ALL) is specified, LOGSTREAMID(name) must be specified. For information about defining log streams for CICS use, see *CICS System Definition Guide*.

**Abbreviation:**LSID

**MANAGEMENTCLASS(***class***)**
*For SMS-managed data sets:* Specifies the name, 1-to-8 characters, of the management class for a new data set. Your storage administrator defines the names of the management classes you can use. If MANAGEMENTCLASS is not used, but STORAGECLASS is used or defaulted, MANAGEMENTCLASS is derived from automatic class selection (ACS). If MANAGEMENTCLASS is specified and STORAGECLASS is not specified **or** derived, the DEFINE is unsuccessful. If SMS is inactive and MANAGEMENTCLASS is specified, the DEFINE will be unsuccessful. MANAGEMENTCLASS cannot be listed as a subparameter of DATA or INDEX.

**Abbreviation:**MGMTCLAS

**MODEL(***entryname***[** *catname***])**
Specifies an existing entry to be used as a model for the entry being defined. See “Specifying Attribute Selection Order” on page 14 for information on how the system selects modeled attributes.

A VVDS cannot be modeled.

DATACLASS, MANAGEMENTCLASS, and STORAGECLASS attributes are not modeled.

You can use an existing cluster's entry as a model for the attributes of the cluster being defined. For details about how a model is used, see *z/OS DFSMS: Managing Catalogs*.

You can use some attributes of the model and override others by explicitly specifying them in the definition of the cluster or component. If you do not want to add or change any attributes, you need specify only the entry type (cluster, data, or index) of the model to be used and the name of the entry to be defined.

See "Specifying Attribute Selection Order" on page 14 for more information about the order in which the system selects an attribute.

When you use a cluster entry as a model for the cluster, the data and index entries of the model cluster are used as models for the data and index components of the cluster still to be defined, unless another entry is specified with the MODEL parameter as a subparameter of DATA or INDEX.

*entryname*
specifies the name of the cluster or component entry to be used as a model.

*catname*
names the model entry's catalog. You identify the catalog that contains the model entry if the model entry's catalog is not identified with a JOBCAT or STEPCAT DD statement, and is not the master catalog.

If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved. See "Catalog Selection Order for DEFINE" on page 10 for information about the order in which a catalog is selected when the catalog's name is not specified.

**OWNER(*ownerid*)**
Identifies the cluster's owner.

For TSO users, if the owner is not identified with the OWNER parameter, the TSO user's userid becomes the ownerid.

**RECATALOG|NORECATALOG**
Indicates whether the catalog entries for the cluster components are to be re-created from information in the VVDS.

**RECATALOG**
Recreates the catalog entries if valid VVDS entries are found on the primary VVDS volume. If they are not, the command ends.

Catalog entries can be re-created only in the catalog specified in the VVR except for entries that are swap space, page space, or SYS1 data sets.

The RECORDSIZE parameter is required when doing a DEFINE RECATALOG of a variable-length relative record data set (VRRDS).

Identification of RECATALOG requires that NAME, INDEXED, LINEAR, NONINDEXED, NUMBERED, and VOLUMES be used as they were when the cluster was originally defined. If you specify RECATALOG, you are not required to use CYLINDERS, RECORDS, or TRACKS.

If ATTEMPTS, AUTHORIZATION, CATALOG, CODE, , FOR, MODEL, OWNER, or TO parameters are used during the original define, they must be respecified with RECATALOG to restore their original values; otherwise, their default values are used.

When you use the TO parameter with RECATALOG, only the cluster's expiration date is updated. The DATA and INDEX components are not updated.

If the RACF user has ADSP specified, a profile is defined to RACF for the data set being recataloged.

If the cluster was SMS-managed, the volume serials should be the same as the volumes actually selected by SMS.

The catalog for the entries being re-created must have the same name as the catalog that contained the original entries.

**Abbreviation:** RCTLG

**NORECATALOG**
Indicates that the catalog entries are not re-created from VVDS entries. Catalog entries are created for the first time.

**Abbreviation:** NRCTLG

**RECORDSIZE(***average maximum***|***default***)**
Specifies the average and maximum lengths, in bytes, of the records in the data component. The minimum record size is 1 byte.

RECORDSIZE can be given as a parameter of either CLUSTER or DATA.

This parameter overrides the LRECL specification on the DATACLASS parameter.

For nonspanned records, the maximum record size + 7 cannot exceed the data component's control interval size (that is, the maximum nonspanned record size, 32761, + 7 equals the maximum data component control interval size, 32768).

When you use a record size that is larger than one control interval, you must also specify spanned records (SPANNED). The formula for the maximum record size of spanned records as calculated by VSAM is as follows:

```
MAXLRECL = CI/CA * (CISZ - 10)
```

where:
- MAXLRECL is the maximum spanned record size.
- CI/CA represents the number of control intervals per control area.
- CA is the number of control areas.
- CISZ is the quantity control interval size.

When you select NUMBERED, you identify a data set as a relative record data set. If you use NUMBERED and select the same value for average as for maximum, the relative records must be fixed-length. If you specify NUMBERED and select two different values for the average and maximum record sizes, the relative records can be variable-length. If you know that your relative records are fixed-length, however, be sure to define them as fixed-length. Performance is affected for relative record data sets defined as variable-length.Each variable-length relative record is increased internally in length by four.

When your records are fixed length, you can use the following formula to find a control interval size that contains a whole number (n) of records:

```
CISZ = (n x RECSZ) + 10
or
n = (CISZ - 10)
      RECSZ
```

If you select SPANNED or NUMBERED for your fixed-length records:

```
CISZ =(n x (RECSZ + 3)) + 4
or
n = (CISZ - 4)
    (RECSZ + 3)
```

where:

- ***n*** is the number of fixed-length records in a control interval.
- CISZ is the control interval size (see also the CONTROLINTERVALSIZE parameter).
- RECSZ is the average record size.

*default*
When SPANNED is used, the default is RECORDSIZE(4086 32600). Otherwise, the default is RECORDSIZE(4089 4089).

**Important:**

```
REC(sec) x RECSZ(avg) > RECSZ(max)
```

where:

– REC(sec) is the secondary space allocation quantity, in records.
– RECSZ(avg) is the average record size (default = 4086 or 4089 bytes).
– RECSZ(max) is the maximum record size (default = 4089 or 32600 bytes).

When the SPANNED record size default prevails (32600 bytes), the secondary allocation quantity should be at least 8 records.

**Restriction:** REPRO and EXPORT do not support data sets with record sizes greater than 32760.

**Abbreviation:**RECSZ

**REUSE|NOREUSE**
Specifies whether or not the cluster can be opened again and again as a reusable cluster.

If REUSE or NOREUSE is specified in the SMS data class, the value defined is used as the data set definition, unless it has been previously defined with an explicitly specified or modeled DEFINE attribute.

**REUSE**
Specifies that the cluster can be opened again and again as a reusable cluster. When a reusable cluster is opened, its high-used RBA is set to zero if you open it with an access control block that specifies the RESET attribute.

REUSE lets you create an entry-sequenced, key-sequenced, or relative record work file.

When you create a reusable cluster, you cannot build an alternate index to support it. Also, you cannot create a reusable cluster with key ranges (see the KEYRANGE parameter). Reusable data sets can be multivolume and can have up to 123 physical extents.

**Restriction:** If you select REUSE and your command also contains the keyword **UNIQUE**, you must remove the UNIQUE keyword or the DEFINE command will be unsuccessful.

**Abbreviation:**RUS

**NOREUSE**
Indicates that the cluster cannot be opened again as a new cluster.

**Abbreviation:**NRUS

**SHAREOPTIONS(***crossregion***[** *crosssystem***]|1 3)**
Shows how a component or cluster can be shared among users. However, SMS-managed volumes, and catalogs containing SMS-managed data sets, must not be shared with non-SMS systems. For a description of data set sharing, see *z/OS DFSMS: Using Data Sets*. To ensure integrity, you should be sure that share options specified at the DATA and INDEX levels are the same.

*crossregion*
Specifies the amount of sharing allowed among regions within the same system or within multiple systems using global resource serialization (GRS). Independent job steps in an operating system, or multiple systems in a GRS ring, can access a VSAM data set concurrently. For more information about GRS, see *z/OS MVS Planning: Global Resource Serialization*. To share a data set, each user must use DISP=SHR in the data set's DD statement. You can use the following options:

**OPT 1** The data set can be shared by any number of users for read processing, or the data set can be accessed by only one user for read and write processing. VSAM ensures complete data integrity for the data set. This setting does not allow any non-RLS access when the data set is already open for RLS processing. An RLS open will fail with this option if the data set is already open for any processing.

**OPT 2** The data set can be accessed by any number of users for read processing, and it can also be accessed by one user for write processing. It is the user's responsibility to provide read integrity. VSAM ensures write integrity by obtaining exclusive control for a control interval while it is being updated. An RLS open is not allowed while the data set is open for non-RLS output.

If the data set has already been opened for RLS processing, a non-RLS open for input is allowed; a non-RLS open for output fails. [3] If the data set is opened for input in non-RLS mode, an RLS open is allowed.

**OPT 3** The data set can be fully shared by any number of users. Each user is responsible for maintaining both read and write integrity for the data the program accesses. This setting does not allow any non-RLS access when the data set is already open for RLS processing. If the data set is opened for input in non-RLS mode, an RLS open is allowed.

**OPT 4** The data set can be fully shared by any number of users. For each request, VSAM refreshes the buffers used for direct processing. This setting does not allow any non-RLS access

3. You must apply APARs OW25251 and OW25252 to allow non-RLS read access to data sets already opened for RLS processing.

when the data set is already open for RLS processing. If the data set is opened for input in non-RLS mode, an RLS open is allowed.

As in SHAREOPTIONS 3, each user is responsible for maintaining both read and write integrity for the data the program accesses.

*crosssystem*

Specifies the amount of sharing allowed among systems. Job steps of two or more operating systems can gain access to the same VSAM data set regardless of the disposition indicated in each step's DD statement for the data set. However, if you are using GRS across systems or JES3, the data set might not be shared depending on the disposition of the system.

To get exclusive control of the data set's volume, a task in one system issues the RESERVE macro. The level of cross-system sharing allowed by VSAM applies only in a multiple operating system environment.

The cross-system sharing options are ignored by RLS processing. The values are:

**1** Reserved

**2** Reserved

**3** Specifies that the data set can be fully shared. With this option, each user is responsible for maintaining both read and write integrity for the data that user's program accesses. User programs that ignore write integrity guidelines can cause VSAM program checks, uncorrectable data set errors, and other unpredictable results. This option requires each user to be responsible for maintenance. The RESERVE and DEQ macros are required with this option to maintain data set integrity. (For information on using RESERVE and DEQ, see *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN* and *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU*.)

**4** Indicates that the data set can be fully shared. For each request, VSAM refreshes the buffers used for direct processing. This option requires that you use the RESERVE and DEQ macros to maintain data integrity while sharing the data set. Improper use of the RESERVE macro can cause problems similar to those described under SHAREOPTIONS 3. (For information on using RESERVE and DEQ, see *z/OS MVS Programming: Authorized Assembler Services Reference ALE-DYN* and *z/OS MVS Programming: Authorized Assembler Services Reference LLA-SDU*.) Output processing is limited to update, or add processing, or both that does not change either the high-used RBA or the RBA of the high key data control interval if DISP=SHR is specified.

To ensure data integrity in a shared environment, VSAM provides users of SHAREOPTIONS 4 (cross-region and cross-system) with the following assistance:

- Each PUT request immediately writes the appropriate buffer to the VSAM cluster's DASD space. That is, the buffer in the user's address space that

contains the new or updated data record, and the buffers that contain new or updated index records when the user's data is key-sequenced.

- Each GET request refreshes all the user's input buffers. The contents of each data and index buffer being used by the user's program is retrieved from the VSAM cluster's DASD.

**Abbreviation:**SHR

**SPANNED|NONSPANNED**

Specifies whether a data record is allowed to cross control interval boundaries.

If SPANNED or NONSPANNED is specified in the SMS data class, the value defined is used as the data set definition, unless it has been previously defined with an explicitly specified or modeled DEFINE attribute.

This parameter cannot be used when defining a linear data set cluster.

**SPANNED**

Specifies that, if the maximum length of a data record (as specified with RECORDSIZE) is larger than a control interval, the record is contained on more than one control interval. This allows VSAM to select a control interval size that is optimum for the DASD.

When a data record that is larger than a control interval is put into a cluster that allows spanned records, the first part of the record completely fills a control interval. Subsequent control intervals are filled until the record is written into the cluster. Unused space in the record's last control interval is not available to contain other data records.

**Attention:** Using this parameter for a variable-length relative record data set causes an error.

**Abbreviation:**SPND

**NONSPANNED**

Indicates that the record must be contained in one control interval. VSAM selects a control interval size that accommodates your largest record.

**Abbreviation:**NSPND

**SPEED|RECOVERY**

Specifies whether the data component's control areas are to be preformatted before alternate index records are loaded into them.

This parameter is only considered during the actual loading (creation) of a data set. Creation occurs when the data set is opened and the high-used RBA is equal to zero. After normal CLOSE processing at the completion of the load operation, the physical structure of the data set and the content of the data set extents are exactly the same. Any processing of the data set after the successful load operation is the same, and the specification of this parameter is not considered.

If you use RECOVERY, the initial load takes longer because the control areas are first written with either empty or software end-of-file intervals. These preformatted CIs are then updated, using update writes, with the alternate index records. SPEED is recommended, because the initial load is quicker.

**SPEED**

Does not preformat the data component's space.

If the initial load is unsuccessful, you must load the alternate index records again from the beginning, because VSAM cannot determine the location of your last correctly written record. VSAM cannot find a valid end-of-file indicator when it searches your alternate index records.

**RECOVERY**
Specifies that the data component's control areas are written with records that indicate end-of-file. When an alternate index record is written into a control interval, it is always followed by a record that identifies the record just written as the last record in the alternate index.

RECOVERY is a way to verify storage that is used on the device for each CA before the data is actually written.

**Abbreviation:**RCVY

**STORAGECLASS(***class***)**
*For SMS-managed data sets:* Gives the name, 1-to-8 characters of the storage class.

Your storage administrator defines the names of the storage classes you can use. A storage class is assigned either when you use STORAGECLASS, or an ACS routine selects a storage class for the new data set. The storage class provides the storage attributes that are specified on the UNIT and VOLUME operand for non-SMS managed data sets. Use the storage class to select the storage service level to be used by SMS for storage of the data set. If SMS is inactive and STORAGECLASS is used, the DEFINE will be unsuccessful.

STORAGECLASS cannot be selected as a subparameter of DATA or INDEX.

**Abbreviation:** STORCLAS

**TO(***date***)|FOR(***days***)**
Specifies the retention period for the cluster being defined. If neither TO nor FOR is used, the cluster can be deleted at any time. The MANAGEMENTCLASS maximum retention period, if selected, limits the retention period specified by this parameter.

For non-SMS-managed data sets, the correct retention period is reflected in the catalog entry. The VTOC entry cannot contain the correct retention period. Enter a LISTCAT command for the correct expiration date.

For SMS-managed data sets, the expiration date in the catalog is updated and the expiration date in the format-1 DSCB is changed. If the expiration date in the catalog does not agree with the expiration date in the VTOC, the VTOC entry overrides the catalog entry.

**TO(***date***)**
Specifies the date up to which to keep the cluster before it is allowed to be deleted. The date is given in the form [yy]yyddd, where yyyy is a four-digit year, yy is a two-digit year, and ddd is the three-digit (001 through 366) day of the year. Two-digit years are treated as if 19 is specified as the first two digits of yyyy. The dates (19)99365 and (19)99366 are considered never-expire dates.

For expiration dates of January 1, 2000, and later, you must use the form TO(yyyyddd).

The maximum yyyy is a four-digit year (through 2155) and the maximum ddd is a three digit day number from 000 through 365 for non leap years. For leap years, the maximum ddd is 366.

**FOR(*days*)**
Shows the number of days to keep the cluster being defined. The maximum number is 9999. If the number is 0 through 9998, the cluster is retained for the number of days; if the number is 9999, the cluster is retained indefinitely.

**WRITECHECK|NOWRITECHECK**
indicates whether the cluster or component is to be checked by a machine action called write check when a record is written into it.

The WRITECHECK setting is ignored when the data set is opened for RLS.

**WRITECHECK**
Shows that a record is written and then read, without data transfer, to test for the data check condition.

**Abbreviation:**WCK

**NOWRITECHECK**
Use when the cluster or component is not to be checked by a write check.

**Abbreviation:**NWCK

## Data and Index Components of a Cluster

You should use attributes separately for the cluster's data and index components. A list of the DATA and INDEX parameters is provided at the beginning of this section. These parameters are described in detail as parameters of the cluster as a whole. Restrictions are noted with each parameter's description.

## DEFINE CLUSTER Examples

The DEFINE CLUSTER command can perform the functions shown in the following examples.

### Define an SMS-Managed Key-Sequenced Cluster: Example 1

In this example, an SMS-managed key-sequenced cluster is defined. The DEFINE CLUSTER command builds a catalog entry and allocates space to define the key-sequenced cluster SMS04.KSDS01.

```
//DEFINE   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
    DEFINE CLUSTER -
           (NAME (SMS04.KSDS01) -
           STORAGECLASS (FINCE02) -
           MANAGEMENTCLASS (MC1985) -
           DATACLASS (VSAMDB05))
/*
```

The parameters for this command are:

- STORAGECLASS specifies an installation-defined name of a storage class, FINCE02, to be assigned to this cluster.
- MANAGEMENTCLASS specifies an installation-defined name of a management class, MC1985, to be assigned to this cluster. Attributes of MANAGEMENTCLASS control the data set's retention, backup, migration, etc.
- DATACLASS specifies an installation-defined name of a data class, VSAMDB05, to be assigned to this cluster. Record size, key length and offset, space allocation, etc., are derived from the data class and need not be specified.

## Define an SMS-Managed Key-Sequenced Cluster Specifying Data and Index Parameters: Example 2

In this example, an SMS-managed key-sequenced cluster is defined. The SMS data class space allocation is overridden by space allocations at the data and index levels. The DEFINE CLUSTER command builds a catalog entry and allocates space to define the key-sequenced cluster SMS04.KSDS02.

```
//DEFINE   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
    DEFINE CLUSTER -
           (NAME (SMS04.KSDS02) -
           STORAGECLASS (FINCE02) -
           MANAGEMENTCLASS (MC1985) -
           DATACLASS (VSAMDB05)) -
         DATA -
           (MEGABYTES (10 2)) -
         INDEX -
           (KILOBYTES (25 5))
/*
```

The parameters for this command are as follows:

- STORAGECLASS is an installation defined name of a storage class, FINCE02, to be assigned to the cluster.
- MANAGEMENTCLASS is an installation defined name of a management class, MC1985, to be assigned to the cluster. Attributes associated with a management class control the cluster's retention, backup, migration, etc.
- DATACLASS is an installation defined name of a data class, VSAMDB05, assigned to the cluster. Record size, key length and offset, etc., are derived from the data class and need not be specified. If MAXVOLUMES or the space parameters (MEGABYTES and KILOBYTES) were not specified, the values in the data class would be used.

The DATA and INDEX parameters are:

- MEGABYTES, used for DATA, allocates a primary space of 10 megabytes to the data component. A secondary space of 2 megabytes is specified for extending the data component.
- KILOBYTES, used for INDEX, allocates a primary space of 25 kilobytes to the index component. A secondary space of 5 kilobytes is specified for extending the index component.

## Define a Key-Sequenced Cluster Specifying Data and Index Parameters: Example 3

In this example, a key-sequenced cluster is defined. The DATA and INDEX parameters are specified and the cluster's data and index components are explicitly named. This example assumes that an alias name VWX is defined for the catalog RSTUCAT1. This naming convention causes VWX.MYDATA to be cataloged in RSTUCAT1.

```
//DEFCLU1  JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
    DEFINE CLUSTER -
           (NAME(VWX.MYDATA) -
           VOLUMES(VSER02) -
           RECORDS(1000 500)) -
         DATA -
```

```
            (NAME(VWX.KSDATA) -
            KEYS(15 0) -
            RECORDSIZE(250 250) -
            FREESPACE(20 10) -
            BUFFERSPACE(25000)  ) -
          INDEX -
            (NAME(VWX.KSINDEX) -
          CATALOG (RSTUCAT1)
/*
```

The DEFINE CLUSTER command builds a cluster entry, a data entry, and an index entry to define the key-sequenced cluster VWX.MYDATA. The parameters for the cluster as a whole are:

- NAME indicates that the cluster's name is VWX.MYDATA.
- VOLUMES is used when the cluster is to reside on volume VSER02.
- RECORDS specifies that the cluster's space allocation is 1000 data records. The cluster is extended in increments of 500 records. After the space is allocated, VSAM calculates the amount required for the index and subtracts it from the total.

In addition to the parameters specified for the cluster as a whole, DATA and INDEX parameters specify values and attributes that apply only to the cluster's data or index component. The parameters specified for the data component of VWX.MYDATA are:

- NAME indicates that the data component's name is VWX.KSDATA.
- KEYS shows that the length of the key field is 15 bytes and that the key field begins in the first byte (byte 0) of each data record.
- RECORDSIZE specifies fixed-length records of 250 bytes.
- BUFFERSPACE verifies that a minimum of 25 000 bytes must be provided for I/O buffers. A large area for I/O buffers can help to improve access time with certain types of processing. For example, with direct processing if the high-level index can be kept in virtual storage, access time is reduced. With sequential processing, if enough I/O buffers are available, VSAM can perform a read-ahead, thereby reducing system overhead and minimizing rotational delay.
- FREESPACE specifies that 20% of each control interval and 10% of each control area are to be left free when records are loaded into the cluster. After the cluster's records are loaded, the free space can be used to contain new records.

The parameters specified for the index component of VWX.MYDATA are:

- NAME specifies that the index component's name is VWX.KSINDEX.
- CATALOG specifies the catalog name.

## Define a Key-Sequenced Cluster and an Entry-Sequenced Cluster: Example 4

In this example, two VSAM clusters are defined. The first DEFINE command defines a key-sequenced VSAM cluster, VWX.EXAMPLE.KSDS1. The second DEFINE command defines an entry-sequenced VSAM cluster, KLM.EXAMPLE.ESDS1. In both examples, it is assumed that alias names, VWX and KLM, have been defined for user catalogs RSTUCAT1 and RSTUCAT2, respectively.

```
//DEFCLU2  JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
    DEFINE CLUSTER -
```

```
          (NAME(VWX.EXAMPLE.KSDS1) -
          MODEL(VWX.MYDATA) -
          VOLUMES(VSER02) -
          NOIMBED  )
    DEFINE CLUSTER -
          (NAME(KLM.EXAMPLE.ESDS1) -
          RECORDS(100 500) -
          RECORDSIZE(250 250) -
          VOLUMES(VSER03) -
          NONINDEXED  )
/*
```

The first DEFINE command builds a cluster entry, a data entry, and an index entry to define the key-sequenced cluster VWX.EXAMPLE.KSDS1. Its parameters are:

- NAME specifies the name of the key-sequenced cluster, VWX.EXAMPLE.KSDS1. The cluster is defined in the user catalog for which VWX has been established as an alias.
- MODEL identifies VWX.MYDATA as the cluster to use as a model for VWX.EXAMPLE.KSDS1. The attributes and specifications of VWX.MYDATA that are not otherwise specified with the DEFINE command parameters are used to define the attributes and specifications of VWX.EXAMPLE.KSDS1. VWX.MYDATA is located in the user catalog for which VWX has been established as an alias.
- VOLUMES specifies that the cluster is to reside on volume VSER02.
- NOIMBED specifies that space is not to be allocated for sequence-set control intervals within the data component's physical extents.

The second DEFINE command builds a cluster entry and a data entry to define an entry-sequenced cluster, KLM.EXAMPLE.ESDS1. Its parameters are:

- NAME specifies the name of the entry-sequenced cluster, KLM.EXAMPLE.ESDS1. The cluster is defined in the user catalog for which KLM has been established as an alias.
- RECORDS specifies that the cluster space allocation is 100 records. When the cluster is extended, it is extended in increments of 500 records.
- RECORDSIZE specifies that the cluster records are fixed length (the average record size equals the maximum record size) and 250 bytes long.
- VOLUMES specifies that the cluster is to reside on volume VSER03.
- NONINDEXED specifies that the cluster is to be an entry-sequenced cluster.

## Define a Relative Record Cluster in a Catalog: Example 5

In this example, a relative record cluster is defined.

```
//DEFCLU4  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DEFINE CLUSTER -
          (NAME(EXAMPLE.RRDS1) -
          RECORDSIZE(100 100) -
          VOLUMES(VSER01) -
          TRACKS(10 5) -
          NUMBERED) -
        CATALOG(USERCAT)
/*
```

### DEFINE CLUSTER

The DEFINE CLUSTER command builds a cluster entry and a data entry to define the relative record cluster EXAMPLE.RRDS1 in the user catalog. The DEFINE CLUSTER command allocates ten tracks for the cluster's use. The command's parameters are:

- NAME specifies that the cluster's name is EXAMPLE.RRDS1.
- RECORDSIZE specifies that the records are fixed-length, 100 byte records. Average and maximum record length must be equal for a fixed-length relative record data set, but not equal for a variable-length RRDS.
- VOLUMES specifies that the cluster is to reside on volume VSER01. This example assumes that the volume is already cataloged in the user catalog, USERCAT.
- TRACKS specifies that 10 tracks are allocated for the cluster. When the cluster is extended, it is to be extended in increments of 5 tracks.
- NUMBERED specifies that the cluster's data organization is to be relative record.
- CATALOG specifies the catalog name.

## Define a Reusable Entry-Sequenced Cluster in a Catalog: Example 6

In this example, a reusable entry-sequenced cluster is defined. You can use the cluster as a temporary data set. Each time the cluster is opened, its high-used RBA can be reset to zero.

```
//DEFCLU5  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DEFINE CLUSTER -
           (NAME(EXAMPLE.ESDS2) -
           RECORDSIZE(2500 3000) -
           SPANNED -
           VOLUMES(VSER03) -
           CYLINDERS(2 1) -
           NONINDEXED -
           REUSE -
           CATALOG(RSTUCAT2)
/*
```

The DEFINE CLUSTER command builds a cluster entry and a data entry to define the entry-sequenced cluster, EXAMPLE.ESDS2. The DEFINE CLUSTER command assigns two tracks for the cluster's use. The command's parameters are:

- NAME specifies that the cluster's name is EXAMPLE.ESDS2.
- RECORDSIZE specifies that the records are variable length, with an average size of 2500 bytes and a maximum size of 3000 bytes.
- SPANNED specifies that data records can cross control interval boundaries.
- VOLUMES specifies that the cluster is to reside on volume VSER03.
- CYLINDERS specifies that two cylinders are to be allocated for the cluster's space. When the cluster is extended, it is to be extended in increments of 1 cylinder.
- NONINDEXED specifies that the cluster's data organization is to be entry-sequenced. This parameter overrides the INDEXED parameter.
- REUSE specifies that the cluster is to be reusable. Each time the cluster is opened, its high-used RBA can be reset to zero and it is effectively an empty cluster.
- CATALOG specifies that the cluster is to be defined in a user catalog, RSTUCAT2.

## Define a Key-Sequenced Cluster in a Catalog: Example 7

In this example, a key-sequenced cluster is defined. In other examples, an alternate index is defined over the cluster, and a path is defined that relates the cluster to the alternate index. The cluster, its alternate index, and the path entry are all defined in the same catalog, USERCAT.

```
//DEFCLU6  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DEFINE CLUSTER -
           (NAME(EXAMPLE.KSDS2)) -
         DATA -
           (RECORDS(500 100) -
           EXCEPTIONEXIT(DATEXIT) -
           ERASE -
           FREESPACE(20 10) -
           KEYS(6 4) -
           RECORDSIZE(80 100) -
           VOLUMES(VSER01) ) -
         INDEX -
           (RECORDS(300 300) -
           VOLUMES(VSER01) ) -
         CATALOG(USERCAT)
/*
```

The DEFINE CLUSTER command builds a cluster entry, a data entry, and an index entry to define the key-sequenced cluster, EXAMPLE.KSDS2. The DEFINE CLUSTER command allocates space separately for the cluster's data and index components.

The parameter that applies to the cluster is NAME which specifies that the cluster's name is EXAMPLE.KSDS2.

The parameters that apply only to the cluster's data component are enclosed in the parentheses following the DATA keyword:

- RECORDS specifies that an amount of tracks equal to at least 500 records is to be allocated for the data component's space. When the data component is extended, it is to be extended in increments of tracks equal to 100 records.
- EXCEPTIONEXIT specifies the name of the exception exit routine, DATEXIT, that is to be processed if an I/O error occurs while a data record is being processed.
- ERASE specifies that the cluster's data is to be erased (overwritten with binary zeros) when the cluster is deleted.
- FREESPACE specifies the amounts of free space to be left in the data component's control intervals (20%) and the control areas (10% of the control intervals in the control area) when data records are loaded into the cluster.
- KEYS specifies the location and length of the key field in each data record. The key field is 6 bytes long and begins in the fifth byte (byte 4) of each data record.
- RECORDSIZE specifies that the cluster's records are variable length, with an average size of 80 bytes and a maximum size of 100 bytes.
- VOLUMES specifies that the cluster is to reside on volume VSER01.

The parameters that apply only to the cluster's index component are enclosed in the parentheses following the INDEX keyword:

- RECORDS specifies that an amount of tracks equal to at least 300 records is to be allocated for the index component's space. When the index component is extended, it is to be extended in increments of tracks equal to 300 records.

- VOLUMES specifies that the index component is to reside on volume VSER01.

The CATALOG parameter specifies that the cluster is to be defined in a user catalog, USERCAT4.

## Define an Entry-Sequenced Cluster Using a Model: Example 8

In this example, two entry-sequenced clusters are defined. The attributes of the second cluster defined are modeled from the first cluster.

```
//DEFCLU7  JOB    ...
//STEP1   EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
    DEFINE CLUSTER -
           (NAME(GENERIC.A.BAKER) -
           VOLUMES(VSER02) -
           RECORDS(100 100) -
           RECORDSIZE(80 80) -
           NONINDEXED ) -
         CATALOG(USERCAT4)
    DEFINE CLUSTER -
           (NAME(GENERIC.B.BAKER) -
           MODEL(GENERIC.A.BAKER USERCAT4)) -
         CATALOG(USERCAT4)
/*
```

The first DEFINE CLUSTER command defines an entry-sequenced cluster, GENERIC.A.BAKER. Its parameters are:

- NAME specifies the name of the entry-sequenced cluster, GENERIC.A.BAKER.
- VOLUMES specifies that the cluster is to reside on volume VSER02.
- RECORDS specifies that the cluster's space allocation is 100 records. When the cluster is extended, it is extended in increments of 100 records.
- RECORDSIZE specifies that the cluster's records are fixed length (the average record size equals the maximum record size) and 80 bytes long.
- NONINDEXED specifies that the cluster is entry-sequenced.
- CATALOG specifies that the cluster is to be defined in the USERCAT4 catalog.

The second DEFINE CLUSTER command uses the attributes and specifications of the previously defined cluster, GENERIC.A.BAKER, as a model for the cluster still to be defined, GENERIC.B.BAKER. A list of the parameters follows:

- NAME specifies the name of the entry-sequenced cluster, GENERIC.B.BAKER.
- MODEL identifies GENERIC.A.BAKER, cataloged in user catalog USERCAT4, as the cluster to use as a model for GENERIC.B.BAKER. The attributes and specifications of GENERIC.A.BAKER that are not otherwise specified with the DEFINE command's parameters are used to define the attributes and specifications of GENERIC.B.BAKER.
- CATALOG specifies that the cluster is to be defined in the USERCAT4 catalog.

## Define a VSAM Volume Data Set: Example 9

In this example, a VVDS is explicitly defined. The cluster is named using the restricted VVDS name format 'SYS1.VVDS.Vvolser'.

```
//DEFCLU8  JOB    ...
//STEP1   EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
    DEFINE CLUSTER -
           (NAME(SYS1.VVDS.VVSER03) -
```

```
          VOLUMES(VSER03) -
          NONINDEXED -
          CYLINDERS(1 1)) -
        CATALOG(USERCAT4)
/*
```

This DEFINE CLUSTER command defines an entry-sequenced cluster that is used as a VVDS. The parameters are:

- NAME specifies the name of a VVDS, 'SYS1.VVDS.Vvolser', SYS1.VVDS.VVSER03.
- VOLUMES specifies that the cluster is to reside on volume VSER03. Only one volume serial can be specified.
- NONINDEXED specifies that the cluster is entry-sequenced.
- CYLINDERS specifies that the cluster's space allocation is 1 cylinder. When the cluster is extended, it is extended in increments of 1 cylinder.
- CATALOG specifies that the cluster is to be defined in the USERCAT4 catalog.

## Define a Relative Record Data Set with Expiration Date Beyond 1999: Example 10

In this example, an entry-sequenced cluster is defined specifying an expiration date beyond the year 1999, using the TO parameter.

```
//DEFCLU8  JOB    ...
//STEP1   EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE CLUSTER -
           (NAME(EXAMPLE.RRDS1) -
           RECORDSIZE(100 100) -
           VOLUMES(VSER01) -
           TRACKS(10 5) -
           NUMBERED -
           TO(2015012) ) -
        CATALOG(USERCAT)
/*
```

The DEFINE CLUSTER command builds a cluster entry and a data entry to define the relative record cluster, EXAMPLE.RRDS1, in the user catalog, USERCAT. The DEFINE CLUSTER command allocates ten tracks for the cluster's use. The expiration date is set to January 12, 2015. The parameters are:

- NAME specifies that the cluster's name is EXAMPLE.RRDS1.
- RECORDSIZE specifies that the records are fixed-length, 100-byte records. Average and maximum record length must be equal for a fixed-length relative record data set, but not equal for a variable-length RRDS.
- VOLUMES specifies that the cluster is to reside on volume VSER01.
- TRACKS specifies that ten tracks are allocated for the cluster. When the cluster is extended, it is to be extended in increments of five tracks.
- NUMBERED specifies that the cluster's data organization is to be relative record.
- TO specifies that the retention period is set to expire January 12, 2015. Note that the year (2015) is specified as a four-digit number and concatenated with the day (012). A four-digit year must be specified when the expiration date is beyond 1999. The retention period could also have been set by using the FOR parameter, followed by the number of days the cluster is to be retained.
- CATALOG specifies that the cluster is to be defined in a user catalog, USERCAT.

## Define a Linear Data Set Cluster in a Catalog: Example 11

In this example, a linear data set cluster is defined in a catalog.

```
//DEFLDS   JOB    ...
//STEP1   EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
    DEFINE CLUSTER -
           (NAME(EXAMPLE.LDS01) -
           VOLUMES(VSER03) -
           TRACKS(20 10) -
           LINEAR -
         CATALOG(USERCAT)
/*
```

The DEFINE CLUSTER command builds a cluster entry and a data entry to define the linear data set cluster EXAMPLE.LDS01. The parameters are:

- NAME specifies that the cluster's name is EXAMPLE.LDS01.
- VOLUMES specifies that the cluster is to reside on volume VSER03.
- TRACKS specifies that 20 tracks are allocated for the cluster's space. When the cluster is extended, it is to be extended in increments of 10 tracks.
- LINEAR specifies that the cluster's data organization is to be linear.
- CATALOG specifies that the cluster is to be defined in a user catalog, USERCAT.

# Chapter 15. DEFINE GENERATIONDATAGROUP

The DEFINE GENERATIONDATAGROUP command creates a catalog entry for a generation data group (GDG). For information on generation data group wrapping rules, see *z/OS MVS JCL User's Guide*. The syntax of the DEFINE GENERATIONDATAGROUP is:

| | |
|---|---|
| **DEFINE** | **GENERATIONDATAGROUP**<br>**(NAME(***entryname***)**<br>**LIMIT(***limit***)**<br>**[EMPTY\|NOEMPTY]**<br>**[OWNER(***ownerid***)]**<br>**[SCRATCH\|NOSCRATCH]**<br>**[TO(***date***)\|FOR(***days***)])**<br>**[CATALOG(***catname***)]** |

DEFINE can be abbreviated: DEF

## DEFINE GENERATIONDATAGROUP Parameters

### Required Parameters

**GENERATIONDATAGROUP Abbreviation: GDG**
Specifies that a generation data group (GDG) entry is to be defined. A GDG can contain both SMS- and non-SMS-managed generation data sets. A generation data set (GDS) cannot be a VSAM data set. If you create a GDG and its catalog is on an SMS-managed volume, you should remove any dependencies on pattern DSCBs. See *z/OS DFSMS: Using Data Sets* for information about GDGs and GDSs.

**NAME(***entryname***)**
Specifies the name of the GDG being defined.

**LIMIT(***limit***) Abbreviation: LIM**
Specifies the maximum number, from 1 to 255, of GDSs that can be associated with the GDG being defined.

### Optional Parameters

**CATALOG(***catname***) Abbreviation: CAT**
Identifies the catalog in which the generation data group is to be defined. If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved. See "Catalog Selection Order for DEFINE" on page 10 for the order in which a catalog is selected when the catalog's name is not specified.

*catname*
Specifies the name of the catalog.

**EMPTY|NOEMPTY**
Specifies what action is to be taken when the maximum number of GDSs for the GDG is exceeded and another GDS is to be cataloged. The disposition of the data set's DSCB in the volume's VTOC is determined with the SCRATCH|NOSCRATCH parameter. For SMS-managed data sets, the data set's NVR is also determined with the SCRATCH|NOSCRATCH parameter. If

© Copyright IBM Corp. 1973, 2002

NOSCRATCH is specified for an SMS-managed GDS, the GDS is uncataloged from its GDG base and can be recataloged outside its GDG base as an SMS non-VSAM entry with the roll-off status.

**EMPTY Abbreviation: EMP**
Specifies that all the generation data sets are to be uncataloged when the maximum is exceeded (each data set's non-VSAM entry is automatically deleted from the catalog).

**NOEMPTY Abbreviation: NEMP**
Specifies that only the oldest generation data set is to be uncataloged when the maximum is reached.

**OWNER(*ownerid*)**
Identifies the generation data set's owner.

For TSO users, if the owner is not identified with the OWNER parameter, the TSO userid is the default ownerid.

**SCRATCH|NOSCRATCH**
Specify whether or not a generation data set's DSCB is to be deleted from the volume's VTOC when the data set is uncataloged (that is, when its entry is deleted from the catalog automatically, as described under EMPTY|NOEMPTY, or explicitly as a result of a user entered DELETE request). For SMS-managed GDSs, SCRATCH|NOSCRATCH specifies if the NVR is to be removed from the VVDS when the data set is uncataloged.

You can override the SCRATCH|NOSCRATCH attribute when issuing the DELETE command.

**SCRATCH Abbreviation: SCR**
Specifies that the generation data set's DSCB is to be deleted from the volume's VTOC when the generation data set is uncataloged. Direct access device space management (DADSM) removes the data set's DSCB from the VTOC, erases the data set's space on the volume, and makes the space available to other system users. The generation data set ceases to exist. For SMS-managed GDSs, SCRATCH also specifies that the NVR is to be removed from the VVDS when the data set is uncataloged.

**NOSCRATCH Abbreviation: NSCR**
Specifies that the generation data set's DSCB is not to be removed from the volume's VTOC when the generation data set is uncataloged. The data set's DSCB in the volume's VTOC is left intact and can be used to locate the data set. Your program, however, can process the data set by using a JCL DD statement to describe and allocate the data set.

**TO(*date*)|FOR(*days*)**
Specifies the retention period for the GDG being defined.

**TO(*date*)**
Specifies the date up to which to keep the GDG before it is allowed to be deleted. The date appears in the form [yy]yyddd, where yyyy is a four-digit year, yy is a two-digit year, and ddd is the three-digit (001 through 366) day of the year. Two-digit years are treated as if "19" is specified as the first two digits of yyyy. The dates (19)99365 and (19)99366 are considered never-expire dates.

For expiration dates of January 1, 2000, and later, you must use the form TO(yyyyddd). The maximum yyyy is a four-digit year (through 2155) and the maximum ddd is a three digit day number from 000 through 365 for non leap years. For leap years, the maximum ddd is 366.

**FOR(*days*)**
Specifies the number of days to keep the GDG being defined. The maximum number that can be specified is 9999. If the number specified is 0 through 9998, the GDG is retained for the number of days specified; if the number is 9999, the GDG is retained indefinitely. If neither TO nor FOR is specified, the GDG can be deleted at any time.

## DEFINE GENERATIONDATAGROUP Examples

### Define a Generation Data Group and a Generation Data Set within it: Example 1

In this example, a generation data group is defined in the master catalog. Next, a generation data set is defined within the GDG by using JCL statements.

```
//DEFGDG1  JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//GDGMOD   DD     DSNAME=GDG01,DISP=(,KEEP),
//         SPACE=(TRK,(0)),UNIT=DISK,VOL=SER=VSER03,
//         DCB=(RECFM=FB,BLKSIZE=2000,LRECL=100)
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
    DEFINE GENERATIONDATAGROUP -
           (NAME(GDG01) -
           EMPTY -
           NOSCRATCH -
           LIMIT(255) )
/*
//DEFGDG2  JOB    ...
//STEP1    EXEC   PGM=IEFBR14
//GDGDD1   DD     DSNAME=GDG01(+1),DISP=(NEW,CATLG),
//         SPACE=(TRK,(10,5)),VOL=SER=VSER03,
//         UNIT=DISK
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
/*
```

Job control language statement:

- GDGMOD DD, which describes the GDG. When the scheduler processes the DD statement, no space is allocated to GDG01.

The model DSCB must exist on the GDGs catalog volume.

The DEFINE GENERATIONDATAGROUP command defines a GDG base catalog entry, GDG01. Its parameters are:

- NAME specifies the name of the GDG, GDG01. Each GDS in the group will have the name GDG01.GxxxxVyy, where “xxxx” is the generation number and “yy” is the version number.
- EMPTY specifies that all data sets in the group are to be uncataloged by VSAM when the group reaches the maximum number of data sets (as specified by the LIMIT parameter) and one more GDS is added to the group.
- NOSCRATCH specifies that when a data set is uncataloged, its DSCB is not to be removed from its volume’s VTOC. Therefore, even if a data set is uncataloged, its records can be accessed when it is allocated to a job step with the appropriate JCL DD statement.
- LIMIT specifies that the maximum number of GDGs in the group is 255. The LIMIT parameter is required.

Use the second job, DEFGDG2, to allocate space and catalog a GDS in the newly-defined GDG. The job control statement GDGDD1 DD specifies a GDS in the GDG.

## Use Access Method Services to Define a GDG and JCL to Define a GDS in that GDG: Example 2

In this example, a GDG is defined with access method services commands and then JCL is used to define a GDS into the newly defined GDG. It is assumed that the storage administrator has created a storage class named GRPVOL1 and a data class named ALLOCL01.

```
//DEFGDG   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     DEFINE GENERATIONDATAGROUP -
            (NAME(ICFUCAT1.GDG02) -
            EMPTY -
            NOSCRATCH -
            LIMIT(255))
/*
//DEFGDS   JOB  ...
//STEP1    EXEC PGM=IEFBR14
//GDSDD1   DD   DSN=ICFUCAT1.GDG02(+1),DISP=(NEW,CATLG),
//         SPACE(TRK,(5,2)),STORCLAS=GRPVOL1,DATACLAS=ALLOC01
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
/*
```

**Restriction:** Because the GDG is created in SMS-managed storage and its catalog, ICFUCAT1, is on an SMS volume, any dependencies on pattern DSCBs should be removed.

The DEFINE GENERATIONDATAGROUP command defines a GDG base catalog entry, ICFUCAT1.GDG02. A description of the parameters follows:

- NAME specifies the name of the GDG, ICFUCAT1.GDG02.
- EMPTY specifies that all data sets in the group are to be uncataloged by VSAM when the group reaches the maximum number of data sets (as specified by the LIMIT parameter) and one more GDS is added to the group.
- NOSCRATCH specifies that when a data set is uncataloged, its DSCB is not to be removed from its volume's VTOC. Therefore, even if a data set is uncataloged, its records can be accessed when it is allocated to a job step with the appropriate JCL DD statement.
- LIMIT, a required parameter, specifies that the maximum number of GDGs in the group is 255.

The second job, DEFGDS, allocates space and catalogs a GDS into the newly-defined GDG, ICFUCAT1.GDG02. The job control statement GDSDD1 DD specifies that an SMS GDS, ICFUCAT1.GDG02(+1), is allocated by the scheduler with a storage class GRPVOL1,

# Chapter 16. DEFINE NONVSAM

Using Access Method Services, you can set up jobs to execute a sequence of commands with a single invocation of IDCAMS. Modal command execution is based on the success or failure of prior commands.

The DEFINE NONVSAM command defines a catalog entry for non-VSAM data sets or a collection of objects. The syntax of the DEFINE NONVSAM command is:

| | |
|---|---|
| **DEFINE** | **NONVSAM**<br>**(NAME(***entryname***)**<br>**DEVICETYPES(***devtype***[** *devtype***...])**<br>**VOLUMES(***volser***[** *volser***...])**<br>**[COLLECTION]**<br>**[FILESEQUENCENUMBERS(***number***[** *number***...])]**<br>**[OWNER(***ownerid***)]**<br>**[RECATALOG\|NORECATALOG]**<br>**[TO(***date***)\|FOR(***days***)])**<br>**[CATALOG(***catname***)]** |

DEFINE can be abbreviated: DEF

A sequence of commands commonly used in a single job step includes DELETE—DEFINE—REPRO or DELETE—DEFINE—BLDINDEX. You can specify either a DD name or a data set name with these commands. When you refer to a DD name, however, allocation occurs at job step initiation. This could result in a job failure if a command such as REPRO follows a DELETE—DEFINE sequence that changes the location (volser) of the data set. A failure can occur with either SMS-managed data sets.

To avoid potential failures with a modal command sequence in your IDCAMS job:

- Specify the data set name instead of the DD name; or
- Use a separate job step to perform any sequence of commands (for example, REPRO, IMPORT, BLDINDEX, PRINT, or EXAMINE) that follow a DEFINE command.

**Restrictions:**

1. You cannot rename a non-VSAM data set that contains an indirect volume serial number.
2. You cannot use %SYS conversion for any non-VSAM data set that contains an indirect volume serial number.
3. The program or function that deletes and re-catalogs non-VSAM data sets that contain indirect volume serial numbers cannot re-catalog them with indirect volume serial numbers.

For information on generation data group wrapping rules, see *z/OS MVS JCL Reference*.

## DEFINE NONVSAM Parameters

The DEFINE NONVSAM command uses the following parameters.

© Copyright IBM Corp. 1973, 2002

## Required Parameters

**NONVSAM**
specifies that a non-VSAM non-SMS-managed data set is to be defined or that a collection of objects is to be defined. The collection of objects you define are SMS-managed. To define a non-VSAM SMS-managed data set that is not a collection of objects, use either the ALLOCATE command or JCL.

**Abbreviation:** NVSAM

**NAME(*entryname*)**
specifies the name of the non-VSAM data set or collection of objects being defined. The entryname is the name that appears in the catalog; it is the name used in all future references to the data set. The entryname must be unique within the catalog in which it is defined.

You identify a GDS with its GDG name followed by the data set's generation and version numbers (GDGname.GxxxxVyy). Relative generation numbers (that is, GDGname(+1)) cannot be used with the entryname when you use the DEFINE NONVSAM command to catalog GDS and attach it to a GDG. The update or higher RACF authority to the GDG is required. The GDG must exist before the GDS is defined.

**Exception:** For OAM, the entryname is the name of the OAM collection.

To define an OAM collection, you must specify both the COLLECTION and RECATALOG parameters.

**DEVICETYPES(*devtype*[ *devtype*...])**
specifies the device types of the volumes containing the non-VSAM data set being defined. If the non-VSAM data set resides on different device types, the device types must be specified in the same order as the volume serial numbers listed in the VOLUMES parameter.

You can specify a generic device name that is supported by your system, for example, 3390. See "Device Type Translate Table" on page 346 for a list of generic device types.

**Important:**

Do not specify an esoteric device group such as SYSDA, because allocation can be unsuccessful if:

- Input/output configuration is changed by adding or deleting one or more esoteric device groups.
- The esoteric definitions on the creating and using systems do not match when the catalog is shared between the two systems.
- The data set was cataloged on a system not defined with the Hardware Configuration Definition (HCD), but used on a system that is defined with HCD.

If you expect to change the device type of the system residence volume, you can code DEVICETYPES(0000) and this field is resolved at LOCATE, and DELETE time to the device type. This will allow you to use the non-VSAM data sets without having to recatalog them to point to the new volume. When you code DEVICETYPES(0000) you must also code VOLUMES(******), or an error will result.

You can code DEVICETYPES(0000) if the VOLUMES parameter specifies an indirect volume serial ('******'), or an extended indirect volume serial (a system symbol). A value of DEVICETYPES(0000) will cause the actual device type to be determined from the current system residence volume (or its logical extension) at the time the catalog entry is retrieved. DEVICETYPES(0000) is only valid with an indirect volume serial specification in the VOLUMES parameter.

In addition to the above, if you are using the symbolic form of volume serials, the volume must be mounted and online at the time the catalog entry is retrieved from the catalog. If it is not, the catalog request will be terminated with a return and reason code.

**Exception:** DEVICETYPES is not applicable for an OAM non-VSAM entry and is ignored if specified.

**Abbreviation:** DEVT

**VOLUMES(***volser***[** *volser***...])**
specifies the volumes to contain the non-VSAM data set. VOLUMES is required when you define a non-OAM non-VSAM data set.

There are two special forms of the VOLUMES parameter that can be provided, and they are referred to as the indirect volume serial forms. They result in the system dynamically resolving the volume serial to the system residence (or its logical extension) serial number when the catalog entry is retrieved. It is not resolved when the DEFINE NONVSAM is processed. This allows you to later change the volume serial number(s) of the system residence volume (or its logical extensions) without having to recatalog the non-VSAM data sets on those volumes.

The two special forms are:
1. VOLUMES(******)
2. VOLUMES(&xxxxx) where "&xxxxx" is a symbol contained in the SYS1.PARMLIB IEASYMXX member that was specified at IPL time. The symbol name is intended to represent the volume that is a logical extension of the system residence volume. The symbol name must be specified as a single, simple (not substringed) symbol of no more than six characters including the leading ampersand. If a symbol is intended to represent a six-character volume serial number, the symbol must be six characters long and the ending period must be omitted. As an example:
   VOLUMES(&SYSR2)

   If &SYSR2 has been defined at IPL by an entry in the IEASYMxx member, the value of that symbol will be used when this catalog entry is retrieved from the catalog. If the symbol is not defined, the value returned for the volume serial will be &SYSR2.

   IBM recommends the use of the symbol &SYSR2 for the first logical extension to the system reference volume, &SYSR3 for the second, and so on.

**Exception:** VOLUMES is not applicable for an OAM non-VSAM entry and is ignored if specified.

If you code `VOLUMES(******)`, then the system dynamically resolves this to the system residence volume serial number whenever the catalog entry is used. It

is not resolved when the DEFINE NONVSAM is processed. This allows you to later change the volume serial number of system residence volume without also having to recatalog the non-VSAM data sets on that volume.

**Abbreviation:** VOL

Use RACF commands to specify an ERASE attribute in a generic or discrete profile for a non-VSAM data set. Use of this attribute renders all allocated DASD tracks unreadable before space on the volume is made available for reallocation. Refer to the appropriate RACF publications for information about how to specify and use this facility.

## Optional Parameters

**CATALOG(***catname***)**
identifies the catalog in which the non-VSAM data set, or OAM object is to be defined. See "Catalog Selection Order for DEFINE" on page 10 for the order in which a catalog is selected when the catalog's name is not specified.

To specify catalog names for SMS-managed data sets, you must have authority from the RACF STGADMIN.IGG.DIRCAT facility class. See "Storage Management Subsystem (SMS) Considerations" on page 2 for more information.

*catname*
specifies the name of the catalog in which the entry is to be defined.

**Abbreviation:** CAT

**COLLECTION**
specifies that the entry being defined is an Object Access Method (OAM) entry. This parameter is required when you define an OAM entry. If you use COLLECTION, you must also specify the RECATALOG parameter.

**Abbreviation:** COLLN

**FILESEQUENCENUMBERS(***number***[** *number***...])**
specifies the file sequence number of the non-VSAM data set being defined. FILESEQUENCENUMBERS is not applicable for an OAM non-VSAM entry and is ignored if specified.

This number indicates the position of the file being defined with respect to other files on the tape. If the data set spans volumes or if more than one volume is specified, you must specify a file sequence number for each volume. Either 0 or 1 indicates the first data set on the tape volume. The default is 0.

**Abbreviation:** FSEQN

**OWNER(***ownerid***)**
identifies the owner of the non-VSAM data set, or OAM object.

**RECATALOG|NORECATALOG**
specifies whether the catalog entries for the non-VSAM data set are to be re-created or are to be created for the first time. If RACF is installed, RACF access authority, defined under SMS, is required.

**Exception:** If OWNER is not specified, the TSO userid is the default ownerid.

**RECATALOG**
specifies that the catalog entries are re-created if valid VVDS entries are found on the primary VVDS volume. If valid VVDS entries are not found on

the primary VVDS volume, the command ends. RECATALOG can be specified only for an SMS-managed data set.

Catalog entries can be re-created only in the catalog specified in the NVR except for entries that are swap space, page space, or SYS1 data sets. In a multihost environment, non-SYS1 IPL data sets that are SMS-managed cannot be recataloged to a different catalog from the one specified in the NVR. SMS-managed IPL data sets must be SYS1 data sets to be shared in a multihost environment.

The VOLUMES and DEVICETYPES parameters are required, specified as they were when the data set was originally defined. If the CATALOG, OWNER, FOR, TO, or FILESEQUENCENUMBERS parameters were specified for the original define, they should be respecified with RECATALOG.

**Exception:** RECATALOG must be specified when you use the COLLECTION parameter. DEFINE RECATALOG COLLECTION is intended to be used to rebuild catalog entries.

**Abbreviation:** RCTLG

**NORECATALOG**
creates the catalog entries for the first time.

**Abbreviation:** NRCTLG

**TO(*date*)|FOR(*days*)**
specifies the retention period for the non-VSAM data set being defined. If neither a TO nor FOR is specified, the non-VSAM data set can be deleted at any time.

For non-SMS-managed non-VSAM data sets, the correct retention period is selected in the catalog entry. The VTOC entry might not contain the correct retention period. Issue a LISTCAT command to see the correct expiration date.

For SMS-managed data sets, the expiration date in the catalog is updated and the expiration date in the format-1 DSCB is changed. Should the expiration date in the catalog not agree with the expiration date in the VTOC, the VTOC entry overrides the catalog entry. In this case, issue a LISTVTOC to see the correct expiration date.

**Exception:** TO|FOR is not applicable for an OAM non-VSAM entry and is ignored if specified. With OAM, a never-expire retention is assigned to the entry which then requires you to specify the PURGE parameter of the DELETE command to delete the OAM non-VSAM entry.

**TO(*date*)**
specifies the date up to which to keep the non-VSAM data set before it is allowed to be deleted. The date is specified in the form [yy]yyddd, where yyyy is a four-digit year, yy is a two-digit year, and ddd is the three-digit (001 through 366) day of the year. Two-digit years are treated as if “19” is specified as the first two digits of yyyy. The dates (19)99365 and (19)99366 are considered never-expire dates.

For expiration dates of January 1, 2000, and later, you must use the form TO(yyyyddd). The maximum yyyy is a four-digit year (through 2155) and the maximum ddd is a three digit day number from 000 through 365 for non leap years. For leap years, the maximum ddd is 366.

**FOR(*days*)**
specifies the number of days to keep the non-VSAM data set being defined. The maximum number that can be specified is 9999.

If the number specified is 0 through 9998, the data set is retained for the number of days specified; if the number is 9999, the data set is retained indefinitely.

# DEFINE NONVSAM Examples

The DEFINE NONVSAM command can perform the functions shown in the following examples.

## Define a Non-VSAM Data Set with the RECATALOG Parameter: Example 1

This example defines an existing SMS-managed non-VSAM data set with the RECATALOG parameter.

```
//DEFNVSM  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     DEFINE NONVSAM -
          (NAME(EXAMPLE.NONVSAM3) -
          DEVICETYPE(3380) -
          VOLUMES(VSER01) -
          RECATALOG)
  /*
```

The parameters are:

- NAME specifies the name of the non-VSAM data set, EXAMPLE.NONVSAM3.
- DEVICETYPE specifies the type of device that contains the non-VSAM data sets, an IBM 3380 Direct Access Storage. This parameter is required because RECATALOG is specified.
- VOLUMES specifies the volume, VSER01, that contains the SMS-managed non-VSAM data sets. This parameter is also required because RECATALOG is specified.
- RECATALOG specifies that the catalog entries are to be re-created. This assumes that valid VVDS entries are found on the primary VVDS volume, and the data set is SMS-managed. If either of these assumptions is not true, the command will be unsuccessful.

  It is also assumed that CATALOG, FILESEQUENCENUMBER, OWNER, TO and FOR were not specified for the original define. If any of these parameters were specified for the original define, they should be respecified in this example containing RECATALOG.

## Define a Non-VSAM Data Set: Example 2

In this example, two existing non-VSAM data sets are defined in a catalog, USERCAT4. The DEFINE NONVSAM command cannot be used to create a non-VSAM data set because the command does not allocate space.

```
//DEFNVS   JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE NONVSAM -
           (NAME(EXAMPLE.NONVSAM) -
           DEVICETYPES(3380) -
```

```
          VOLUMES(VSER02) ) -
        CATALOG(USERCAT4/USERMRPW)
    DEFINE NONVSAM -
          (NAME(EXAMPLE.NONVSAM2) -
          DEVICETYPES(3380) -
          VOLUMES(VSER02) ) -
        CATALOG(USERCAT4)
/*
```

Both DEFINE NONVSAM commands define a non-VSAM data set in catalog USERCAT4. The parameters are:

- NAME specifies the name of the non-VSAM data sets, EXAMPLE.NONVSAM and EXAMPLE.NONVSAM2.
- DEVICETYPES specifies the type of device that contains the non-VSAM data sets, an IBM 3380 Direct Access Storage Device.
- VOLUMES specifies the volume that contains the non-VSAM data sets, VSER02.
- CATALOG identifies the catalog that is to contain the non-VSAM entries, USERCAT4.

# Chapter 17. DEFINE PAGESPACE

The DEFINE PAGESPACE command defines an entry for a page space data set. The syntax of the DEFINE PAGESPACE command is:

```
DEFINE    PAGESPACE
            (NAME(entryname)
            {CYLINDERS(primary)|
              KILOBYTES(primary)|
              MEGABYTES(primary)|
              RECORDS(primary)|
              TRACKS(primary)}
            VOLUME(volser)
            [DATACLASS(class)]
            [FILE(ddname)]
            [MANAGEMENTCLASS(class)]
            [MODEL(entryname [catname])]
            [OWNER(ownerid)]
            [RECATALOG|NORECATALOG]
            [STORAGECLASS(class)]
            [SWAP|NOSWAP]
            [TO(date)|FOR(days)]
          [CATALOG(catname)]
```

The parameter **VOLUME** can also be specified as **VOLUMES**.

DEFINE can be abbreviated: DEF

## DEFINE PAGESPACE Parameters

### Required Parameters

**PAGESPACE**
specifies that a page space is to be defined.

**Abbreviation:** PGSPC

**NAME(*entryname*)**
specifies the name of the page space being defined.

**CYLINDERS(*primary*)|**

**KILOBYTES(*primary*)|**

**MEGABYTES(*primary*)|**

**RECORDS(*primary*)|**

**TRACKS(*primary*)**
specifies the amount of space that is to be allocated. This parameter is optional if the volume is managed by SMS. If it is specified, it overrides the DATACLASS space specification. If it is not specified, it must be modeled or defaulted by SMS. If it cannot be determined, the DEFINE is unsuccessful.

© Copyright IBM Corp. 1973, 2002

If you specify KILOBYTES or MEGABYTES the amount of space allocated is the minimum number of tracks or cylinders required to contain the specified number of kilobytes or megabytes.

If RECORDS or TRACKS is specified, the quantity specified is rounded up to the nearest cylinder and the space is allocated in cylinders.

To maintain device independence, do not specify the TRACKS or CYLINDERS parameters. If TRACKS or CYLINDERS is specified for an SMS-managed pagespace, space is allocated on the volume selected by SMS in units equivalent to the device default geometry.

The amount of space need not be specified if the RECATALOG parameter is specified.

To determine the exact amount of space allocated, list the page space's catalog entry, using the LISTCAT command.

If you do not specify the MODEL parameter, you must specify one, and only one, of the following parameters: CYLINDERS, KILOBYTES, MEGABYTES, RECORDS, or TRACKS.

*primary*
Specifies the amount of space that is to be allocated to the page space. After the primary extent is full, the page space is full. The page space cannot extend onto secondary extents. The maximum number of paging slots for each page space is 16M.

**Abbreviations:** CYL, KB, MB, REC, and TRK

**VOLUME(*volser*)**
specifies the volume that contains the page space. If you do not specify the MODEL parameter, or if the page space is not SMS-managed, VOLUME must be specified as a parameter of PAGESPACE.

VOLUME can be specified or modeled for a data set that is to be SMS-managed, but be aware that the volume specified might not be used and, in some cases, can result in an error. If VOLUME is not specified for an SMS-managed data set, SMS selects the volume. See *z/OS DFSMSdfp Storage Administration Reference*for information about SMS volume selection.

Nonspecific volumes are indicated for an SMS-managed data set by coding an * for each volume serial. SMS then determines the volume serial.

The VOLUME parameter interacts with other DEFINE PAGESPACE parameters. Ensure that the volumes you specify for the page space are consistent with the page space's other attributes:

- The volume must contain enough unallocated space to satisfy the page space's space requirement.
- The volume information supplied with the DD statement pointed to by FILE must be consistent with the information specified for the page space.

**Abbreviation:** VOL

## Optional Parameters

**CATALOG(*catname*)**
When the CATALOG parameter identifies a non-SMS-managed user catalog, you must also supply a STEPCAT or JOBCAT DD statement to describe and allocate the user catalog. See "Catalog Selection Order for DEFINE" on page 10 for the order in which catalogs are selected.

To specify catalog names for SMS-managed data sets, you must have authority from the RACF STGADMIN.IGG.DIRCAT facility class. See “Storage Management Subsystem (SMS) Considerations” on page 2 for more information.

*catname*
Specifies the name of the catalog.

**Abbreviation:** CAT

**CODE(*code*)**
specifies a code name for the page space. If an attempt is made to access a password-protected entry without a password, the code name is used in a prompting message; the code enables the operator or TSO terminal user to be prompted for the password without disclosing the name of the entry. If code is not specified and an attempt is made to access a page space entry that is password-protected without supplying a password, the operator or TSO terminal user is prompted with the name of the entry.

This parameter is effective only when the cluster’s or component’s master password is not null.

**DATACLASS(*class*)**
specifies the name, 1-to-8 characters, of the data class for the data set. It provides the allocation attributes for new data sets.

Your storage administrator defines the data class. However, you can override the parameters defined for DATACLASS by explicitly specifying other attributes. See “Specifying Attribute Selection Order” on page 14 for the order of precedence (filtering) the system uses to select which attribute to assign.

DATACLASS parameters apply to both SMS-managed and non-SMS-managed data sets. If DATACLASS is specified and SMS is inactive, DEFINE is unsuccessful.

**Abbreviation:** DATACLAS

**FILE(*ddname*)**
specifies the name of the DD statement that identifies the device and volume to be allocated to the page space. If the FILE parameter is not specified and the volume is physically mounted, the volume identified with the VOLUME parameter is dynamically allocated. The volume must be mounted as permanently resident or reserved.

**MANAGEMENTCLASS(*class*)**
specifies, for SMS-managed data sets only, the 1-to-8 character name of the management class for a new data set. Your storage administrator defines the names of the management classes you can specify. If MANAGEMENTCLASS is not specified, but STORAGECLASS is specified or defaulted, MANAGEMENTCLASS is derived from automatic class selection (ACS). If MANAGEMENTCLASS is specified and STORAGECLASS is not specified or derived, the DEFINE is unsuccessful. If SMS is inactive and MANAGEMENTCLASS is specified, the DEFINE is unsuccessful.

**Abbreviation:** MGMTCLAS

**MODEL(*entryname*) [ *catname*])**
Specifies that an existing page space entry is to be used as a model for the entry being defined. It is possible to use an already defined page space as a model for another page space. When one entry is used as a model for another, its attributes are copied as the new entry is defined.

You can use some attributes of the model and override others by explicitly specifying them in the definition of the page space. If you do not want to add or change any attributes, you need specify only the entry type (page space) of the model to be used and the name of the entry to be defined.

See “Specifying Attribute Selection Order” on page 14 for more information about the order in which the system selects an attribute.

*entryname*
Specifies the name of the page space entry to be used as a model.

*catname*
Specifies the name of the catalog in which the entry to be used as a model is defined. You identify the catalog that contains the model entry if:

- The model entry’s catalog is not identified with a JOBCAT or STEPCAT DD statement, and is not the master catalog.

**OWNER(*ownerid*)**
specifies the identification of the owner of the page space.

**RECATALOG|NORECATALOG**
*For integrated catalog facility catalogs only,* specifies whether the catalog entries for the cluster components are to be re-created or are to be created for the first time.

**RECATALOG**
If RECATALOG is specified, the catalog entries are re-created if valid VVDS entries are found on the primary VVDS volume. If valid VVDS entries are not found on the primary VVDS volume, the command ends. For information on resolving VVDS problems, see the text on deleting VVDS records in *z/OS DFSMS: Managing Catalogs* .

Specification of RECATALOG requires that the NAME and VOLUMES parameters be specified as they were when the cluster was originally defined.

The CYLINDERS|RECORDS|TRACKS parameter is not required if RECATALOG is specified.

If the ATTEMPTS, AUTHORIZATION, CATALOG, CODE, FOR, MODEL, OWNER or TO parameters were specified during the original define, they must be respecified with RECATALOG to restore their original values; otherwise, their default values are used.

**Abbreviation:** RCTLG

**NORECATALOG**
If NORECATALOG is specified, the integrated catalog facility catalog entries are created for the first time.

**Abbreviation:** NRCTLG

**STORAGECLASS(*class*)**
For SMS-managed data sets: Specifies the name, 1-to-8 characters, of the storage class.

Your storage administrator defines the names of the storage classes you can specify. A storage class is assigned if you use STORAGECLASS or an ACS routine selects a storage class for the new data set.

The storage class provides the storage attributes that are specified on the UNIT and VOLUME operand for non-SMS-managed data sets. Use the storage class

to specify the storage service level to be used by SMS for storage of the data set. If SMS is inactive and STORAGECLASS is specified, the DEFINE is unsuccessful.

**Abbreviation:** STORCLAS

**SWAP|NOSWAP**
specifies whether page space is defined for local system queue area (LSQA) pages or for pageable private area pages. (Auxiliary storage management separates private area address space pages into LSQA pages and pageable private area pages.)

**SWAP**
Specifies that the page space is a high-speed data set used during a swap operation to store and retrieve the set of LSQA pages owned by an address space.

**NOSWAP**
Indicates that the page space is a conventional page space used to record pageable private area pages.

**Abbreviation:** NSWAP

**TO(*date*)|FOR(*days*)**
specifies the retention period for the page space. If neither TO nor FOR is specified, the page space can be deleted at any time.

The expiration date in the catalog is updated and the expiration date in the format-1 DSCB is changed. Should the expiration date in the catalog not agree with the expiration date in the VTOC, the VTOC entry overrides the catalog entry.

The MANAGEMENTCLASS maximum retention period, if specified, limits the retention period specified by this parameter.

**TO(*date*)**
specifies the date up to which to keep the page space before it is allowed to be deleted. The date is specified in the form [yy]yyddd, where yyyy is a four-digit year, yy is a two-digit year, and ddd is the three-digit (001 through 366) day of the year. Two-digit years are treated as if “19” is specified as the first two digits of yyyy. The dates (19)99365 and (19)99366 are considered never-expire dates.

For expiration dates of January 1, 2000, and later, you must use the form TO(yyyyddd). The maximum yyyy is a four-digit year (through 2155) and the maximum ddd is a three digit day number from 000 through 365 for non leap years. For leap years, the maximum ddd is 366.

**FOR(*days*)**
specifies the number of days to keep the page space. The maximum number that can be specified is 9999. If the number specified is 0 through 9998, the page space is retained for the number of days specified; if the number is 9999, the page space is retained indefinitely.

## DEFINE PAGESPACE Examples

The DEFINE PAGESPACE command can perform the functions shown in the following examples.

## Define a NOSWAP Page Space: Example 1

```
//DEFPGSP1 JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//VOLUME   DD     VOL=SER=VSER05,UNIT=DISK,DISP=OLD
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE PAGESPACE -
           (NAME(SYS1.PAGE2) -
           CYLINDERS(10) -
           VOLUMES(VSER05)
          /*
```

Job control language statement:

- VOLUME DD describes the volume on which the data space is to be defined.

The DEFINE PAGESPACE command defines a page space. These are the parameters:

- NAME specifies the name of the page space, SYS1.PAGE2.
- CYLINDERS specifies that the page space is to occupy 10 cylinders. The page spaces are never extended.
- VOLUMES specifies that the page space is to reside on volume VSER05.

The page space defaults to NOSWAP.

## Define a SWAP Page Space: Example 2

```
//DEFPGSP2 JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     DEFINE PAGESPACE -
           (NAME(SYS1.PAGE1) -
           CYLINDERS(10) -
           VOLUMES(VSER05) -
           SWAP
          /*
```

The DEFINE PAGESPACE command defines a page space. These are the parameters:

- NAME specifies the name for the page space is SYS1.PAGE1.
- CYLINDERS specifies that the page space occupies 10 cylinders and cannot be extended.
- VOLUMES identifies the volume on which the page space is to reside. Because no DD statement describes the volume, an attempt is made to dynamically allocate the volume. Volume VSER05 must be mounted as permanently resident or reserved.
- SWAP specifies that the page space is used to store local system queue area (LSQA) pages.

# Chapter 18. DEFINE PATH

The DEFINE PATH command defines a path directly over a base cluster or over an alternate index and its related base cluster. The syntax of the DEFINE PATH command is:

| | |
|---|---|
| **DEFINE** | **PATH**<br>**(NAME(***entryname***)**<br>**PATHENTRY(***entryname***)**<br>**[MODEL(***entryname***[** *catname***])]**<br>**[OWNER(***ownerid***)]**<br>**[RECATALOG\|NORECATALOG]**<br>**[TO(***date***)\|FOR(***days***)]**<br>**[UPDATE\|NOUPDATE]**<br>**[CATALOG(***catname***)]** |

DEFINE can be abbreviated: DEF

## DEFINE PATH Parameters

The DEFINE PATH command uses the following parameters.

### Required Parameters

**PATH**
specifies that a path is to be defined or that a path entry is to be recataloged.

**NAME(***entryname***)**
specifies the path's name.

**PATHENTRY(***entryname***)**
when the path consists of an alternate index and its base clusters, entryname identifies the alternate index entry. When the path is opened to process data records, both the alternate index and the base cluster are opened.

When the path consists of a cluster without an alternate index, entryname identifies the cluster. You can define the path as though it were an alias for the cluster. This allows you to specify no-update access to the cluster, so that the upgrade set will not be required or updated when the cluster is opened (provided the open does not cause sharing of a control block structure specifying UPDATE). You can also establish protection attributes for the alternate name, separate from the protection attributes of the cluster.

Entry name must not identify a VVDS.

**Abbreviation:** PENT

### Optional Parameters

**CATALOG(***catname***)**
identifies the catalog that contains the entry of the cluster or alternate index named in the PATHENTRY parameter. See "Catalog Selection Order for DEFINE" on page 10 for the order in which a catalog is selected if the catalog's name is not specified.

© Copyright IBM Corp. 1973, 2002

*catname*
specifies the catalog's name.

If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved.

**Abbreviation:** CAT

**MODEL(***entryname***[** *catname***])**
Specifies an existing path entry that is to be used as a model for the path being defined. You can use some attributes of the model and override others by explicitly specifying them in the definition of the path. When you do not want to add or change any attributes, you specify only the entry type (PATH), the path's name, its alternate index's or cluster's name, and the model entry's name.

See "Specifying Attribute Selection Order" on page 14 for more information about the order in which the system selects an attribute.

*entryname*
names the entry to be used as a model. The *entryname* must name a path entry.

*catname*
names the model entry's catalog.

If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved. See "Catalog Selection Order for DEFINE" on page 10 for information about the order in which a catalog is selected when the catalog's name is not specified. Unless you have RACF authorization to the directed catalog facility, you should not specify catalog names for SMS-managed data sets. For TSO users, if the owner is not identified with the OWNER parameter, the TSO user's userid becomes the ownerid.

**OWNER(***ownerid***)**
specifies the identification of the path's owner.

**RECATALOG|NORECATALOG**
specifies whether a path entry is to be created for the first time or recataloged.

**RECATALOG**
specifies that a path entry is to be recataloged. This requires that the NAME and PATHENTRY parameters be specified as they were when the path was originally defined.

If ATTEMPTS, AUTHORIZATION, CATALOG, CODE, FOR, MODEL, OWNER, TO, or UPDATE|NOUPDATE parameters were specified during the original define, they must be respecified with RECATALOG to restore their original values; otherwise, their default values are used.

**Abbreviations:** RCTLG

**NORECATALOG**
specifies that a new path entry is to be created in a catalog.

**Abbreviation:** NRCTLG

**TO(***date***)|FOR(***days***)**
specifies the retention period for the path. The path is not automatically deleted when the expiration date is reached. When a retention period is not specified,

the path can be deleted at any time. The MANAGEMENTCLASS maximum retention period, if specified, limits the retention period specified by this parameter for SMS-managed data sets.

**TO(*date*)**
specifies the date up to which to keep the path before it is allowed to be deleted. The date is specified in the form [yy]yyddd, where yyyy is a four-digit year, yy is a two-digit year, and ddd is the three-digit (001 through 366) day of the year. Two-digit years are treated as if “19” is specified as the first two digits of yyyy. The dates (19)99365 and (19)99366 are considered never-expire dates.

For expiration dates of January 1, 2000, and later, you must use the form TO(yyyyddd). The maximum yyyy is a four-digit year (through 2155) and the maximum ddd is a three digit day number from 000 through 365 for non leap years. For leap years, the maximum ddd is 366.

**FOR(*days*)**
specifies the number of days to keep the path. The maximum number that can be specified is 9999. If the number specified is 0 through 9998, the path is retained for the number of days specified; if the number is 9999, the path is retained indefinitely.

**UPDATE|NOUPDATE**
specifies whether the base cluster’s upgrade set is to be allocated when the path is opened for processing.

The upgrade set is a group of alternate indexes associated with the base cluster. The alternate indexes are opened whenever the base cluster is opened.

**UPDATE**
specifies that, when records in the base cluster are modified or deleted, or when records are added to the base cluster, each alternate index in the base cluster’s upgrade set is modified to reflect the change in the cluster’s data, just as a key-sequenced cluster’s index is modified each time the cluster’s data changes.

**Abbreviation:** UPD

**NOUPDATE**
specifies that, when opening the path, the path’s base cluster is to be allocated and the base cluster’s upgrade set is not to be allocated.

You can specify the NOUPDATE attribute for the path even though the UPGRADE attribute is set for one of the base cluster’s alternate indexes.

When a path points to a base cluster that has a large upgrade set (that is, many alternate indexes are associated with the base cluster), and the path is defined with the NOUPDATE attribute, you can open the path, and consequently the base cluster, and none of the alternate indexes will be opened.

NOUPDATE will be overridden by opening the path, allowing sharing of a control block structure that permits UPDATE.

**Abbreviation:** NUPD

## DEFINE PATH Examples

The DEFINE PATH command can perform the functions shown in the following examples.

## Define a Path: Example 1

In this example, a path is defined. Previous examples illustrate the definition of the path's alternate index, EXAMPLE.AIX, and the alternate index's base cluster, EXAMPLE.KSDS2. The alternate index, path, and base cluster are defined in the same catalog, USERCAT.

```
//DEFPATH  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DEFINE PATH -
           (NAME(EXAMPLE.PATH) -
           PATHENTRY(EXAMPLE.AIX) -
          CATALOG(USERCAT)
 /*
```

The DEFINE PATH command builds a path entry to define the path EXAMPLE.PATH. A list of the command's parameters follows:

- NAME specifies that the path's name is EXAMPLE.PATH.
- PATHENTRY identifies the alternate index, EXAMPLE.AIX, that the path provides access to.
- CATALOG supplies the user catalog's name, USERCAT.

## Define a Path (Recatalog) in a Catalog: Example 2

In this example, a path previously defined and found damaged is redefined. The cluster and path are defined in the same catalog, USERCAT4.

```
//DEFPATHF JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DEFINE PATH -
           (NAME(EXAMPLE1.PATH) -
           PATHENTRY(EXAMPLE1.KSDS01) -
           RECATALOG) -
        CATALOG(USERCAT4)
/*
```

The DEFINE PATH command builds a path entry to redefine the path EXAMPLE1.PATH.

- NAME specifies that the path's name is EXAMPLE1.PATH.
- PATHENTRY identifies the cluster, EXAMPLE1.KSDS01, that the path provides access to.
- RECATALOG specifies that the path entry is to be redefined in the catalog record for EXAMPLE1.KSDS01.
- CATALOG supplies the user catalog's name, USERCAT4.

# Chapter 19. DEFINE USERCATALOG

The DEFINE USERCATALOG command defines a user catalog. When you use this command, you can specify attributes for the catalog as a whole and for the components of the catalog. The syntax of the DEFINE USERCATALOG command is:

**DEFINE USERCATALOG|MASTERCATALOG** (parameters) -
[**DATA**(parameters)] -
[**INDEX**(parameters)] -
[**CATALOG**(subparameters)]

| | |
|---|---|
| **DEFINE** | **USERCATALOG\|MASTERCATALOG**<br>**(NAME(***entryname***)**<br>**{CYLINDERS(***primary***[** *secondary***])\|**<br>**KILOBYTES(***primary***[** *secondary***])\|**<br>**MEGABYTES(***primary***[** *secondary***])\|**<br>**RECORDS(***primary***[** *secondary***])\|**<br>**TRACKS(***primary***[** *secondary***])}**<br>**VOLUME(***volser***)**<br>**[BUFFERSPACE(***size***\|3072)]**<br>**[BUFND(***number***)]**<br>**[BUFNI(***number***)]**<br>**[CONTROLINTERVALSIZE(***size***)]**<br>**[DATACLASS(***class***)]**<br>**[ECSHARING\|NOECSHARING]**<br>**[FILE(***ddname***)]**<br>**[FREESPACE(***CI-percent***[** *CA-percent***]\|0 0)]**<br>**[ICFCATALOG\|VOLCATALOG]**<br>**[LOCK\|UNLOCK]**<br>**[MANAGEMENTCLASS(***class***)]**<br>**[MODEL(***entryname***[** *catname***])]**<br>**[OWNER(***ownerid***)]**<br>**[RECORDSIZE(***average maximum***\|4086 32400)]**<br>**[SHAREOPTIONS(***crossregion***[***crosssystem***]\| 3 4)]**<br>**[STORAGECLASS(***class***)]**<br>**[STRNO(***number***\|2)]**<br>**[TO(***date***)\|FOR(***days***)]**<br>**[WRITECHECK\|NOWRITECHECK])**<br>**[DATA (**<br>**{CYLINDERS(***primary***[** *secondary***])\|**<br>**KILOBYTES(***primary***[** *secondary***])\|**<br>**MEGABYTES(***primary***[** *secondary***])\|**<br>**RECORDS(***primary***[** *secondary***])\|**<br>**TRACKS(***primary***[** *secondary***])}**<br>**[BUFFERSPACE(***size***)]**<br>**[BUFND(***number***)]**<br>**[CONTROLINTERVALSIZE(***size***)]**<br>**[FREESPACE(***CI-percent***[** *CA-percent***]\| 0 0)]**<br>**[RECORDSIZE(***average maximum***\| 4086 32400)]**<br>**[WRITECHECK\|NOWRITECHECK])]**<br>**[INDEX (**<br>**{CYLINDERS(***primary***[** *secondary***])\|**<br>**KILOBYTES(***primary***[** *secondary***])\|**<br>**MEGABYTES(***primary***[** *secondary***])\|** |

© Copyright IBM Corp. 1973, 2002

| | |
|---|---|
| | **RECORDS(***primary***[***secondary***])\|**<br>**TRACKS(***primary***[***secondary***])}**<br>**[BUFNI(***number***)]**<br>**[CONTROLINTERVALSIZE(***size***)]**<br>**[WRITECHECK\|NOWRITECHECK])]**<br>**[CATALOG(***mastercatname***)]** |

DEFINE can be abbreviated: DEF

# DEFINE USERCATALOG Parameters

The DEFINE USERCATALOG command uses the following parameters.

## Required Parameters

**USERCATALOG|MASTERCATALOG**
Specifies that a catalog is to be defined.

**USERCATALOGAbbreviation: UCAT**
specifies that a user catalog is to be defined. USERCATALOG is followed by the parameters specified for the catalog as a whole. For information about using an alias to identify a user catalog, see *z/OS DFSMS: Managing Catalogs*. The update or higher RACF authority to the master catalog is required.

**MASTERCATALOGAbbreviation: MCAT**
This keyword parameter is provided for compatibility with OS/VS1. Processing is identical for the MASTERCATALOG and USERCATALOG parameters. When you specify MASTERCATALOG, a user catalog is created. You can, however, establish a user catalog as a master catalog at IPL time. See *z/OS DFSMS: Managing Catalogs* for a description of this procedure.

**NAME(***entryname***)**
Specifies the name of the catalog being defined.

**CYLINDERS(***primary***[** *secondary***])| Abbreviations: CYL**

**KILOBYTES(***primary***[** *secondary***])|Abbreviations:KB**

**MEGABYTES(***primary***[** *secondary***])|Abbreviations:MB**

**RECORDS(***primary***[** *secondary***])| Abbreviations:REC**

**TRACKS(***primary***[** *secondary***]) Abbreviations:TRK**
Specifies the amount of space to be allocated from the volume's available space. You can specify the amount of space as a parameter of USERCATALOG, as a parameter of USERCATALOG and DATA, or as a parameter of USERCATALOG, DATA and INDEX.

This parameter is optional if the cluster is managed by SMS. If it is specified for an SMS-managed cluster, it will override the DATACLASS space specification. If it is not specified for an SMS-managed cluster, it can be modeled or defaulted by SMS. If it cannot be determined, the DEFINE will be unsuccessful.

If you specify KILOBYTES or MEGABYTES, the amount of space allocated is the minimum number of tracks or cylinders required to contain the specified number of kilobytes or megabytes.

To maintain device independence, do not specify the TRACKS or CYLINDERS parameters. If TRACKS or CYLINDERS is specified for an SMS-managed user catalog, space is allocated on the volumes selected by SMS in units equivalent to the device default geometry.

*z/OS DFSMS: Managing Catalogs* describes how space allocation differs depending on the parameters you specify. It also provides information about estimating the amount of space to be specified for a catalog.

*primary***[** *secondary***]**
Specifies the size of the primary and secondary extents to be allocated. After the primary extent is filled, the space can expand to include a maximum of 122 additional secondary extents if you have specified a secondary allocation amount. Secondary allocation should be specified in case the catalog has to be extended. If you specify a secondary space allocation greater than 4.0 gigabytes, the value is reset to the maximum value for that DASD device.

**Exception:** The abbreviations CYL, CYLINDER, REC, and RECORD are acceptable to access method services but cannot be used in TSO because the abbreviations do not have enough initial letters to make the keyword unique.

**VOLUME(***volser***) Abbreviation: VOL**
Specifies the volume that is to contain the catalog. VOLUME must be specified as a parameter of USERCATALOG, unless:
- You specify the MODEL parameter, or
- The data set is managed by SMS.

If the data set is SMS-managed, you should not request specific volume serial numbers with the VOLUME parameter. The ACS routines will assign the data set to a storage class containing attributes such as VOLUME and UNIT. You can allocate your data set to a *specific* volume serial number only if your storage administrator has selected GUARANTEED SPACE=YES in the storage class assigned to the data set. Only then can you specify volume serial numbers that will override the volume serial numbers used by SMS. However, if space is not available on the volumes with the serial numbers you specified, your request will be unsuccessful. See *z/OS DFSMSdfp Storage Administration Reference* for information about SMS volume selection.

You can choose to let SMS assign specific volume serial numbers to an SMS-managed data set by coding an * for each volume serial. SMS then determines the volume serial. If you omit *volser*, you get one volume.

If you designate both user-specified and SMS-specified volume serial numbers for an SMS-managed data set, the user-specified volume serials numbers (volsers) must be requested first in the command syntax. Catalogs can only reside on one volume.

The VOLUME parameter interacts with other DEFINE CATALOG parameters. Ensure that the volume you specify for the catalog is consistent with the catalog's other attributes:
- CYLINDERS, RECORDS, TRACKS: The volume contains enough unallocated space to satisfy the catalog's primary space requirement. Space on the volume might already be allocated to non-VSAM data sets and system data sets.

- FILE: The volume information supplied with the DD statement is consistent with the information specified for the catalog and its components.

A user catalog can be defined on a mass storage volume.

## Optional Parameters

**BUFFERSPACE(*size*|3072) Abbreviation: BUFSP or BUFSPC**
Provides the amount of space for buffers. The size you specify for the buffer space helps VSAM determine the size of the data component's and index component's control interval. If BUFFERSPACE is not coded, VSAM attempts to get enough space to contain two data set control intervals and, if the data set is key-sequenced, one index control interval.

The size specified cannot be less than enough space to contain two data component control intervals. If the data is key-sequenced, it should contain only one index control interval. If the specified size is less than VSAM requires for the buffers needed to run your job, the default BUFFERSPACE calculation overrides the size.

This is the default BUFFERSPACE calculation: Data control interval size x 2 + index control interval size.

*size*
Provides the amount of space, in bytes, for buffers. Size can be expressed in decimal (*n*), hexadecimal (X'*n*'), or binary (B'*n*') form. However, it must not exceed 16 776 704.

**BUFND(*number*) Abbreviation: BFND**
Specifies the number of I/O buffers VSAM is to use for transmitting data between virtual and auxiliary storage.

The size of the buffer is the size of the data component control interval. The minimum number you can specify is the number specified for STRNO plus 1.

Note that minimum buffer specification does not provide optimum sequential processing performance. Additional data buffers benefit direct inserts or updates during control area splits and will also benefit spanned record accessing.

*number*
The number of data buffers to be used. The minimum number allowed is 3; the maximum number allowed is 255.

**BUFNI(*number*) Abbreviation: BFNI**
Specifies the number of I/O buffers VSAM is to use for transmitting the contents of index entries between virtual and auxiliary storage for keyed access.

The size of the buffer is the size of the index control interval. The minimum number you can specify is the number specified for STRNO.

Additional index buffers will improve performance by providing for the residency of some or all the high-level index (index set records), thereby minimizing the number of high-level index records to be retrieved from DASD for key-direct processing.

*number*
The number of index buffers to be used. The minimum number allowed is 2 and the maximum number allowed is 255.

**CATALOG(*mastercatname*) Abbreviation: CAT**
Specifies the name and password of the master catalog.

Use the CATALOG parameter only if you need to provide the password for a master catalog that is password-protected. For information on moving a user catalog to a different system, see *z/OS DFSMS: Managing Catalogs*.

*mastercatname*
The name of the master catalog that is required when a user catalog is being defined.

**CONTROLINTERVALSIZE(*size*) Abbreviation: CISZ or CNVSZ.**
Specifies the size of the control interval for the catalog or component.

The size of the control interval depends on the maximum size of the data records and the amount of buffer space you provide.

If you do not code the CONTROLINTERVALSIZE, VSAM determines the size of control intervals. If you have not specified BUFFERSPACE and the size of your records permits, VSAM calculates the optimum control interval size for the data and index components. This is based partly on device characteristics.If the control interval size calculated by VSAM as required for the index component is greater than the value specified in the parameter, the value calculated by VSAM will be used.

*size*
The size of the data and index components of a catalog.

The maximum control interval size is 32 768 bytes.

You can specify a size from 512 to 8K in increments of 512 or from 8K to 32K in increments of 2K. K is 1024 in decimal notation. If you select a size that is not a multiple of 512 or 2048, VSAM chooses the next higher multiple.

Refer to *z/OS DFSMS: Using Data Sets* for a discussion of the relationship between control interval size and physical block size. The discussion also includes restrictions that apply to control interval size and physical block size.

**DATACLASS(*class*) Abbreviation: DATACLAS**
Specifies the name, 1-to-8 characters, of the data class. DATACLASS can be specified for SMS-managed and non-SMS-managed data sets. It provides the allocation attributes for new data sets. Your storage administrator defines the data class. However, you can override the parameters defined for DATACLASS by explicitly specifying other attributes. See “Specifying Attribute Selection Order” on page 14 for the order of precedence (filtering) the system uses to select the attributes to assign.

The record organization attribute of DATACLASS is not used by DEFINE USERCATALOG/MASTERCATALOG. If DATACLASS is specified and SMS is inactive, DEFINE will be unsuccessful. DATACLASS cannot be specified as a subparameter of DATA or INDEX.

**ECSHARING|NOECSHARING Abbreviation: ECSHR**
Indicate whether or not sharing the catalog can be performed via the coupling facility.

**ECSHARING**
Enhanced Catalog Sharing (ECS) is allowed. ECS is a catalog sharing method that makes use of a coupling facility to improve the performance of shared catalog requests. Please read about ECS in *z/OS DFSMS: Managing Catalogs* before enabling ECS for a catalog.

**NOECSHARING Abbreviation: NOECSHR**
Enhanced Catalog Sharing (ECS) is not allowed. This is the default. Catalog sharing will be performed, but the ECS sharing method will not be used.

**FILE(***ddname***)**
Specifies the name of the DD statement that identifies the device and volume to be used for the catalog. The DD statement should specify DISP=OLD to prevent premature space allocation on the volume. If FILE is not specified and the catalog's volume is physically mounted, the volume identified with the VOLUME parameter is dynamically allocated. The volume must be mounted as permanently resident or reserved.

**FREESPACE(***CI-percent***[** *CA-percent***]|0 0) Abbreviation: FSPC**
Specifies the amount of space that is to be left free when the catalog is loaded and after any split of control intervals (*CI-percent*) and control areas (*CA-percent*).

The empty space in the control interval and control area is available for data records that are updated and inserted after the catalog is initially loaded.

The amounts are specified as percentages. *CI-percent* translates into a number of bytes that is equal to, or slightly less than, the percentage value of *CI-percent*. *CA-percent* translates into a number of control intervals that is equal to, or less than, the percentage value of *CA-percent*.

*CI-percent* and *CA-percent*, must be equal to or less than 100. If you use FREESPACE(100 100), one data record is placed in each control interval used for data and one control interval in each control area is used for data (that is, one data record is stored in each control area when the data set is loaded).

When no FREESPACE value is coded, the default specifies that no free space is to be reserved when the data set is loaded.

**[ICFCATALOG|VSAMCATALOG|VOLCATALOG]**
Specify the type of catalog to be defined.

**ICFCATALOG Abbreviation: ICFCAT**
Defines an integrated catalog facility catalog.

**VOLCATALOG Abbreviation: VOLCAT**
Defines an integrated catalog facility tape volume catalog (VOLCAT). A VOLCAT can contain only tape library and tape volume entries. You can define either a general VOLCAT or a specific VOLCAT.

- A general VOLCAT is the default tape volume catalog. A general VOLCAT contains all tape library entries and any tape volume entries that do not point to a specific VOLCAT. Each system can have access to only one general VOLCAT. You must define the general VOLCAT prior to bringing the tape libraries online.

  The general VOLCAT must be in the form:
  XXXXXXXX.VOLCAT.VGENERAL

  where XXXXXXXX either defaults to SYS1 or to another high level qualifier specified by the LOAD*xx* member in SYS1.PARMLIB. For more information on changing the high-level qualifier for VOLCATs, see the section on bypassing SYSCATxx with LOADxx in *z/OS DFSMS: Managing Catalogs*.
- A specific VOLCAT is a tape volume catalog that contains a specific group of tape volume entries based on the tape volume serial numbers (tape volsers). A specific VOLCAT cannot contain tape library entries.

The specific VOLCAT must be in the form:
XXXXXXXX.VOLCAT.V*y*

– where XXXXXXXX either defaults to SYS1 or is another high-level qualifier specified by the LOAD*xx* member in SYS1.PARMLIB.
– where *y* represents the first character of a tape volser. A specific VOLCAT contains all the tape volume entries with volsers whose first character is equal to *y*. See “Tape Volume Names” on page 9 for a discussion of the naming restrictions for tape volume volsers.

**IMBED|NOIMBED**
IMBED|NOIMBED is no longer supported; if it is specified, it will be ignored and no message will be issued.

**LOCK|UNLOCK**
Controls the setting of the catalog lock attribute, and therefore checks access to a catalog. LOCK and UNLOCK can be specified only when the entryname identifies an integrated catalog facility catalog. UNLOCK is the default. Before you lock a catalog, review the information on locking catalogs in *z/OS DFSMS: Managing Catalogs*.

**LOCK**
Specifies that the catalog identified by entryname is to be defined with the lock attribute on. Defining the catalog with the lock on restricts catalog access to authorized personnel. Specification of this parameter requires read authority to the profile name, IGG.CATLOCK, with class type FACILITY. Integrated catalog facility catalogs are usually defined with the lock attribute on only after a DELETE RECOVERY during catalog recovery operations. Locking a catalog makes it inaccessible to all users without read authority to RACF facility class profile IGG.CATLOCK (including users sharing the catalog on other systems).

**UNLOCK**
Specifies that the catalog identified by entryname is to be defined with the lock attribute off. This is the default if LOCK|UNLOCK is not specified.

**MANAGEMENTCLASS(*class*) Abbreviation: MGMTCLAS**
*For SMS-managed data sets:* Specifies the name, 1-to-8 characters, of the management class. Your storage administrator defines the names of the management classes you can specify. If MANAGEMENTCLASS is not specified, but STORAGECLASS is specified or defaulted, MANAGEMENTCLASS is derived from automatic class selection (ACS). If MANAGEMENTCLASS is specified and SMS is inactive, DEFINE will be unsuccessful. MANAGEMENTCLASS cannot be specified as a subparameter of DATA or INDEX.

**MODEL(*entryname*[ *catname*])**
Specifies that an existing master or user catalog is to be used as a model for the user catalog being defined.

When one entry is used as a model for another, its attributes are copied as the new entry is defined. You can use some attributes of the model and override others by explicitly specifying them in the definition of the user catalog.

If a model is used, you must specify certain parameters even though no attributes are to be changed or added. The name of the user catalog to be defined and volume and space information must always be specified as parameters of USERCATALOG. See “Specifying Attribute Selection Order” on page 14 for information about the order in which the system selects an attribute.

STORAGECLASS and MANAGEMENTCLASS classes can be modeled. If DATACLASS exists for the entry being used as a model, it is ignored.

*entryname*
Specifies the name of the master or user catalog to be used as a model.

*catname*
specifies the name of the catalog to be used as a model. This parameter is required if the model catalog is neither the master catalog nor a catalog identified by a JOBCAT or STEPCAT DD statement.

**OWNER(***ownerid***)**
Specifies the identification of the owner of the catalog being defined.

**RECORDSIZE(***average maximum***|4086 32400) Abbreviation: RECSZ**
If you specify record size it is ignored, and no error message gets issued.

**REPLICATE|NOREPLICATE**
REPLICATE|NOREPLICATE is no longer supported; if it is specified, it will be ignored and no message will be issued.

**SHAREOPTIONS(***crossregion***[** *crosssystem***]|3 4) Abbreviation: SHR**
Specifies how a catalog can be shared among users. This specification applies to both the data and index components of the catalog.

*crossregion*
Specifies the amount of sharing allowed among regions within the same system or within multiple systems using global resource serialization (GRS). Independent job steps in an operating system or multiple systems in a GRS ring can access the catalog concurrently.

**1** Reserved

**2** Reserved

**3** Specifies that the catalog can be fully shared by any number of users. With this option, each user opening the catalog as a data set is responsible for maintaining both read and write integrity for the data the program accesses. User programs that ignore the write integrity guidelines can cause VSAM program checks, lost or inaccessible records, uncorrectable catalog errors, and other unpredictable results. This option places heavy responsibility on each user sharing the catalog.

**4** Reserved

*crosssystem*
Specifies the amount of sharing allowed among systems. Job steps of two or more operating systems can gain access to the same catalog. To get exclusive control of the catalog's volume, a task in one system issues the RESERVE macro. The level of cross-system sharing allowed by VSAM applies only in a multiple operating system environment. You can use:

**1** Reserved

**2** Reserved

**3** Specifies that the catalog is not being shared across systems. SHAREOPTIONS(3 3) would direct the catalog open process to unconditionally bypass the setting of the buffer invalidation indicator. Hence, even though the catalog resided on a shared DASD device, buffer invalidation would not occur. This performance option must

be selected only when the user can guarantee that the catalog is not being shared across multiple processors.

**4** Specifies that the catalog can be fully shared. The integrity of the buffers and control block structure is maintained by ICF catalog management.

**STORAGECLASS(*class*) Abbreviation: STORCLAS**
*For SMS-managed data sets:* Specifies the 1-to-8 character name of the storage class. Your storage administrator defines the names of the storage classes you can specify. Use the storage class to specify the storage service level to be used by SMS for storage of the catalog. If STORAGECLASS is specified and SMS is inactive, DEFINE will be unsuccessful.

STORAGECLASS cannot be specified as a subparameter of DATA or INDEX.

**STRNO(*number*|2)**
Specifies the number of requests (RPLs) requiring concurrent data set positioning that VSAM is to be prepared to accommodate.

*number*
The number of requests catalog administration must be prepared to accommodate. The minimum number allowed is 2 and the maximum number is 255.

**TO(*date*)|FOR(*days*)**
Specifies the retention period for the catalog being defined. If no value is coded, the catalog can be deleted whenever it is empty.

The MANAGEMENTCLASS maximum retention period, if specified, limits the retention period specified by this parameter.

For non-SMS-managed catalogs, the correct retention period is reflected in the catalog entry. The VTOC entry cannot contain the correct retention period. Enter a LISTCAT command to see the correct expiration date.

For SMS-managed catalogs, the expiration date in the catalog is updated and the expiration date in the format-1 DSCB is changed. Should the expiration date in the catalog not agree with the expiration date in the VTOC, the VTOC entry overrides the catalog entry. In this case, enter a LISTVTOC command to see the correct expiration date.

**TO(*date*)**
Specifies the date up to which to keep the catalog before it is allowed to be deleted. The date is shown in the form [yy]yyddd, where yyyy is a four-digit year, yy is a two-digit year, and ddd is the three-digit (001 through 366) day of the year. Two-digit years are treated as if “19” is specified as the first two digits of yyyy. The dates (19)99365 and (19)99366 are considered never-expire dates.

For expiration dates of January 1, 2000, and later, you must use the form TO(yyyyddd). The maximum yyyy is a four-digit year (through 2155) and the maximum ddd is a three digit day number from 000 through 365 for non leap years. For leap years, the maximum ddd is 366.

**FOR(*days*)**
Specifies the number of days to keep the catalog. The maximum number that can be specified is 9999. If the number specified is 0 through 9998, the catalog is retained for the number of days specified; if the number is 9999, the catalog is retained indefinitely.

**WRITECHECK|NOWRITECHECK**
Specify whether or not the catalog is to be checked by a direct access device operation called write check when a record is written to the device.

**WRITECHECK Abbreviation: WCK**
Specifies that a record is to be written and then read, without data transfer.

**NOWRITECHECK Abbreviation: NWCK**
Specifies that the catalog is not to be checked by a write Checks to see if a record has been written to the device.

## Data and Index Components of a User Catalog

Attributes can be specified separately for the catalog's data and index components. A list of the DATA and INDEX parameters is provided at the beginning of this section. These parameters are described in detail as parameters of the catalog as a whole. Restrictions are noted with each parameter's description.

## DEFINE USERCATALOG Examples

### Define an Integrated Catalog Facility User Catalog, Specifying SMS Keywords: Example 1

In this example, an SMS-managed integrated catalog facility user catalog is defined.

```
//DEFUCAT  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD  *
    DEFINE USERCATALOG -
           (NAME(USERCAT1) -
           ICFCATALOG -
           STRNO(3) -
           DATACLAS(VSDEF) -
           STORCLAS(SMSSTOR) -
           MGMTCLAS(VSAM))
/*
```

The DEFINE USERCATALOG command defines an SMS-managed integrated catalog facility user catalog, USERCAT1. Its parameters are:

- NAME specifies the user catalog, USERCAT1.
- ICFCATALOG specifies that the user catalog is to be in the integrated catalog facility catalog format.
- STRNO specifies that up to 3 concurrent requests to this catalog are to be processed. Like BUFSP, STRNO is not one of the data class attributes. If STRNO or BUFSP is not specified, the system will take the default established by access method services.
- DATACLAS specifies an installation-defined name of an SMS data class, VSDEF. The data set will assume the space parameters, and the FREESPACE, SHAREOPTIONS, and RECORDSIZE parameters contained in this data class. If your storage administrator has established ACS routines that will select a default data class, this parameter is optional. If a default data class is not assigned to this data set, however, you must explicitly specify any required parameters, in this case the space parameter, or the job will be unsuccessful.
- STORCLAS specifies an installation-defined name of an SMS storage class, SMSSTOR. This parameter is optional. If it is not specified, the data set will assume the storage class default assigned by the ACS routines.

- MGMTCLAS specifies an installation-defined name of an SMS management class, VSAM. This parameter is optional. If it is not specified, the data set might assume the management class default assigned by the ACS routines.

## Define an Integrated Catalog Facility User Catalog, Taking All Defaults: Example 2

In this example, an integrated catalog facility user catalog is defined and all defaults are taken.

```
//DEFUCAT  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
     DEFINE USERCATALOG -
            (NAME(USERCAT1) -
            ICFCATALOG )
/*
```

The DEFINE USERCATALOG command defines an SMS-managed integrated catalog facility user catalog, USERCAT1. Its parameters are:

- NAME specifies the user catalog, USERCAT1.
- ICFCATALOG specifies that the user catalog is to be in the integrated catalog facility catalog format.
- All the parameters are allowed to default. The ACS routines established by the storage administrator will assign a storage class to the catalog and can assign a management class. Because the access method services space parameter is not specified, the command is unsuccessful if a default data class is not assigned to this data set.

## Define an Integrated Catalog Facility User Catalog, Using SMS Keywords and the VOLUME Parameter: Example 3

In this example, an SMS-managed integrated catalog facility user catalog is defined and a specific volume is referenced.

```
//DEFUCAT  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
     DEFINE USERCATALOG -
            (NAME(USERCAT1) -
            VOLUME(VSER01) -
            ICFCATALOG -
            STRNO(3) -
            DATACLAS(VSDEF) -
            STORCLAS(SPECIAL) -
            MGMTCLAS(VSAM))
/*
```

The DEFINE USERCATALOG command defines an SMS-managed integrated catalog facility user catalog, USERCAT1. Its parameters are:

- NAME specifies the user catalog, USERCAT1.
- VOLUME specifies that the user catalog is to reside on volume VSER01. In this example, the installation defined SMS storage class of SPECIAL has the GUARANTEED SPACE=YES attribute. This allows specific volume allocation on this DEFINE using the VOLUME keyword.
- ICFCATALOG specifies that the user catalog to be defined is to be in the integrated catalog facility catalog format.

- STRNO specifies that up to 3 concurrent requests to this catalog are to be processed.
- DATACLAS specifies an installation defined name of an SMS data class. The data set will assume the space parameters, and the FREESPACE, SHAREOPTIONS, and RECORDSIZE parameters contained in this data class. If your storage administrator has established an ACS routine that will select a default data class, this parameter is optional. However, if a default data class is not assigned to this data set, you must explicitly provide the required parameters or the job will be unsuccessful.
- STORCLAS specifies an installation defined name of an SMS storage class. In this example, STORCLAS is not optional and you should not allow the catalog to assume the storage class default assigned by the ACS routines. The storage class named SPECIAL has the GUARANTEED SPACE=YES attribute and must be explicitly specified to enable specific volume allocation.
- MGMTCLAS specifies an installation defined name of an SMS management class. This parameter is optional. If it is not specified, the data set will assume the management class default assigned by the ACS routines.

## Define an Integrated Catalog Facility User Catalog, Using SMS Keywords and the VOLUME Parameter: Example 4

In this example, an SMS-managed integrated catalog facility user catalog is defined and a specific volume is referenced.

```
//DEFUCAT  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//VOL1     DD   VOL=SER=VSER01,UNIT=DISK,DISP=OLD
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
     DEFINE USERCATALOG -
           (NAME(USERCAT1) -
           VOLUME(VSER01) -
           ICFCATALOG -
           STRNO(3) -
           DATACLAS(VSDEF) -
           STORCLAS(SPECIAL) -
           MGMTCLAS(VSAM))
/*
```

Job control language statement:

- VOL1 DD describes the volume on which the catalog is to be defined.

The DEFINE USERCATALOG command defines an SMS-managed integrated catalog facility user catalog, USERCAT1. Its parameters are:

- NAME specifies the user catalog, USERCAT1.
- VOLUME specifies that the user catalog is to reside on volume VSER01. In this example, the installation defined SMS storage class of SPECIAL has the GUARANTEED SPACE=YES attribute. This allows specific volume allocation on this DEFINE using the VOLUME keyword.
- ICFCATALOG specifies that the user catalog is to be in the integrated catalog facility catalog format.
- STRNO specifies that up to 3 concurrent requests to this catalog are to be processed.
- DATACLAS specifies an installation defined name of an SMS data class. The data set will assume the space parameters, and the FREESPACE, SHAREOPTIONS, and RECORDSIZE parameters assigned to this data class by

the ACS routines. This parameter is optional. If it is not specified, the data set will assume the data class default assigned by the ACS routines.

- STORCLAS specifies an installation defined name of an SMS storage class. In this example, STORCLAS is not optional and you should not allow the catalog to assume the storage class default assigned by the ACS routines. The storage class named SPECIAL has the GUARANTEED SPACE=YES attribute and must be explicitly specified to enable specific volume allocation.
- MGMTCLAS specifies an installation defined name of an SMS management class. This parameter is optional. If it is not specified, the data set will assume the management class default assigned by the ACS routines.

## Define an Integrated Catalog Facility User Catalog: Example 5

In this example, an integrated catalog facility user catalog is defined.

**Hint:** If a small maximum RECORDSIZE is specified, extensions records for large Generation Data Groups (GDGs) must be created. The update to a single GDG that is in multiple records will require multiple unrelated I/Os.

```
//DEFCAT1  JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//VOL1     DD     VOL=SER=VSER01,UNIT=DISK,DISP=OLD
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE USERCATALOG -
           (NAME(USERCAT4) -
                    CYLINDERS(3 2) -
           VOLUME(VSER01) -
           ICFCATALOG -
           STRNO(3) -
           FREESPACE(10 20) -
           SHAREOPTIONS(3 4) -
           RECORDSIZE(4086 4086)) -
         DATA -
           (BUFND(4) -
           CONTROLINTERVALSIZE(4096)) -
         INDEX -
           (BUFNI(4) -
           CONTROLINTERVALSIZE(2048)) -
           CATALOG(ICFMAST1)
/*
```

Job control language statement:

- VOL1 DD describes the volume on which the catalog is to be defined.

The DEFINE USERCATALOG command defines an integrated catalog facility user catalog, USERCAT4. Its parameters are:

- NAME names the user catalog, USERCAT4.
- CYLINDERS specifies that 3 cylinders are to be allocated for the catalog. When the catalog is extended, it is in increments of 2 cylinders.
- VOLUME specifies that the user catalog is to reside on volume VSER01.
- ICFCATALOG specifies that the user catalog is to be in the integrated catalog facility catalog format.
- STRNO specifies that up to 3 concurrent requests to this catalog are to be processed.
- FREESPACE specifies the amount of free space to be left in the data component's control intervals (10%) and the control areas (20% of the control intervals in the control area) when data records are loaded into the user catalog.

- SHAREOPTIONS specifies the extent of cross-region sharing 3 (fully shared by any number of users) and cross-system sharing 4 (fully shared) to be allowed for the user catalog.
- RECORDSIZE specifies that the user catalog's records are variable length, with an average size of 4086 bytes and a maximum size of 4086 bytes.
- DATA and INDEX specify that parameters, BUFND and CONTROLINTERVALSIZE, and BUFNI and CONTROLINTERVALSIZE, are to be specified for the data and index components, respectively.
- BUFND specifies that 4 data buffers, of the data component's control interval size, are to be used when processing this user catalog.
- CONTROLINTERVALSIZE specifies the data and index component's control interval size, 4096 for the data component, and 2048 for the index component.
- BUFNI specifies that 4 index buffers, of the index component's control interval size, are to be used when processing this user catalog.
- CATALOG specifies that the catalog is to be defined in the master catalog, ICFMAST1.

## Define a User Catalog Using the MODEL Parameter: Example 6

In this example, the user catalog, USERCAT4, is used as a model for the user catalog being defined, RSTUCAT2.

```
//DEFCAT4  JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE USERCATALOG( -
           NAME(RSTUCAT2) -
           VOLUME(VSER03) -
           MODEL(USERCAT4 -
             USERCAT4)  ) -
         CATALOG(AMAST1)
/*
```

The DEFINE USERCATALOG command defines catalog RSTUCAT2. Its parameters are:

- NAME names the catalog, RSTUCAT2.
- VOLUME specifies that the catalog is to reside on volume VSER03. Volume VSER03 is dynamically allocated.
- MODEL identifies USERCAT4 as the catalog to use as a model for RSTUCAT2. The attributes and specifications of USERCAT4 that are not otherwise specified with the above parameters are used to define the attributes and specifications of RSTUCAT2. The master catalog, AMAST1, contains a user-catalog connector entry that points to USERCAT4. This is why USERCAT4 is specified as MODEL's catname subparameter. Values and attributes that apply to RSTUCAT2 as a result of using USERCAT4 as a model are:
  - FOR = 365 days (retention period)
  - CYLINDERS = 3 (primary) and 2 (secondary) are allocated to the catalog
  - BUFFERSPACE = 3072 bytes
  - ATTEMPTS = 2
  - NOWRITECHECK
  - CODE is null
  - AUTHORIZATION is null
  - OWNER is null
- CATALOG specifies that the user catalog connector is to be defined in the AMAST1 catalog.

## Define a General Tape Volume Catalog: Example 7

This example defines a general tape volume catalog named TEST1.VOLCAT.VGENERAL. A general tape volume catalog is required for a tape library.

```
//DEFVCAT     JOB     ...
//STEP1       EXEC   PGM=IDCAMS
//SYSPRINT    DD     SYSOUT=A
//SYSIN       DD     *
     DEFINE USERCATALOG -
             (NAME(TEST1.VOLCAT.VGENERAL) -
             VOLCATALOG -
             VOLUME(338001) -
             CYLINDERS(1 1))
/*
```

This example's parameters are:

- NAME specifies the name of the tape volume catalog as TEST1.VOLCAT.VGENERAL. This name determines the catalog to be a general tape volume catalog.
- VOLCATALOG specifies that the catalog is to contain only tape library and tape volume entries.
- VOLUME specifies that the catalog is to reside on volume 338001.
- CYLINDERS specifies that one cylinder is to be allocated to the catalog. When the catalog is extended, it is in increments of one cylinder.
- All other parameters are allowed to default.

## Define a Specific Tape Volume Catalog: Example 8

This example defines a specific tape volume catalog named TEST1.VOLCAT.VT.

```
//DEFVCAT     JOB     ...
//STEP1       EXEC   PGM=IDCAMS
//SYSPRINT    DD     SYSOUT=A
//SYSIN       DD     *
     DEFINE USERCATALOG -
              (NAME(TEST1.VOLCAT.VT) -
              VOLCATALOG -
              VOLUME(338001) -
              CYLINDERS(1 1))
/*
```

This example's parameters are:

- NAME specifies the name of the tape volume catalog to be TEST1.VOLCAT.VT. This name determines this catalog to be a specific tape volume catalog. 'VT' specifies that this catalog will contain all the tape volume entries whose volume serial numbers begin with the character 'T'.
- VOLCATALOG specifies that the catalog is to contain only tape library and tape volume entries.
- VOLUME specifies that the catalog is to reside on volume 338001.
- CYLINDERS specifies that one cylinder is to be allocated to the catalog. When the catalog is extended, it is in increments of one cylinder.
- All other parameters are allowed to default.

# Chapter 20. DELETE

The DELETE command deletes catalogs, VSAM data sets, non-VSAM data sets, and objects. The syntax of the DELETE command is:

| | |
|---|---|
| **DELETE** | **(***entryname***[** *entryname* **...])**<br>**[ALIAS\|**<br>**ALTERNATEINDEX\|**<br>**CLUSTER\|**<br>**GENERATIONDATAGROUP\|**<br>**LIBRARYENTRY\|**<br>**NONVSAM\|**<br>**NVR\|**<br>**PAGESPACE\|**<br>**PATH\|**<br>**TRUENAME\|**<br>**USERCATALOG\|**<br>**VOLUMEENTRY\|**<br>**VVR]**<br>**[ERASE\|NOERASE]**<br>**[FILE(***ddname***)]**<br>**[FORCE\|NOFORCE]**<br>**[PURGE\|NOPURGE]**<br>**[RECOVERY\|NORECOVERY]**<br>**[SCRATCH\|NOSCRATCH]**<br>**[CATALOG(***catname***)]** |

For VSAM RLS recoverable data sets, DELETE CLUSTER removes all pending recovery information for the sphere.

**Exception:** You cannot delete a VSAM sphere that has record-level sharing (RLS) recovery pending from a DFSMS/MVS 1.2 or lower system. A DELETE request for a data set with indirect or extended indirect Volser, will uncatalog the data set. It will not scratch (delete) it from the volume.

DELETE can be abbreviated: DEL

## DELETE Parameters

The DELETE command uses the following parameters.

## Required Parameters

**(***entryname***[** *entryname***...])**

names the entries to be deleted. If more than one entry is to be deleted, the list of entrynames must be enclosed in parentheses. The maximum number of entrynames that can be deleted is 100.

This parameter must be the first parameter following DELETE.

*entryname*

is the name of the entry to be deleted. A generic name can be coded to

© Copyright IBM Corp. 1973, 2002

delete multiple entries with one entryname. (For example, GENERIC.*.BAKER is a generic name where * is any 1-to-8 character simple name.)

Here is an example of how generic level DELETE works given the following data sets:

```
1) AAA.BBB.CCC.DDD
2) AAA.BBB.CCC.DDD
3) AAA.BBB.CCC.DDD.EEE
4) AAA.BBB.CCC

DELETE AAA.*  results in the deletion of no data sets.
DELETE AAA.BBB.* results in the deletion of data set #4
DELETE AAA.BBB.*.DDD results in the selection of data sets #1 and #2
DELETE AAA.BBB.*.DDD.EEE results in the deletion of data set #3.
```

When a generic level name is specified, only one qualifier can replace the asterisk (*)

If you are deleting a member of a non-VSAM partitioned data set, the entryname must be specified in the format: pdsname(membername). If you are deleting a non-VSAM data set that was defined by coding DEVICETYPES(0000) and VOLUMES(******), then DELETE only uncatalogs the data set. It does not scratch the data set from the SYSRES volume.

**Exception:** If data set contains indirect or symbolic VOLSER, the scratch parameter will be ignored, if specified.

## Optional Parameters

**ALIAS|ALTERNATEINDEX|CLUSTER|GENERATIONDATAGROUP|**

**LIBRARYENTRY|NONVSAM|NVR|PAGESPACE|PATH|**

**TRUENAME|USERCATALOG|VOLUMEENTRY|VVR**

specifies the type of object or entry to be deleted. If the object to be deleted is a catalog, truename entry, or VSAM volume record; USERCATALOG, TRUENAME, NVR, or VVR is required.

If you delete a migrated data set without specifying the entry type, DFSMShsm will delete the data set without recalling it.

**ALIAS**

specifies that the entry to be deleted is an alias entry.

**ALTERNATEINDEX**

specifies that the object to be deleted is an alternate index and its data and index entries. When a path entry is associated with the alternate index, the path entry is also deleted.

When the alternate index has the to-be-upgraded attribute and it is the only such alternate index associated with the base cluster, the base cluster's upgrade-set entry is also deleted.

**Exception:** If RLS recovery is associated with the alternate index, all knowledge of the recovery is lost as part of the delete operation.

**Abbreviation:** AIX

**CLUSTER**

specifies that the object to be deleted is a cluster, its associated data and index entries, and any related paths and alternate indexes.

When deleting a VVDS, entryname must be the restricted name 'SYS1.VVDS.Vvolser'.

**Exception:** If RLS recovery is associated with the sphere, all knowledge of the recovery is lost as part of the delete operation.

**Abbreviation:** CL

**GENERATIONDATAGROUP**

specifies that the entry to be deleted is a generation data group (GDG) entry. To delete a generation data group that is not empty, you must specify either the FORCE or the RECOVERY parameter. When FORCE is used, all SMS managed generation data sets pointed to by the GDG base are scratched. Generation data sets are also removed from the catalog when you use FORCE.

For both SMS-managed and non-SMS-managed GDGs, if you use RECOVERY, the GDG entry is deleted from the catalog and generation data sets remain unaffected in the VTOC. To delete a GDG using RECOVERY or FORCE, you must specify both GENERATIONDATAGROUP and RECOVERY or GENERATIONDATAGROUP and FORCE.

The FORCE and RECOVERY generation data set parameters require RACF facility class authorization. For information concerning RACF authorization levels, see “Appendix A. Security Authorization Levels” on page 321.

**Abbreviation:** GDG

**LIBRARYENTRY**

specifies that the entry to be deleted is a tape library entry. You must specify the FORCE parameter to delete a tape library entry that is not empty. A tape library entry is not empty when tape volume entries are still associated with it.

To delete a tape library entry, you must have authorization to RACF facility class profile STGADMIN.IGG.LIBRARY.

Because access method services cannot change the library manager inventory in an automated tape library, ISMF should be used for normal tape library delete functions. The access method services DELETE LIBRARYENTRY command should be used only to recover from volume catalog errors.

**Abbreviation:** LIBENTRY|LIBENT

**NONVSAM**

specifies that the entry to be deleted is a cataloged non-VSAM data set entry or object entry. If the non-VSAM data set is uncataloged, its format-1 DSCB can be deleted from the VTOC by using the SCRATCH function of the OS/VS IEHPROGM utility.

If the non-VSAM data set has aliases, all of its alias entries are deleted.

If the non-VSAM data set is partitioned, you can delete one of its members by specifying pdsname(membername).

Generation data sets in deferred state can be deleted in the same fashion as any other GDS. A GDS in deferred roll-in state can be reclaimed by rerunning the job step. Rolled-off, unexpired generations that have been recataloged can be deleted as NONVSAM data sets.

Under SMS, there is no support for temporary non-VSAM data sets.

**Exception:** Use RACF commands to specify an ERASE attribute in a generic or discrete profile for a non-VSAM data set. Use of the attribute renders all allocated DASD tracks unreadable before space on the volume is made available for reallocation. Refer to the appropriate RACF publications for information about how to specify and use this facility.

**Abbreviation:** NVSAM

**NVR**
specifies that the object to be deleted is an SMS-managed non-VSAM volume record (NVR) entry. This parameter must be specified to delete an NVR from a VSAM volume data set (VVDS) and its corresponding record from the VTOC. The NVR/VTOC entries are deleted only if the related non-VSAM object catalog entry does not exist.

Similar to DELETE VVR, the FILE parameter must specify the DD statement name that identifies the volume containing the VVDS. If you select a catalog through alias orientation or by use of the catalog parameter, it must match the catalog name in the isolated NVR (unless you have read authority to the RACF facility class STGADMIN.IGG.DLVVRNVR.NOCAT).

**PAGESPACE**
specifies that an inactive page space is to be deleted. A page space is identified as “active” during the operator’s IPL procedure.

To delete a page space in an SMS-managed user catalog you must include the CATALOG parameter. To delete a page space in a non-SMS-managed user catalog you must include both the CATALOG parameter and a JOBCAT/STEPCAT statement.

**Restriction:** If you want to delete an inactive pagespace, no other pagespace can be active on that volume. If a pagespace is active, the delete will be unsuccessful and you will receive an error message.

**Abbreviation:** PGSPC

**PATH**
specifies that a path entry is to be deleted. No entries associated with the path are deleted.

**TRUENAME**
specifies that the object to be deleted is the truename entry for a data or index component of a cluster or alternate index, or the name of an alternate index. This parameter must be specified to delete a truename entry. The truename entry is deleted only if the associated base record is missing or is inaccessible.

**Abbreviation:** TNAME

**USERCATALOG**
specifies that the object to be deleted is a user catalog.

The catalog connector entry in the master catalog is deleted. If the user catalog has aliases, all the catalog's alias entries in the master catalog are deleted.

To delete a user catalog when it is empty (that is, it contains only its self-describing entries and its volume's VVDS entry), you must specify USERCATALOG. To delete a user catalog that is not empty, you must specify both USERCATALOG and FORCE.

If you are deleting the catalog as part of recovering from a backup copy, you may want to use the RECOVERY option instead. For more information, see the RECOVERY keyword.

**Exception:** The JOBCAT/STEPCAT is ignored for this command. Use EXPORT DISCONNECT to delete a user catalog entry from a user catalog. A master catalog for another system is considered a user catalog by the processing system.

**Abbreviation:** UCAT

**VOLUMEENTRY**
specifies that the entry to be deleted is a tape library volume.

To delete a tape volume entry, you must have authorization to RACF facility class profile STGADMIN.IGG.LIBRARY. Because access method services cannot change the library manager inventory in an automated tape library, ISMF should be used for normal tape library delete functions. The access method services DELETE VOLUMEENTRY command should be used only to recover from volume catalog errors.

**Abbreviation:** VOLENTRY|VOLENT

**VVR**
specifies that the objects to be deleted are one or more unrelated VSAM volume record (VVR) entries. To delete a VVR from both the VSAM volume data set (VVDS) and from the VTOC, you must specify this parameter.

The VVR entry is deleted only if the related cluster or alternate-index data and index component catalog entries do not exist. When VVR is specified, the component name of the cluster or alternate-index to which the VVR was once related must be specified in the entryname parameter. If you select a catalog through alias orientation or by use of the catalog parameter, it must match the catalog name in the isolated VVR (unless you have read authority to the RACF facility class STGADMIN.IGG.DLVVRNVR.NOCAT).

The FILE parameter must specify the DD statement name that identifies the volume on which the VVDS resides.

**CATALOG(*catname*)**
specifies the name of the catalog that contains the entries to be deleted. See "Catalog Search Order for DELETE" on page 10 for the order in which catalogs are searched.

This parameter cannot be used to delete a user catalog, and is ignored when you delete members of a partitioned data set.

To specify catalog names for SMS-managed data sets, you must have authority from the RACF STGADMIN.IGG.DIRCAT facility class. See “Storage Management Subsystem (SMS) Considerations” on page 2 for more information.

*catname*
identifies the catalog that contains the entry to be deleted.

**Abbreviation:** CAT

**ERASE|NOERASE**
specifies whether the components of a cluster or alternate index to be deleted are to be erased, that is, overwritten with binary zeros.

This parameter will override whatever was coded when the cluster or alternate index was defined or last altered. This parameter should be specified only when a cluster or an alternate index entry is to be deleted. If you use ERASE, you must identify the to-be-deleted entry’s catalog with a JOBCAT or STEPCAT DD statement unless:
- The entry is in the master catalog, or
- The qualifiers in the entry’s qualified name are the catalog’s name or alias.

**ERASE**
specifies that the components are to be overwritten with binary zeros when the cluster or alternate index is deleted. If ERASE is specified, the volume that contains the data component must be mounted.

If the cluster is protected by a RACF generic or discrete profile and the cluster is cataloged in an integrated catalog facility catalog, use RACF commands to specify an ERASE attribute as part of this profile so that the data component is automatically erased upon deletion.

When you erase a data set, serialization is maintained using a RESERVE. For performance or access reasons, you might not want to use ERASE; another option is to convert the RESERVE to a systems enqueue. For more information about RESERVE contention, see *z/OS DFSMS: Managing Catalogs*.

**Abbreviation:** ERAS

**NOERASE**
specifies that the components are not to be overwritten with binary zeros when the cluster or alternate index is deleted.

NOERASE will not prevent the component from being erased if the cluster is protected by a RACF generic or discrete profile that specifies the ERASE attribute and the cluster is cataloged in an integrated catalog facility catalog. You can use RACF commands to alter the ERASE attribute in a profile.

**Abbreviation:** NERAS

**FILE(*ddname*)**
specifies the name of the DD statement that identifies:
- The volume that contains a data set to be deleted with SCRATCH.
- The data set to be deleted if ERASE is specified.
- The partitioned data set from which a member (or members) is to be deleted.
- The volumes that contain VVDS entries for the objects cataloged.
- The VVDS volume that contains a VVR or NVR to be deleted.

Use of the FILE parameter improves the performance of the DELETE command.

When you delete a data set, the volume referred to in the DD statement must be the same as the volume referred to in the usercatalog.

If you do not specify FILE and VSAM requires access to a volume or volumes during the delete processing, VSAM tries to dynamically allocate the volumes. When ERASE is specified and FILE is not specified, VSAM tries to dynamically allocate the entry name. Dynamic allocation requires that the volumes be mounted as permanently resident or reserved.

When more than one volume is to be identified (for example, a multivolume data set), FILE identifies the DD statement that specifies all volumes. If in any of the above cases the volumes are of a different device type, use concatenated DD statements. All volumes that contain associations to a cluster being deleted must also be included on the DD statement referenced by the FILE parameter.

When deleting multivolume non-VSAM data sets with the SCRATCH option, DELETE SCRATCH processing requires access to each volume in the entry's catalog record before the scratch can be issued. This requires either all volumes to be mounted, online, and allocatable to the job, or the use of the FILE parameter specifying a DD statement allocating at least one mountable unit (not permanently resident or reserved). Deferred mount must be specified on the DD statement so that allocation will flag the UCB to allow remove/mount requests to be issued for the unit as required during delete processing. If access to the volumes cannot be provided, use DELETE NOSCRATCH to uncatalog the non-VSAM data set and the user will assume the responsibility of scratching the format-1 DSCBs from all the volumes. If RACF is installed, you must have access authority under RACF to specify DELETE NOSCRATCH.

When the FILE parameter points to a DD statement that has DISP=SHR the data set can be deleted when allocated to another user, but is not open.

The use of this DD name in subsequent commands in the same invocation of IDCAMS may not work properly. Specifically, DEFINE, BLDINDEX, REPRO, and IMPORT may fail if these commands refer to the same DD name for output from those commands. This is because those commands will use volume and device-related information that may no longer be applicable. BLDINDEX, REPRO, and IMPORT should use the OUTDATASET keyword instead of OUTFILE to avoid this problem.

**FORCE|NOFORCE**
specifies whether entries that are not empty should be deleted.

**FORCE**
lets you delete generation data groups, tape library entries, and user catalogs without first ensuring that these entries are empty.

**ATTENTION:**

**The FORCE parameter deletes all clusters in the catalog.**

If you delete a generation data group using FORCE:

- Proper access authority to the RACF resource for catalog functions is necessary for DELETE GDG FORCE. The DELETE GDG FORCE function should not be used to redefine the GDG limit value. ALTER LIMIT should be used instead.
- The GDG entry is deleted even though it might point to non-VSAM entries in the catalog.
- Each SMS-managed non-VSAM data set entry pointed to by the GDG base entry is deleted before the GDG base entry is deleted. The non-VSAM data set is scratched.
- Each non-SMS-managed non-VSAM data set entry pointed to by the GDG base entry is deleted before the GDG base entry is deleted. However, the non-VSAM data set's space and contents on the volume are undisturbed.

If you delete a tape library entry using FORCE:

- The tape library entry is deleted even if tape volume entries are still associated with the specified tape library.
- Any tape volume entries associated with a deleted tape library entry will remain in the catalog for these tape volume entries.

If you delete a user catalog using FORCE:

**Attention:**

**The FORCE parameter deletes all clusters in the catalog.**

- The user catalog is deleted even if it contains entries for objects that have not been deleted.
- All data sets cataloged in the user catalog as well as the catalog data set itself are deleted. All volumes on which these data sets reside must be included with the FILE parameter.
- All VSAM clusters are automatically deleted, but the contents of each cluster and alternate index are not erased. (If you specify FORCE, the ERASE parameter is ineffective.)
- SMS-managed non-VSAM data set entries in the user catalog are deleted and the data sets are scratched.
- Non-SMS-managed non-VSAM data set entries in the user catalog are deleted, but the data sets are not scratched. A non-SMS-managed non-VSAM data set can be located with its DSCB in the volume's VTOC.

**Abbreviation:** FRC

**NOFORCE**
causes the DELETE command to end when you request the deletion of a generation data group, tape library entry, or user catalog that is not empty.

**Abbreviation:** NFRC

**PURGE|NOPURGE**
specifies whether the entry is to be deleted regardless of the retention period specified. If this parameter is used for objects that do not have a date associated with them (for example, VVRs, aliases, and non-SMS-managed non-VSAM data sets), the PURGE|NOPURGE parameter is ignored and the object is deleted. This parameter cannot be used if a truename entry is to be deleted.

PURGE must be specified to delete an OAM non-VSAM entry, because it has a never-expire retention.

**PURGE**
specifies that the entry is to be deleted even if the retention period, specified in the TO or FOR parameter, has not expired.

When deleting a tape library volume entry, PURGE must be specified if the volume's retention period has not expired.

PURGE affects the DFSMShsm delete function interaction for VSAM base clusters and non-VSAM data sets. It causes the migrated data set to be deleted regardless of the expiration date.

**Abbreviation:** PRG

**NOPURGE**
specifies that the entry is not to be deleted if the retention period has not expired.

**Abbreviation:** NPRG

**RECOVERY|NORECOVERY**
specifies whether a user catalog, a VSAM volume data set (VVDS), or a generation data group (GDG) is to be deleted in preparation for recovery.

**RECOVERY**
When RECOVERY is specified and the entry name identifies a user catalog, the user catalog is to be replaced with an imported backup copy. The user catalog, its VSAM volume record (VVR), and its VTOC entries are deleted. The VVR and DSCBs, for each of the objects defined in the user catalog, are not deleted or scratched. If the catalog is RACF-protected, alter authority is required.

VSAM must be able to read the VVDS or be able to process it as an ESDS for the function to complete successfully.

When RECOVERY is specified and entryname identifies a VVDS, the VVDS is unusable or inaccessible and must be rebuilt by deleting, redefining, and loading the appropriate VSAM data sets on the volume. The VVDS entry's DSCB will be scratched from the VTOC. The CATALOG parameter must contain the name of the master catalog when a VVDS is deleted with the RECOVERY parameter. If RACF protected, ALTER authority is required.

When RECOVERY is specified and the entry name identifies a GDG, the SMS-managed or non-SMS-managed GDG entry is deleted from the catalog and generation data sets remain unaffected in the VTOC.

If a VVDS contains a catalog entry or a system data set (SYS1.) entry that is cataloged in a master catalog, the VVDS catalog entry and the DSCB of the associated VVDS will not be removed.

If you delete a generation data group (DELETE GDG RECOVERY) using RECOVERY, proper authority to the RACF resource for catalog function is necessary.

RECOVERY cannot be specified with FORCE, NOFORCE, PURGE, NOPURGE, ERASE, NOERASE, SCRATCH, or NOSCRATCH.

**Abbreviation:** RCVRY

**NORECOVERY**
indicates that the entry is to be processed as described by the other parameters specified.

**Abbreviation:** NRCVRY

**SCRATCH|NOSCRATCH**

specifies whether a data set is to be removed from the VTOC of the volume on which it resides. This parameter can be specified only for a cluster, an alternate index, a page space, or a non-VSAM data set.

**Exceptions:**

- The SCRATCH parameter is not applicable to tape library and tape volume entries because they have no VVDS or VTOC entries. IDCAMS DELETE will determine if the data set to be deleted is a tape data set and issue the NOSCRATCH option on the delete request. For a data set on tape, using the NONVSAM parameter with a fully qualified entryname might cause dynamic allocation of the data set, and therefore a tape mount. To avoid the tape mount in this situation, either specify NOSCRATCH or omit NONVSAM.
- If data set contains indirect or symbolic VOLSER, the scratch parameter will be ignored, if specified.

**SCRATCH**

specifies that a data set is to be scratched from (removed from the VTOC of) the volume on which it resides. For VSAM data sets and SMS-managed non-VSAM data sets, the VSAM volume data set (VVDS) entry is also removed.

When SCRATCH is specified for a VVDS, the VVDS is scratched and the catalog entry for the VVDS is removed. The VVDS must be empty.

If the catalog entry does not exist for a non-VSAM data set, you can use the SCRATCH function of the OS/VS IEHPROGM utility to remove the format-1 DSCB from the VTOC.

If you select SCRATCH, you must use the JOBCAT or STEPCAT DD statement to identify the catalog unless the entry is in the master catalog, one or more of the qualifiers in the entry's qualified name is the same as the catalog's name or alias, or the FILE parameter is specified.

If you specify SCRATCH when deleting a non-VSAM data set defined with an esoteric device type, SYSDA for example, the DELETE will be unsuccessful under the following circumstances:

- Input/output configuration is changed resulting in addition or deletion of one or more esoteric device types.
- The esoteric device type definitions on the creating and using systems do not match when the catalog is shared between the two systems.

**Important:** If the VVDS indicates that the data set is owned by a catalog other than that catalog identified through the usual catalog search order for DELETE, a DELETE NOSCRATCH is done against the catalog that resulted from the catalog search, and a zero return code is returned to the user. See "Catalog Search Order for DELETE" on page 10 for information on the catalog search order for DELETE. For example, if you try a DELETE SCRATCH against a data set in Catalog A, and the VVDS indicates that the data set is owned by Catalog B, a DELETE NOSCRATCH operation is done against Catalog A, and the data set remains intact and accessible from Catalog B.

**Abbreviation:** SCR

**NOSCRATCH**

specifies that the catalog entry is to be deleted from the catalog without mounting the volume that contains the object defined by the entry. VVDS and VTOC entries are not deleted.

If RACF is installed, you must have access authority under RACF to specify NOSCRATCH. With proper authority, DELETE NOSCRATCH is allowed on SMS-managed VSAM and non-VSAM data sets, thus deleting the BCS entry in the catalog without accessing the VVDS or VTOC.

**Attention**

**DELETE NOSCRATCH can result in uncataloged SMS-managed data sets.**

NOSCRATCH removes the catalog entry for a VVDS. This entry can be reinstated with DEFINE RECATALOG. If the volume is mounted and usable, the VVDS is checked to ensure that the catalog entry being removed has no data sets in the VVDS. If the catalog entry indicates there are data sets in the VVDS, the VVDS's VSAM volume control record (VVCR) is removed and the catalog entry for the VVDS is removed.

If the volume is mounted and you specify NOSCRATCH for a VSAM volume data set (VVDS), the catalog entry for the VVDS is removed, and the catalog back pointer in the VSAM volume control record (VVCR) is removed.

You should specify NOSCRATCH for the following:

- If the format-1 DSCB of a non-VSAM data set has already been scratched from the VTOC.
- If you are deleting a non-VSAM data set that was defined with a device type named by the user (for example, SYSDA) and the device type is not valid.
- If the object is defined in an integrated catalog facility catalog and you want to recatalog the object in the same catalog.
- After you convert a volume, the names of catalogs owning data sets on the volume will still be in the VVCR. Only catalogs that reside on the converted volume need to have their names in the VVCR. You can remove unneeded catalog names from the VVCR by using DELETE VVDS NOSCRATCH with the CATALOG parameter referencing the catalog to be deleted from the VVCR. For compatibility, an error indication is still returned if there are VVR or NVRs on the volume for the referenced catalog.
- NOSCRATCH affects the DFSMShsm delete function interaction for VSAM base clusters and non-VSAM data sets. It causes the migrated data set to be recalled because a migrated data set cannot be uncataloged.

**Abbreviation:** NSCR

## DELETE Examples

The DELETE command can perform the following functions.

## Delete a Truename Entry in an Integrated Catalog Facility Catalog: Example 1

In this example, the truename entry for a data component of an alternate index is deleted. The purpose of this example is to remove the truename of an entry when an error has occurred, leaving the associated base record either inaccessible or missing. Removing the name allows a subsequent DEFINE command to reuse the name without an error caused by a duplicate name situation.

```
//DELET12    JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
     DELETE -
            K101.AIX.DATA
            TRUENAME -
            CATALOG(USERCAT4)
/*
```

The DELETE command deletes a truename entry that exists without its associated base record. The parameters are:

- K101.AIX.DATA is the entryname of the alternate index's data component to be deleted.
- TRUENAME specifies the type of entry to be deleted. When a truename entry is to be deleted, the TRUENAME parameter is required.
- CATALOG identifies the catalog that contains the entry to be deleted, USERCAT4.

## Delete an Integrated Catalog Facility User Catalog for Recovery: Example 2

In this example, a user catalog is deleted in preparation for replacing it with an imported backup copy. The VVDS and VTOC entries for objects defined in the catalog are not deleted and the data sets are not scratched.

```
//DELET13    JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//DD1       DD    VOL=SER=VSER01,UNIT=3380,DISP=OLD
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
     DELETE -
            USERCAT4 -
            FILE(DD1) -
            RECOVERY -
            USERCATALOG
/*
```

The DELETE command deletes a user catalog without deleting the VVDS and VTOC entries of the objects defined in the catalog. Its parameters are:

- USERCAT4 is the name of the catalog.
- FILE specifies the ddname of a DD statement that describes the user catalog's volume and causes it to be mounted.
- RECOVERY specifies that only the catalog data set is deleted without deleting the objects defined in the catalog.
- USERCATALOG specifies that the entryname identifies a user catalog. When a user catalog is to be deleted, the USERCATALOG parameter is required.

## Delete VSAM Volume Records: Example 3

In this example, VSAM volume records (VVRs) belonging to a data component of a key-sequenced cluster are deleted from the VVDS. The purpose of this example is to clean up the VVDS when there are residual records as a result of an error.

```
//DELET14    JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//DD1       DD    VOL=SER=VSER01,UNIT=3380,DISP=OLD
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
     DELETE -
            EXAMPLE.KSDS01.DATA -
            FILE(DD1) -
            VVR
/*
```

The DELETE command deletes the VVRs associated with a VSAM cluster from the VVDS and from the VTOC. Its parameters are:

- EXAMPLE.KSDS01.DATA is the name of the data component of the cluster.
- FILE specifies the ddname of a DD statement that describes the volumes that contain VVDS entries associated with this cluster.
- VVR specifies that only the VVRs for the cluster are to be deleted.

## Delete a Non-VSAM Data Sets Entry: Example 4

In this example, a non-VSAM data set's entry is deleted. The SCRATCH parameter is the default. A FILE parameter and its associated DD statement are provided to allocate the data set's volume. In this example, dynamic allocation is not used to provide catalog or volume allocation. This example applies only to non-VSAM data sets that have catalog entries.

```
//DELET4   JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//DD1      DD    VOL=SER=VSER02,UNIT=3380,DISP=OLD,
//   DSNAME=EXAMPLE.NONVSAM
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
           EXAMPLE.NONVSAM -
           FILE (DD1) -
           PURGE -
           CATALOG(USERCAT4)
/*
```

The DELETE command deletes the non-VSAM data set EXAMPLE.NONVSAM. Its parameters are:

- EXAMPLE.NONVSAM is the entryname of the object to be deleted.
- FILE specifies the ddname of a DD statement that describes the non-VSAM data set's volume and causes it to be mounted. When the data set is deleted, its DSCB entry in the volume's VTOC is removed.
- PURGE specifies that the non-VSAM data set's retention period or date is to be ignored.
- CATALOG identifies the catalog that contains the entries, USERCAT4.

## Delete Entries Associated with a Non-VSAM Object from VVDS and VTOC: Example 5

The following example shows how to delete entries associated with a non-VSAM object from the VVDS and VTOC. The purpose of this command is to clean up the VVDS and VTOC when there are residual records as a result of an error.

```
//DELET14  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//DD1      DD    VOL=SER=VSER01,UNIT=3380,DISP=OLD
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
           EXAMPLE.NONVSAM -
           FILE(DD1) -
           NVR
/*
```

The above DELETE command deletes the NVR associated with a non-VSAM object from the VVDS and its corresponding entry from the VTOC if they exist. Its parameters are:

- EXAMPLE.NONVSAM, the name of the non-VSAM object. There must not be a BCS entry for this object.
- FILE, specifies the ddname of a DD statement that describes the volume containing the VVDS entry associated with this object.
- NVR, specifies that only the NVR and its corresponding VTOC entry for this object are to be deleted.

## Delete a Key-Sequenced VSAM Cluster in a Catalog: Example 6

In this example, a key-sequenced cluster is deleted. Alternate indexes and paths related to the key-sequenced cluster are deleted automatically by access method services. Access method services will dynamically allocate the key-sequenced data set so that the data can be overwritten (as specified by the ERASE option).

```
//DELET1   JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//DD1      DD    VOL=SER=VSER02,UNIT=3380,DISP=OLD,
//   DSNAME=EXAMPLE.KSDS01
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
           EXAMPLE.KSDS1 -
           FILE(DD1)
           PURGE -
           ERASE -
           CATALOG(GGGUCAT2)
/*
```

The DELETE command deletes the key-sequenced VSAM cluster from the GGGUCAT2 catalog. Its parameters are:

- EXAMPLE.KSDS1, which is a key-sequenced VSAM cluster, is the entryname of the object being deleted.
- FILE is not required but improves performance if specified.
- PURGE specifies that the cluster is to be deleted regardless of its retention period or date.
- ERASE specifies that the cluster's data component be overwritten with binary zeros. If the NOERASE attribute was specified when the cluster was defined or altered, this is ignored.
- CATALOG identifies the catalog, GGGUCAT2, containing the cluster's entries.

## Delete Two Key-Sequenced Clusters in a Catalog: Example 7

In this example, two key-sequenced clusters, EXAMPLE.KSDS01 and EXAMPLE.KSDS02, are deleted from a catalog. It shows how more than one cataloged object is deleted with a single DELETE command.

```
//DELET3   JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
           (EXAMPLE.KSDS01 -
           EXAMPLE.KSDS02) -
           PURGE -
           CLUSTER
/*
```

The DELETE command deletes the key-sequenced clusters EXAMPLE.KSD01 and EXAMPLE.KSD02. Both entries are dynamically allocated in order for their respective catalog recovery areas (CRA) to be updated. The parameters are:

- EXAMPLE.KSDS01 and EXAMPLE.KSDS02 identify the objects to be deleted. These are the entrynames of two key-sequenced clusters.
- PURGE specifies that the cluster be deleted regardless of its retention period or date.
- CLUSTER specifies that EXAMPLE.KSDS01 and EXAMPLE.KSDS02 identify cluster catalog records.

## Delete a User Catalog: Example 8

In this example, a user catalog is deleted. A user catalog can be deleted when it is empty—that is, when there are no objects cataloged in it other than the catalog's volume. If the catalog is not empty, it cannot be deleted unless the FORCE parameter is specified.

**Attention:** The FORCE parameter deletes all clusters in the catalog.

```
//DELET6  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
           XXXUCAT1 -
           PURGE -
           USERCATALOG
/*
```

The DELETE command deletes both the catalog and the catalog's user catalog connector entry in the master catalog. The parameters are:

- XXXUCAT1 is the name of the user catalog.
- PURGE indicates the user catalog's retention period or date is to be ignored. If PURGE is not specified and the catalog's retention period has not yet expired, the catalog will not be deleted.
- USERCATALOG identifies XXXUCAT1 as a user catalog.

## Delete an Alias Entry in a Catalog: Example 9

In this example, an alias entry, EXAMPLE.NONVSAM1, is removed from catalog USERCAT4.

```
//DELET7    JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
```

```
//SYSIN     DD    *
     DELETE -
            EXAMPLE.NONVSAM1 -
            ALIAS -
            CATALOG(USERCAT4)
/*
```

The DELETE command removes an alias entry from catalog USERCAT4. Its parameters are:

- EXAMPLE.NONVSAM1 is the entryname of the object to be deleted. EXAMPLE.NONVSAM1 identifies an alias entry.
- ALIAS specifies the type of entry to be deleted. VSAM verifies that EXAMPLE.NONVSAM1 is an alias entry and then deletes it. If EXAMPLE.NONVSAM1 identifies another entry by mistake, VSAM does not delete the entry, but notes the discrepancy with a message to the programmer.
- CATALOG identifies the catalog containing the entry, USERCAT4.

## Delete Generically Named Entries in a Catalog: Example 10

In this example, each catalog entry with the name GENERIC.*.BAKER is deleted, where * is any 1-to-8 character simple name. The name GENERIC.*.BAKER is a generic name, and all catalog entries with the same generic name are deleted. Use this example to delete multiple entries. Multiple entries are entries with three levels of qualification where the first is GENERIC and the third is BAKER.

```
//DELET8   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     DELETE -
           GENERIC.*.BAKER -
           PURGE -
           CATALOG(USERCAT4)
/*
```

The DELETE command removes all entries (and their associated entries) with the generic name GENERIC.*.BAKER from catalog USERCAT4. Its parameters are:

- GENERIC.*.BAKER, a generic name, identifies all catalog entries with the high-level qualifier GENERIC and the low-level qualifier BAKER.
- PURGE specifies that each entry is to be purged regardless of the retention period or date specified when it was defined.
- CATALOG identifies the catalog, USERCAT4.

## List a Generation Data Group’s Entries, Then Delete the Group and Its Data Sets in a Catalog: Example 11

In this example, a generation data group, GDG01, and its associated (generation data set) entries are listed, the only generation data set in the group is deleted, and the generation data group base catalog entry is deleted.

```
//DELET9    JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
     LISTCAT -
            ENTRIES(GDG01) -
            ALL
     DELETE -
            GDG01.G0001V00 -
            PURGE
       DELETE -
```

```
            GDG01 -
            GENERATIONDATAGROUP -
            PURGE
/*
```

The LISTCAT command lists the generation data group, GDG01, and its associated generation data set entries. The parameters are:

- ENTRIES specifies that the entry GDG01 be listed. Because GDG01 is a generation data group entry, its associated generation data set's (non-VSAM) entries are also listed. If one of the generation data sets has aliases, the alias entries associated with the generation data set's entry are listed.
- ALL specifies that all fields are to be listed.

The first DELETE command removes the non-VSAM data set entry for the only generation data set, GDG01.G0001V00, in the generation data group. Its parameters are:

- GDG01.G0001V00 is the entryname of the object being deleted. GDG01.G0001V00 identifies the only generation data set in generation data group GDG01.
- PURGE specifies that the generation data set's retention period or date be ignored.

The second DELETE command removes the generation data group base catalog entry from the catalog. Its parameters are:

- GDG01 is the entryname of the object being deleted. GDG01 identifies the generation data group base entry.
- GENERATIONDATAGROUP specifies the type of entry being deleted. VSAM verifies that GDG01 is a generation data group entry, then deletes it. If GDG01 incorrectly specifies another type of entry, VSAM does not delete the entry, but notes the discrepancy with a message to the programmer.
- PURGE specifies that the generation data group's retention period or date be ignored.

## Delete a Generation Data Group with Recovery: Example 12

In this example, a generation data group base catalog entry, GDG01, is deleted from the catalog. The generation data sets associated with GDG01 remain unaffected in the VTOC.

```
//DELETXX   JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
           DELETE -
           GDG01 -
           GENERATIONDATAGROUP -
           RECOVERY
/*
```

The DELETE command removes the GDG base catalog entry from the catalog. Its parameters are:

- GDG01 is the name of the GDG base entry.
- GENERATIONDATAGROUP specifies the type of entry being deleted. VSAM verifies that GDG01 is a GDG entry, then deletes it. If GDG01 is not a GDG entry, VSAM issues a message and does not delete it.
- RECOVERY specifies that only the GDG base entry name in the catalog is deleted. Its associated generation data sets remain intact in the VTOC.

## Delete a Member of a Partitioned (Non-VSAM) Data Set in a Catalog: Example 13

In this example, the MEM1 member of partitioned data set EXAMPLE.NONVSAM2 is deleted, then the data set itself is deleted.

```
//DELET10    JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
     DELETE -
            EXAMPLE.NONVSAM2(MEM1) -
     DELETE -
            EXAMPLE.NONVSAM2 -
            PURGE -
            CATALOG(USERCAT4)
/*
```

The first DELETE command deletes a member of a partitioned data set, EXAMPLE.NONVSAM2(MEM1), from the user catalog USERCAT4. Its parameters are:

- EXAMPLE.NONVSAM2(MEM1) is the entryname of a member of the partitioned data set, EXAMPLE.NONVSAM2. The entryname identifies the object to be deleted.

The second DELETE command deletes all remaining members and then the partitioned non-VSAM data set, EXAMPLE.NONVSAM2, itself. Its parameters are:

- EXAMPLE.NONVSAM2 is the entryname of the object being deleted.
- PURGE specifies that the non-VSAM data set's retention period or date be ignored. If PURGE is not specified and the non-VSAM data set's retention period has not expired, VSAM does not delete its entry.
- CATALOG identifies the catalog, USERCAT4.

In the second part of this example, the DSCB entry in the volume's VTOC is removed. Dynamic allocation is used to allocate the data set's volume.

## Delete a Page Space: Example 14

In this example, page space SYS1.PAGE2 is deleted from the master catalog. You must ensure other BCS's do not have that data set cataloged by performing a DELETE NOSCRATCH on each, and then performing a DELETE SCRATCH on the BCS that originally 'owned' the system data set.

```
//DELET11   JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
     DELETE -
            SYS1.PAGE2 -
            PURGE -
            PAGESPACE
/*
```

The DELETE command removes the page space entry, SYS1.PAGE2, from the master catalog. Its parameters are:

- SYS1.PAGE2 is the entryname of the object being deleted. SYS1.PAGE2 identifies a page space entry.
- PURGE specifies that the page space entry be deleted regardless of the retention period or date specified when it was defined.

- PAGESPACE specifies the type of entry being deleted. VSAM verifies that SYS1.PAGE2 is a page space entry, then deletes it. If SYS1.PAGE2 incorrectly identifies another type of entry, VSAM does not delete it, but sends an error message to the programmer.

## Delete a VVDS with Recovery: Example 15

In this example, the VVDS is deleted. The VTOC and catalog entries for the objects reflected by the VSAM volume records (in the VVDS) remain intact.

```
//DELET13    JOB    ...
//STEP1      EXEC   PGM=IDCAMS
//DD1        DD     VOL=SER=338001,UNIT=3380,DISP=OLD
//SYSPRINT   DD     SYSOUT=A
//SYSIN      DD     *
     DELETE  -
            SYS1.VVDS.V338001 -
            FILE(DD1) -
            RECOVERY
/*
```

- SYS1.VVDS.V338001 is the name of the VVDS.
- FILE specifies the name of a DD statement that both describes the VVDS volume and causes it to be mounted.
- RECOVERY specifies that the VVDS entry is being deleted from the VTOC as part of a recovery operation.

## Delete an OAM Collection Name Catalog Entry: Example 16

In this example, an OAM non-VSAM collection name entry is deleted from a catalog. A FILE parameter and its associated DD statement are provided to allocate the volume where the catalog containing the collection name entry is located.

```
//DELET15    JOB    ...
//STEP1      EXEC   PGM=IDCAMS
//DD1        DD     VOL=SER=VSER01, UNIT=3380, DISP=OLD
//SYSPRINT   DD     SYSOUT=A
//SYSIN      DD     *
     DELETE -
             OAM.COLLECTION.NONVSAM -
             FILE (DD1) -
             PURGE -
             NOSCRATCH -
             CATALOG(COLNCAT)
/*
```

The DELETE command deletes the non-VSAM collection name entry from the catalog. Its parameters are:

- OAM.COLLECTION.NONVSAM is the name of the collection name entry in the catalog.
- FILE specifies the DD statement within the JCL that locates the volume where the catalog containing the collection name entry marked for deletion resides.
- PURGE specifies that the retention period for the non-VSAM collection name entry be ignored.
- NOSCRATCH specifies that only the collection name entry on the catalog be deleted.
- CATALOG identifies the catalog where the collection name entry marked for deletion resides is COLNCAT. If you are trying to delete SMS-managed entries, you must have RACF ALTER authority.

There must be synchronization between the OAM collection name entries in the catalog and the OAM DB2 Administration Database. Deleting a collection name entry from the catalog will not delete the corresponding entry from the OAM DB2 Administration Database. The above DELETE command should be used in recovery situations to ensure synchronization between the catalog and the OAM DB2 Administration Database.

## Delete a Tape Library Entry: Example 17

This example deletes a tape library entry. Because the FORCE parameter was not used, the tape library entry ATLLIB1 is deleted only if no tape volume entries are associated with it.

```
//DELLIB     JOB     ...
//STEP1      EXEC   PGM=IDCAMS
//SYSPRINT   DD     SYSOUT=A
//SYSIN      DD     *
     DELETE (ATLLIB1) -
            LIBRARYENTRY
/*
```

This command's parameters are:

- ATLLIB1 is the name of the tape library entry that is to be deleted.
- LIBRARYENTRY specifies the type of entry to be deleted.

## Delete a Tape Volume Entry: Example 18

This example deletes a tape volume entry.

```
//DELVOL     JOB     ...
//STEP1      EXEC   PGM=IDCAMS
//SYSPRINT   DD     SYSOUT=A
//SYSIN      DD     *
   DELETE (VAL0001) -
            VOLUMEENTRY -
            PURGE
/*
```

This command's parameters are:

- VAL0001 is the name of the tape volume entry that is to be deleted. This tape volume entry's volser is AL0001.
- VOLUMEENTRY specifies that a tape volume entry is to be deleted.
- PURGE specifies that the volume entry is to be deleted regardless of the expiration date.

# Chapter 21. DIAGNOSE

The DIAGNOSE command scans a basic catalog structure (BCS) or a VSAM volume data set (VVDS) to validate the data structures and detect structure errors.

See “Chapter 22. EXAMINE” on page 235 for information on the EXAMINE command, which can inspect the structural integrity of the data or index component of a key-sequenced data set cluster or of a BCS.

The syntax of the DIAGNOSE command is:

| | |
|---|---|
| **DIAGNOSE** | **{ICFCATALOG\|VVDS}**<br>**{INFILE(***ddname***)\| INDATASET(***datasetname***)}**<br>**[COMPAREDD(***ddname* **[** *ddname***...])\|**<br>**COMPAREDS(***dsname* **[** *dsname***...])]**<br>**[DUMP\|NODUMP]**<br>**[ERRORLIMIT(***value***)]**<br>**[EXCLUDE**<br>**({ENTRIES(***entryname***[** *entryname***...])\|**<br>**CATALOG(***catalogname***[** *catalogname***...])\| LEVEL(***level***)})\|**<br>**INCLUDE**<br>**({ENTRIES(***entryname***[** *entryname***...])\|**<br>**CATALOG(***catalogname***[** *catalogname***...])\| LEVEL(***level***)})]**<br>**[LIST\|NOLIST]**<br>**[OUTFILE(***ddname***)]** |

DIAGNOSE can be abbreviated: DIAG

Because the DIAGNOSE command checks the content of the catalog records, and the records might, for example, contain damaged length field values, there is a possibility that the job will abend. For detailed information on using DIAGNOSE, see *z/OS DFSMS: Managing Catalogs*.

## DIAGNOSE Parameters

The DIAGNOSE command uses the following parameters.

### Required Parameters

**ICFCATALOG|VVDS**

Specifies which data set is to be scanned for diagnosis.

You must have access authority under the RACF facility class to diagnose a BCS or a VVDS.

**ICFCATALOG**

specifies that the data set to be scanned for diagnosis is the basic catalog structure (BCS).

**Abbreviation:** ICFCAT

**VVDS**

Specifies that the data set to be scanned for diagnosis is a VVDS for an integrated catalog facility BCS.

**INFILE(***ddname***)|**

© Copyright IBM Corp. 1973, 2002

**INDATASET(*datasetname*)**
names the DD statement or data set that specifies the data set to be scanned.

Because a VVDS must be referenced by its volume serial number and unit, use INFILE to specify a VVDS. A BCS can be specified by either INFILE or INDATASET.

If you are authorized to the RACF facility class name of STGADMIN.IDC.DIAGNOSE.CATALOG, you are allowed to open an integrated catalog facility catalog without performing usual catalog security processing. If you are authorized to this facility class name, normal RACF checking is bypassed. If you try to open an integrated catalog facility catalog and you are not authorized to this facility class name, message IDC2918I is issued, processing continues, and normal RACF checking takes place.

**INFILE(*ddname*)**
specifies the DD statement of the data set to be scanned.

**Abbreviation:** IFILE

**INDATASET(*datasetname*)**
specifies the data set name of the data set to be scanned.

**Abbreviation:** IDS

## Optional Parameters

The DIAGNOSE command can use the following optional parameters.

**COMPAREDD(*ddname* [ *ddname*...])|**

**COMPAREDS(*dsname* [ *dsname*...])]**
indicates which data sets are to be checked to confirm that they point to the BCS or VVDS being diagnosed. Because a VVDS must be referenced by its volume serial number and unit, use COMPAREDD to specify a VVDS. A BCS can be specified by either COMPAREDD or COMPAREDS. For diagnosis of a BCS, the compare parameters identify VVDS names (you can specify a maximum of 99 names). When diagnosing a VVDS, these parameters identify appropriate BCS data sets.

If COMPAREDS or COMPAREDD are specified for the catalog whose name is indicated by the VVDS entry, the catalog should have:

- A non-VSAM record corresponding to the NVR, or a cluster record corresponding with the data or index VVR.
- The same storage class, data class, and management class names in the corresponding non-VSAM or cluster record.

If you are authorized to the RACF facility class name of STGADMIN.IDC.DIAGNOSE.VVDS, you are allowed to open an integrated catalog facility catalog without performing normal catalog security processing. If you are authorized to this facility class name, normal RACF checking is bypassed. If you try to open an integrated catalog facility catalog and you are not authorized to this facility class name, message IDC2918I is issued, processing continues, and normal RACF checking takes place.

**COMPAREDD(*ddname*[ *ddname* ...])**
indicates the ddnames of the specific data sets to be checked.

**Abbreviation:** CMPRDD

**COMPAREDS(*dsname* [ *dsname*...])**
indicates the names of the data sets to be checked.

**Abbreviation:** CMPRDS

**DUMP|NODUMP**
specifies whether entry hexadecimal dumps are to be provided for compare errors.

**DUMP**
indicates that entry hexadecimal dumps are to be provided for compare errors. This results in message IDC21365I followed by a display of a record or records.

**NODUMP**
indicates that no dump is to be provided.

**ERRORLIMIT(*value*)**
specifies a modification of the default error limit. Designed to prevent runaway output, ERRORLIMIT defaults to 16, but any number from 0 to 2,147,483,647 can be specified. During DIAGNOSE, each incorrect entry contributes to the error count used against ERRORLIMIT. When ERRORLIMIT is reached, message IDC31370I is printed and analysis of the source data set is ended.

**Abbreviation:** ELIMIT

**EXCLUDE({ENTRIES(*entryname*[ *entryname*...])|**

**CATALOG(*catalogname*[ *catalogname*...])| LEVEL(*level*)})**
specifies that entries is excluded from the scan. INCLUDE and EXCLUDE are mutually exclusive parameters. If omitted, the entire data set is processed. See *z/OS DFSMS: Managing Catalogs* for more information on the effect of specifying INCLUDE and EXCLUDE with the DIAGNOSE commands.

**Abbreviation:** EXCL

**ENTRIES(*entryname*[ *entryname*...])**
specifies that the entries listed is excluded from the scan. Up to 255 entrynames can be coded.

**Abbreviation:** ENT

**CATALOG(*catalogname* [ *catalogname*...])**
specifies that entries that refer to the named catalog are not scanned. Up to 99 catalog names can be coded. CATALOG can only be coded for DIAGNOSE VVDS.

**Abbreviation:** CAT

**LEVEL(*level*)**
specifies the high-level qualifiers for entrynames. Only entries with the high-level qualifier specified is excluded from the scan. One level name can be coded.

**Abbreviation:** LVL

**INCLUDE({ENTRIES(*entryname*[ *entryname*...])|**

**CATALOG(*catalogname*[ *catalogname*...])| LEVEL(*level*)})**
specifies what information is included in the scan. INCLUDE and EXCLUDE are mutually exclusive parameters. If omitted, the entire data set is processed. See *z/OS DFSMS: Managing Catalogs* for more information on the effect of specifying INCLUDE and EXCLUDE with the DIAGNOSE commands.

**Abbreviation:** INCL

**ENTRIES(***entryname* **[** *entryname***...])**
specifies that only the entries listed are scanned. Up to 255 entrynames can be coded.

**Abbreviation:** ENT

**CATALOG(***catalogname* **[** *catalogname***...])**
specifies that only entries that refer to the named catalog are scanned. CATALOG can only be coded for DIAGNOSE VVDS.

**Abbreviation:** CAT

**LEVEL(***level***)**
specifies the high-level qualifiers for entrynames. Only entries with the specified high-level qualifier are scanned. One level name can be coded.

**Abbreviation:** LVL

**LIST|NOLIST**
specifies whether entries that have no errors are to be listed.

**LIST**
indicates the entries that have no errors are to be listed in addition to entries that have errors. This results in message IDC01360I, followed by a list of entrynames.

**NOLIST**
indicates that only entries with errors are listed.

**Abbreviation:** NLST

**OUTFILE(***ddname***)**
specifies a data set, other than the SYSPRINT data set, to receive the output produced by DIAGNOSE (that is, the output resulting from the scan operation).

ddname identifies a DD statement that describes the alternate target data set. If OUTFILE is not specified, the output is listed in the SYSPRINT data set. If an alternate data set is specified, it must meet the requirements shown in “JCL DD Statement for an Alternate Target Data Set” on page 4.

**Abbreviation:** OFILE

## DIAGNOSE Examples

The DIAGNOSE command can perform the functions shown in the following examples.

## Diagnose a VVDS: Compare the BCS and VVDS Example 1

In this example, the VVDS is diagnosed and the BCS and VVDS are compared. The BCS and the VVDS are passed as data set names.

```
//DIAGPWD  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//DIAGDD   DD   UNIT=SYSDA,VOL=SER=PERM03,DISP=SHR,
//         DSN=SYS1.VVDS.VPERM03,AMP='AMORG'
//SYSIN    DD   *
     DIAGNOSE -
           VVDS -
           INFILE(DIAGDD) -
           COMPAREDS(CAT002)
/*
```

Job control language statement:

- DIAGDD DD specifies the input data set, SYS1.VVDS.VPERM03.

The DIAGNOSE command diagnoses VVDS and compares the BCS, CAT002, with the VVDS. The parameters are:

- VVDS specifies that the input data set is a VVDS.
- INFILE identifies the DD statement, DIAGDD, containing the VVDS for diagnosis.
- COMPAREDS indicates that comparison checking is to occur and specifies the data set name of the BCS, CAT002.

## Diagnose Only the BCS: Example 2

In this example, only the BCS is diagnosed; the BCS and VVDS are not compared. The catalog is identified with a ddname. DIAGNOSE defaults to DUMP, NOLIST, and ERRORLIMIT(16).

```
//DIAGEX1  JOB
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//DIAGDD   DD    DISP=SHR,DSN=UCAT1
//SYSIN    DD    *
     DIAGNOSE -
           ICFCATALOG -
           INFILE(DIAGDD)
/*
```

Job control language statement:

- DIAGDD DD specifies the input data set. Because only the DSNAME is given, the BCS, UCAT1, must be cataloged in the master catalog.

The DIAGNOSE command scans a BCS, UCAT1. Its parameters are:

- ICFCATALOG indicates the input is a BCS and not a VVDS.
- INFILE(DIAGDD) identifies the DD statement containing the input data set name.

## Diagnose the BCS: Compare the BCS and Certain VVDSs: Example 3

In this example, the BCS is diagnosed and the BCS and certain VVDSs are compared. The BCS and the VVDSs are passed as ddnames. DIAGNOSE defaults to DUMP, NOLIST, and ERRORLIMIT(16).

```
//DIAGEX2  JOB
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//DIAGDD   DD    DISP=SHR,DSN=DIAGCAT3
//DIAG01   DD    UNIT=SYSDA,VOL=SER=PERM03,DISP=SHR,
//         DSN=SYS1.VVDS.VPERM03,AMP='AMORG'
//DIAG02   DD    UNIT=SYSDA,VOL=SER=DIAG02,DISP=SHR,
//         DSN=SYS1.VVDS.VDIAG02,AMP='AMORG'
//DIAG03   DD    UNIT=SYSDA,VOL=SER=DIAG03,DISP=SHR,
//         DSN=SYS1.VVDS.VDIAG03,AMP='AMORG'
//SYSIN    DD    *
     DIAGNOSE -
           ICFCATALOG -
           INFILE(DIAGDD) -
           COMPAREDD(DIAG01 DIAG02 DIAG03)
/*
```

Job control language statements:

- DIAGDD DD identifies the BCS being scanned. This BCS must be cataloged in the master catalog.
- DIAG01 DD, DIAG02 DD, and DIAG03 DD identify VVDSs to be compared.

The DIAGNOSE command diagnoses the BCS, DIAGCAT3, and compares the BCS to certain VVDSs. The parameters are:

- ICFCATALOG denotes that the input data set is an integrated catalog facility BCS.
- INFILE(DIAGDD) identifies the DD statement containing the input data set name.
- COMPAREDD(DIAG01 DIAG02 DIAG03) indicates that any BCS entries using the specified VVDSs are to undergo comparison checking. The VVDS names are passed on DD statements.

## Diagnose a VVDS: Compare the BCS and VVDS: Example 4

In this example, the VVDS is diagnosed and the BCS and VVDS are compared. The BCS and VVDS are passed as ddnames. DIAGNOSE defaults to DUMP, NOLIST, and ERRORLIMIT(16).

```
//DIAGEX3  JOB
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//DIAGDD   DD    UNIT=SYSDA,VOL=SER=PERM03,DISP=SHR,
//         DSN=SYS1.VVDS.VPERM03,AMP='AMORG'
//DIAG01   DD    DISP=SHR,DSN=CAT001
//SYSIN    DD    *
     DIAGNOSE -
           VVDS -
           INFILE(DIAGDD) -
           COMPAREDD(DIAG01)
/*
```

Job control language statements:

- DIAGDD DD contains the input data set name.
- DIAG01 DD contains the name of a BCS to be compared to the input data set.

The DIAGNOSE command scans a VVDS and compares the BCS (CAT001) with the VVDS. The parameters are:

- VVDS indicates the input data set is a VVDS.
- INFILE(DIAGDD) identifies the DD statement containing the name of the input data set.
- COMPAREDD(DIAG01) indicates that the VVDS be compared with a BCS. DIAG01 is the name of the DD statement containing the BCS name.

## Diagnose a VVDS: Compare the BCS and VVDS: Example 5

In this example, the VVDS is diagnosed and the BCS and VVDS are compared. The BCS is passed as a data set name; the VVDS is passed as a ddname. Only the entries cataloged in CAT001 are processed. The listing of valid entries is done with LIST. DIAGNOSE defaults to DUMP and ERRORLIMIT(16).

```
//DIAGEX4  JOB
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//DIAGDD   DD    UNIT=SYSDA,VOL=SER=PERM03,DISP=SHR,
//         DSN=SYS1.VVDS.VPERM03,AMP='AMORG'
//SYSIN    DD    *
     DIAGNOSE -
           VVDS -
           INFILE(DIAGDD) -
           COMPAREDS(CAT001) -
           INCLUDE (CATALOG (CAT001)) -
           LIST
/*
```

Job control language statement:

- DIAGDD DD contains the VVDS name being diagnosed.

Use the DIAGNOSE command to diagnose a VVDS and compare a VVDS with the BCS, CAT001. The parameters are:

- VVDS identifies the input data set as a VVDS.
- INFILE(DIAGDD) denotes the input data set name is contained in the DD statement named DIAGDD.
- COMPAREDS(CAT001) indicates that a VVDS and BCS compare be done. The BCS name is specified as CAT001.
- INCLUDE(CATALOG(CAT001)) specifies that only the VVDS entries cataloged for CAT001 be diagnosed.
- LIST specifies that entries both with and without errors be listed.

## Diagnose a VVDS: Compare the BCS and VVDS: Example 6

In this example, the VVDS is diagnosed and the BCS and VVDS are compared. The BCS and the VVDS are passed as data set names. The entries cataloged in CAT001 are not to be processed. The listing of valid entries is to be done, but error dumps are to be suppressed. The diagnosis is to be ended after one error is detected.

```
//DIAGEX5  JOB
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//DD1      DD    UNIT=SYSDA, VOL=SER=PERM03,DISP=SHR,
//         DSN=SYS1.VVDS.VPERM03,AMP='AMORG'
//SYSIN    DD    *
     DIAGNOSE -
           VVDS -
           INFILE(DD1) -
           EXCLUDE (CATALOG (CAT001)) -
           COMPAREDS(CAT002) -
           LIST -
           NODUMP -
           ERRORLIMIT(1)
/*
```

The VVDS is diagnosed and the BCS, CAT002, is compared with the VVDS. The parameters are:

- VVDS identifies the input data set as a VVDS.
- INFILE(DD1) designates the DD statement containing the VVDS being diagnosed.
- COMPAREDS(CAT002) indicates that comparison checking be done and specifies the data set name of the BCS.
- EXCLUDE(CATALOG(CAT001)) indicates that VVDS entries cataloged in CAT001 not be processed.
- LIST requests that entries both with and without errors be listed.
- NODUMP specifies that entries with errors are not to be hex-dumped.
- ERRORLIMIT(1) changes the number of errors to be processed to one.

## Diagnose a VVDS: Compare the BCS and VVDS: Example 7

In this example, the VVDS is diagnosed and the BCS is compared with the VVDS. The BCS and the VVDS are passed as ddnames. Only those entries with a high-level qualifier of CAT are processed. The default values of DUMP, NOLIST, and ERRORLIMIT(16) are taken.

```
//DIAGEX6  JOB
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//DIAGDD   DD    UNIT=SYSDA,VOL=SER=PERM03,DISP=SHR,
//         DSN=SYS1.VVDS.VPERM03,AMP='AMORG'
//DIAG01   DD    DISP=SHR,DSN=CAT001
//SYSIN    DD    *
     DIAGNOSE -
           VVDS -
           INFILE(DIAGDD) -
           COMPAREDD (DIAG01)-
           INCLUDE (LEVEL (CAT))
/*
```

Job control language statements:
- DIAGDD DD indicates the VVDS name.
- DIAG01 DD indicates the BCS name.

The DIAGNOSE command's parameters are:
- VVDS indicates the input data set is a VVDS.
- INFILE(DIAGDD) identifies the DD statement containing the VVDS name.
- COMPAREDD(DIAG01) indicates that the VVDS and BCS be compared and identifies the DD statement containing the BCS name.
- INCLUDE(LEVEL(CAT)) indicates that only certain VVDS entries be processed, specifically, entrynames with a high-level qualifier of CAT—for example:

  `CAT.CNTLR.NOV80 and CAT.BACKUP.SMFDATA.J34`

  Any entries without such a high-level qualifier are excluded from processing.

# Chapter 22. EXAMINE

The EXAMINE command analyzes and reports on the structural integrity of the index and data components of a key-sequenced data set cluster (KSDS) and of a variable-length relative record data set cluster (VRRDS). In addition, EXAMINE can analyze and report on the structural integrity of the basic catalog structure (BCS) of an integrated catalog.

See *z/OS DFSMS: Using Data Sets* for more information on KSDSs and VRRDSs. See *z/OS DFSMS: Managing Catalogs* for more information on BCSs.

See "Chapter 21. DIAGNOSE" on page 227 for information on the DIAGNOSE command, which inspects the contents of a VVDS or a BCS and looks for logical synchronization errors.

**Recommendation**: If you are using the EXAMINE command with a catalog, run the VERIFY command first.

The syntax of the EXAMINE command is:

| | |
|---|---|
| **EXAMINE** | **NAME(***clustername***)**<br>**[INDEXTEST\|NOINDEXTEST]**<br>**[DATATEST\|NODATATEST]**<br>**[ERRORLIMIT***(value)***]** |

## EXAMINE Parameters

The EXAMINE command uses the following parameters.

## Required Parameters

**NAME(***clustername***)**
specifies the cluster to be analyzed for structural integrity by EXAMINE. You specify the cluster component you want examined by setting the appropriate EXAMINE parameters.

*clustername*
identifies the cluster to be analyzed.

## Optional Parameters

**INDEXTEST|NOINDEXTEST**
specifies whether or not EXAMINE is to perform tests associated with the index component of the cluster. INDEXTEST is the default.

**INDEXTEST**
performs tests upon the index component of a key-sequenced data set cluster.

**Abbreviation:** ITEST

**NOINDEXTEST**
does not perform any testing upon the index component of a key-sequenced data set cluster.

**Abbreviation:** NOITEST

© Copyright IBM Corp. 1973, 2002

**DATATEST|NODATATEST**
specifies whether or not EXAMINE is to perform tests associated with the data component of the cluster. NODATATEST is the default.

**DATATEST**
performs tests upon the data component of a key-sequenced data set cluster. NOINDEXTEST and DATATEST are specified when *only* a DATATEST is desired.

**Abbreviation:** DTEST

**NODATATEST**
does not perform any testing upon the data component of a key-sequenced data set cluster.

**Abbreviation:** NODTEST

**ERRORLIMIT(value)**
specifies a numeric limit (value) to the number of errors for which detailed EXAMINE error messages are to be printed during program execution. ERRORLIMIT is designed to prevent runaway message output. The default value for ERRORLIMIT is 2,147,483,647 errors, but you can specify any number between 0 and 2,147,483,647. Note that processing continues even though the error limit is reached.

**Abbreviation:** ELIMIT

# EXAMINE Examples

## Examine the Index Component of an Integrated Catalog Facility User Catalog: Example 1

This example shows how to determine whether the index component of your catalog has structural errors.

```
//EXAMEX1   JOB
//STEP1     EXEC   PGM=IDCAMS
//SYSPRINT  DD     SYSOUT=A
//SYSIN     DD     *
     EXAMINE -
            NAME(ICFCAT.V338001) -
            ERRORLIMIT(0)
/*
```

The EXAMINE command is used, in this example, to analyze the index component of an integrated catalog facility catalog. Its parameters are:

- NAME, specifies the catalog name. The catalog must be connected to the master catalog.
- INDEXTEST, specified by default.
- ERRORLIMIT(0), suppresses the printing of detailed error messages.

## Examine Both Components of a Key-Sequenced Data Set: Example 2

This example shows how to get a list of data set structural errors that you can be use to support problem resolution.

```
//EXAMEX2   JOB
//STEP1     EXEC    PGM=IDCAMS
//SYSPRINT  DD      SYSOUT=A
//SYSIN     DD      *
     EXAMINE -
```

```
            NAME(KSDS01) -
            INDEXTEST -
            DATATEST
/*
```

Use the EXAMINE command to analyze both components of a key-sequenced data set. Its parameters are:

- NAME, specifies the cluster name only.
- INDEXTEST, causes the index component to be examined.
- DATATEST, causes the data component to be examined.
- The default for ERRORLIMIT (it was not specified) allows detailed error messages to be printed.

## Examine the Data Component of an Integrated Catalog Facility User Catalog: Example 3

This example shows how to determine whether your catalog has structural errors.

```
//EXAMEX3   JOB
//STEP1     EXEC    PGM=IDCAMS
//SYSPRINT  DD      SYSOUT=A
//SYSIN     DD      *
     EXAMINE -
            NAME(ICFUCAT1) -
            NOINDEXTEST -
            DATATEST -
            ERRORLIMIT(1000)
/*
```

Use the EXAMINE command to analyze the data component of an integrated catalog facility catalog. Its parameters are:

- NAME, specifies the catalog name. The catalog must be connected to the master catalog.
- NOINDEXTEST, specifies that the index component is not to be examined.
- DATATEST, causes the data component to be examined.
- ERRORLIMIT(1000), restricts the printing of detailed error messages to 1000 errors.

# Chapter 23. EXPORT

The EXPORT command either exports a cluster or an alternate index or creates a backup copy of an integrated catalog facility catalog. An empty candidate volume cannot be exported. Access method services acknowledges and preserves the SMS classes during EXPORT.

Access method services does not use RLS. If an RLS keyword is specified on the DD statement of a file to be opened by AMS, the keyword will be ignored and the file will be opened and accessed in non-RLS mode.

The syntax of the EXPORT command is:

| | |
|---|---|
| **EXPORT** | *entryname*<br>**{OUTFILE(***ddname***)\|OUTDATASET(***entryname***)}**<br>**[CIMODE\|RECORDMODE]**<br>**[ERASE\|NOERASE]**<br>**[INFILE(***ddname***)]**<br>**[INHIBITSOURCE\|NOINHIBITSOURCE]**<br>**[INHIBITTARGET\|NOINHIBITTARGET]**<br>**[PURGE\|NOPURGE]**<br>**[TEMPORARY\|PERMANENT]** |

EXPORT can be abbreviated: EXP

**Notes:**

1. You can export a KSDS with extended addressability to a system that does not support extended addressability if the data set is smaller than 4GB. If it is larger, the EXPORT and IMPORT commands appear to complete successfully, but when the data set tries to extend beyond 4GB, a message is issued. You can use REPRO—specifically FROMKEY and TOKEY, or COUNT parameters—to reduce the data set to less than 4GB before using IMPORT.
2. VSAM record-level sharing (RLS) information is lost when the EXPORT is done on a DFSMS/MVS 1.2 or lower system. For the correct procedure to use when copying or moving data sets with pending recovery, see “Using VSAM Record-Level Sharing” in *z/OS DFSMS: Using Data Sets* and *CICS Recovery and Restart Guide* .

## EXPORT Parameters

The EXPORT command uses the following parameters.

## Required Parameters

*entryname*
Names the cluster, alternate index, or user catalog to be exported. This parameter must be the first parameter following EXPORT. If *entryname* specifies an SMS-managed data set, the OUTDATASET must either be an SMS-managed data set, or a non-SMS-managed data set cataloged in the catalog determined by the catalog search order (see “Order of Catalog Use” on page 9).

**OUTFILE(***ddname***)|OUTDATASET(***entryname***)**
Specifies the name of the DD statement or the data set that is to receive the data being exported.

© Copyright IBM Corp. 1973, 2002

Portable data sets loaded by EXPORT must be sequential data sets. VSAM is not a valid data set organization for portable data sets.

**OUTFILE(*ddname*) Abbreviation: OFILE**
Specifies the name of the DD statement of the target data set.

Only the block size for the DCB parameter should be specified in the DD statement. The default for block size for EXPORT is 2048. Block size can be given in the DD statement to override this default and improve performance.

**Exception:** For a nonlabeled tape, the LRECL should be specified if any of the input records are greater in size than the block size. Maximum record size is determined by the value specified with the maximum subparameter of the RECORDSIZE parameter of the DEFINE CLUSTER or DEFINE ALTERNATEINDEX command when the data set was defined.

**OUTDATASET(*entryname*) Abbreviation: ODS**
Specifies the name of the target data set. If OUTDATASET is specified, an attempt is made to dynamically allocate the target data set. The characteristics of the target data set are described in “JCL DD Statement for a Target Data Set” on page 4.

If OUTDATASET specifies an SMS-managed data set, the exported data set must either be an SMS-managed data set, or a non-SMS-managed data set cataloged in the catalog determined by the catalog search order. For information about this search order see “Order of Catalog Use” on page 9.

## Optional Parameters

The EXPORT command uses the following optional parameters.

**CIMODE|RECORDMODE**
Specifies whether control interval processing (CIMODE) or logical record processing (RECORDMODE) is to be used to export the records of the data set cluster. RECORDMODE is the default for ESDS, KSDS, and RRDS clusters. CIMODE is the default for LDS clusters.

**CIMODE Abbreviation: CIM**
Specifies that the cluster data records written to the portable data set are processed as one VSAM control interval. You can use CIMODE processing to export data sets more quickly. Each control interval is processed as one logical record.

If control interval processing is used, the target data set’s catalog entry will not have correct statistics. These statistics are correctly updated the first time the data set is opened for output.

When you use control interval processing to export an ESDS cluster that contains an alternate index, logical record processing is used, and a warning message is issued.

IMPORT will determine the type of processing (control interval or logical record) used by EXPORT to process the cluster data records, and use the same processing type for loading. Thus, a data set that was exported in control intervals is loaded in control intervals. Similarly, a data set exported in logical records is loaded by IMPORT as VSAM logical records.

The CIMODE portable data set created by the EXPORT command is not compatible with a CIMODE portable data set created on a VSE system.

Therefore, any attempt to import an object exported on a VSE system with control interval processing support is not detected by IMPORT and gives unpredictable results.

During CIMODE processing, data set statistics, such as the number of logical records and the number of inserted records, are not maintained. Data set statistics are not maintained because VSAM cannot update logical record information when whole control intervals are processed. After recalling a data set, a LISTCAT might not show accurate freespace bytes, and a read against the VVR might show zero records although records do exist.

**RECORDMODE Abbreviation: RECM**
Specifies that cluster data records written to the portable data set are processed as one VSAM logical record.

On a system without control interval processing support, RECORDMODE is the default.

For LDS clusters, the default is CIMODE.

**ERASE|NOERASE**
Specify whether or not the components of the cluster or alternate index to be exported are to be erased (that is, overwritten with binary zeros). This parameter overrides whatever was specified when the object was defined or last altered.

This parameter can be specified only if the object is to be permanently exported (that is, deleted from the original system). It does not apply to integrated catalog facility catalogs that must be exported as TEMPORARY.

**ERASE**
Specifies that the components are to be overwritten with binary zeros when the cluster or alternate index is deleted. If ERASE is specified, the volume that contains the data component must be mounted.

If the alternate index is protected by a RACF generic or discrete profile, use RACF commands to specify an ERASE attribute as part of this profile so that the data component is automatically erased upon deletion.

**Abbreviation:** ERAS

**NOERASE Abbreviation: NERAS**
specifies that the components are not to be overwritten with binary zeros when the cluster or alternate index is deleted.

NOERASE does not prevent the data component from being erased if it is protected by a RACF generic or discrete profile that specifies the ERASE attribute. You can use RACF commands to alter the ERASE attribute in a profile.

**INFILE(*ddname*) Abbreviation: IFILE**
Specifies the name of the DD statement that identifies the cluster, alternate index, or catalog to be exported. If the cluster, alternate index, or catalog has been defined with a maximum logical record length greater than 32760 bytes, EXPORT processing ends with an error message, except for EXPORT with control interval processing support.

In addition to the DD statement for INFILE, you must identify the entry's catalog with a JOBCAT or STEPCAT DD statement unless one of the following is true:

- The object's entry is in the master catalog.

- The qualifiers in the object's name are the integrated catalog facility catalog's name or alias.
- The object is an integrated catalog facility catalog.
- The entry's catalog is SMS-managed.

When INFILE and its DD statement are not specified for a to-be-exported object, an attempt is made to dynamically allocate the object with a disposition of **OLD**.

**INHIBITSOURCE|NOINHIBITSOURCE**
Specifies how the data records in the source data set (ALTERNATE INDEX and CLUSTER) can be accessed after they have been imported to another system. Use the ALTER command to alter this parameter.

**INHIBITSOURCE Abbreviation: INHS**
Specifies that the original data records in the original system cannot be accessed for any operation other than retrieval. Use it when the object is to be temporarily exported. (A backup copy of the object is made, and the object itself remains in the original system.)

If INHIBITSOURCE is specified when exporting an integrated catalog facility catalog, it is ignored and a warning message issued.

**NOINHIBITSOURCE Abbreviation: NINHS**
specifies that the original data records in the original system can be accessed for any kind of operation.

**INHIBITTARGET|NOINHIBITTARGET**
Specify whether or not the data records copied into the target alternate index or cluster can be accessed for any operation other than retrieval after they have been imported to another system. This specification can be altered through the ALTER command.

**INHIBITTARGET Abbreviation: INHT**
specifies that the target object cannot be accessed for any operation other than retrieval after it has been imported into another system.

If INHIBITTARGET is specified when exporting an integrated catalog facility catalog, it is ignored and a warning message is issued.

**NOINHIBITTARGET Abbreviation: NINHT**
Specifies that the target object can be accessed for any type of operation after it has been imported into another system.

**PURGE|NOPURGE**
Specify whether or not the cluster or alternate index to be exported is to be deleted from the original system regardless of the retention period specified in a TO or FOR parameter when the object was defined.

This parameter can be specified only if the object is to be permanently exported, that is, deleted from the original system. Therefore, it does not apply to catalogs that must be exported as TEMPORARY.

**PURGEAbbreviation: PRG**
Specifies that the object is to be deleted even if the retention period has not expired.

**NOPURGE Abbreviation: NPRG**
Specifies that the object is not to be deleted unless the retention period has expired.

**TEMPORARY|PERMANENT**

Specify whether or not the cluster, alternate index, or catalog to be exported is to be deleted from the original system.

**TEMPORARY Abbreviation: TEMP**

Specifies that the cluster, alternate index, or catalog is not to be deleted from the original system. The object in the original system is marked as temporary to indicate that another copy exists and that the original copy can be replaced.

To replace the original copy, a portable copy created by an EXPORT command must be imported to the original system. The IMPORT command deletes the original copy, defines the new object, and copies the data from the portable copy into the newly defined object. Portable data sets being loaded by EXPORT must be sequential data sets. VSAM is not a valid data set organization for portable data sets.

Integrated catalog facility catalogs are exported as TEMPORARY.

Be sure to properly protect the file of the temporary object if you want to deny unauthorized access to that file.

**PERMANENT Abbreviation: PERM**

Specifies that the cluster or alternate index is to be deleted from the original system. Its storage space is freed. If its retention period has not yet expired, you must also code PURGE.

If PERMANENT is specified when exporting an integrated catalog facility catalog, the catalog will still be exported as TEMPORARY, and a message is issued.

## EXPORT Examples

The EXPORT command can perform the functions shown in the following examples.

## Export an Integrated Catalog Facility Catalog: Example 1

In this example, the catalog, USERCAT4, is exported but not disconnected. The catalog is copied to a portable file, CATBACK, and its catalog entry is modified to indicate it was exported temporary. If the user catalog is cataloged in the master catalog, aliases of the catalog are also exported.

```
//EXPRTCAT JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//RECEIVE  DD    DSNAME=CATBACK,UNIT=(TAPE,,DEFER),
//         DISP=(NEW,KEEP),VOL=SER=327409,LABEL=(1,SL)
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     EXPORT -
           USERCAT4 -
           OUTFILE(RECEIVE) -
           TEMPORARY
/*
```

Job control language statements:

- RECEIVE DD describes the portable file that is to receive a copy of the catalog.

The EXPORT copies the catalog, USERCAT4, and its aliases to a portable file, CATBACK. The parameters are:

- USERCAT4 identifies the object to be exported.

- OUTFILE points to the RECEIVE DD statement. The RECEIVE DD statement describes the portable data set, CATBACK, that is to receive a copy of the catalog.
- TEMPORARY specifies that the catalog is not to be deleted. The catalog is marked “temporary” to indicate that another copy exists and that the original copy can be replaced. This is a required parameter when exporting an integrated catalog facility catalog that cannot be exported with the PERMANENT parameter.

## Export a Key-Sequenced Cluster: Example 2

In this example, a key-sequenced cluster, ZZZ.EXAMPLE.KSDS1, is exported from a user catalog, HHHUCAT1. The cluster is copied to a portable file, TAPE2, and its catalog entries are modified to prevent the cluster’s data records from being updated, added to, or erased.

```
//EXPORT1  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//RECEIVE  DD    DSNAME=TAPE2,UNIT=(TAPE,,DEFER),
//         DISP=NEW,VOL=SER=003030,
//         DCB=(BLKSIZE=6000,DEN=3),LABEL=(1,SL)
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     EXPORT -
           ZZZ.EXAMPLE.KSDS1 -
           OUTFILE(RECEIVE) -
           TEMPORARY -
           INHIBITSOURCE
/*
```

Job control language statement:

- RECEIVE DD describes the portable file, a magnetic tape file, that is to receive a copy of the cluster’s records. The DCB BLKSIZE parameter overrides the EXPORT default of 2048 to improve performance.

The EXPORT command copies key-sequenced cluster, ZZZ.EXAMPLE.KSDS1, and its cataloged attributes to a portable file, TAPE2. The parameters are:

- ZZZ.EXAMPLE.KSDS1 identifies the cluster to be exported.
- OUTFILE points to the RECEIVE DD statement. The RECEIVE DD statement describes the portable file, TAPE2, that is to contain a copy of the cluster.
- TEMPORARY specifies that the cluster is not to be deleted. The cluster’s catalog entry is marked “temporary” to indicate that another copy of the cluster exists and that the original copy can be replaced. (See the IMPORT Example, “Import a Key-Sequenced Cluster: Example 3” on page 259.)
- INHIBITSOURCE specifies that the copy of the cluster that remains in the original system, as a result of TEMPORARY, cannot be modified. User programs are allowed only to read the cluster’s records.

## Export an Entry-Sequenced Cluster: Example 3

In this example, an entry-sequenced cluster is exported to a portable file and then deleted from the system.

```
//EXPORT2  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//RECEIVE  DD    DSNAME=TAPE1,UNIT=(TAPE,,DEFER),
//    VOL=SER=001147,LABEL=(1,SL),DISP=NEW
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     EXPORT -
```

```
          X98.EXAMPLE.ESDS1 -
          OUTFILE(RECEIVE) -
          PURGE
/*
```

Job control language statement:

- RECEIVE DD describes the portable file, TAPE1, that is to contain a copy of the exported entry-sequenced cluster.

The EXPORT command copies the entry-sequenced cluster, X98.EXAMPLE.ESDS1, and its cataloged attributes to a portable file, TAPE1. The cluster is deleted from the system after it is copied into the portable file. The parameters are:

- X98.EXAMPLE.ESDS1 identifies the entry-sequenced cluster to be exported.
- OUTFILE points to the RECEIVE DD statement. The RECEIVE DD statement describes the portable data set, TAPE1, that is to receive a copy of the cluster.
- PURGE allows the cluster to be deleted regardless of its retention period or date.

Because EXPORT defaults to PERMANENT, the cluster is deleted after its contents are copied to TAPE1.

Because EXPORT defaults to NOINHIBITTARGET, access method services assumes the cluster can be updated (by users of the other system) when it is imported to another system.

## Export an Entry-Sequenced Cluster Using CIMODE: Example 4

In this example, a VSAM data set, USERDS1, is exported, using control interval processing. The user data is copied to a portable file, BACKUP.USERDS1.CIMODE, and its catalog entry is modified to indicate that it was temporarily exported.

```
//EXPRTUSR  JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//RECEIVE   DD    DSNAME=BACKUP.USERDS1.CIMODE,UNIT=(TAPE,,DEFER),
//          DISP=(NEW,KEEP),VOL=SER=327409,LABEL=(1,SL)
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    *
     EXPORT -
            USERDS1 -
            OUTFILE(RECEIVE) -
            TEMPORARY        -
            CIMODE
/*
```

Job Control Statement:

- RECEIVE DD describes the portable file that receives a copy of the ESDS cluster (BACKUP.USERDS1.CIMODE).

The parameters of the EXPORT command are:

- USERDS1 identifies the object to be exported.
- OUTFILE points to the RECEIVE DD statement. This statement describes the portable data set, BACKUP.USERDS1.CIMODE, that is to receive a copy of the ESDS cluster.
- TEMPORARY specifies that the cluster is not to be deleted. The entry of the data set in the catalog is marked “temporary” to indicate that another copy of this data set exists and that the original copy can be replaced.
- CIMODE specifies that control interval processing is to be used to process the data one control interval at a time instead of one record at a time.

## Export Multiple Data Sets Using Multiple INFILE Parameters: Example 5

In this example, multiple VSAM data sets are exported in the same step using multiple INFILE parameters.

```
//EXPORT   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//INDS1    DD   DSNAME=MTD.CLUSTER1,DISP=OLD
//INDS2    DD   DSNAME=MTD.CLUSTER2,DISP=OLD
//INDS3    DD   DSNAME=MTD.CLUSTER3,DISP=OLD
//INDS4    DD   DSNAME=MTD.CLUSTER4,DISP=OLD
//PORTDS1  DD   DSNAME=CLUSBAC1,UNIT=3380,VOL=SER=338001,DISP=(NEW,KEEP),
//         SPACE=(TRK,(10,2)),DCB=(RECFM=F,LRECL=4101,BLKSIZE=4401)
//PORTDS2  DD   DSNAME=CLUSBAC2,UNIT=3380,VOL=SER=338001,DISP=(NEW,KEEP),
//         SPACE=(TRK,(10,2)),DCB=(RECFM=F,LRECL=4101,BLKSIZE=4401)
//PORTDS3  DD   DSNAME=CLUSBAC3,UNIT=3380,VOL=SER=338001,DISP=(NEW,KEEP),
//         SPACE=(TRK,(10,2)),DCB=(RECFM=F,LRECL=4101,BLKSIZE=4401)
//PORTDS4  DD   DSNAME=CLUSBAC4,UNIT=3380,VOL=SER=338001,DISP=(NEW,KEEP),
//         SPACE=(TRK,(10,2)),DCB=(RECFM=F,LRECL=4101,BLKSIZE=4401)
//SYSIN    DD   *
     EXPORT -
           MTD.CLUSTER1 -
           INFILE(INDS1) -
           OUTFILE(PORTDS1)
     EXPORT -
           MTD.CLUSTER2 -
           INFILE(INDS2) -
           OUTFILE(PORTDS2)
     EXPORT -
           MTD.CLUSTER3 -
           INFILE(INDS3) -
           OUTFILE(PORTDS3)
     EXPORT -
           MTD.CLUSTER4 -
           INFILE(INDS4) -
           OUTFILE(PORTDS4)
/*
```

Job control language statements:

- INDS1 through INDS4 allocate the data sets to be exported.
- PORTDS1 through PORTDS4 describe the portable files that are to contain a copy of the exported data sets.

The parameters of the EXPORT command are:

- MTD.CLUSTER1 through MTD.CLUSTER4 specify the data sets to be exported.
- INFILE points to the INDS1 through INDS4 statements.
- OUTFILE points to the PORTDS1 through PORTDS4 statements. These statements describe the portable data sets, CLUSBAC1 through CLUSBAC4, that are to receive a copy of data sets INDS1 through INDS4.

# Chapter 24. EXPORT DISCONNECT

The EXPORT DISCONNECT command disconnects a user catalog. The syntax of the EXPORT DISCONNECT command is:

| | |
|---|---|
| **EXPORT** | *usercatname*<br>**DISCONNECT**<br>**[CATALOG(***catname***)]** |

## EXPORT DISCONNECT Parameters

## Required Parameters

*usercatname*
names the user catalog to be disconnected. This parameter must be the first parameter following EXPORT. When you are disconnecting a user catalog, you must supply the alter authority to the catalog from which the entry is being removed.

If the user catalog is SMS-managed, its volume serial number is indicated at the time of disconnect.

**CATALOG(***catname***)**
specifies, for a disconnect operation, the name of the user catalog from which a user catalog connector entry and any associated alias entries are to be deleted. See “Catalog Selection Order for EXPORT DISCONNECT” on page 12 for the order in which a catalog is selected when the CATALOG parameter is not specified.

*catname*
specifies the name of the catalog from which a user catalog connector entry and any associated alias entries are to be deleted.

**Abbreviation:** CAT

**DISCONNECT**
specifies that a user catalog is to be disconnected. The connector entry for the user catalog is deleted from the master catalog. Also, the user catalog’s alias entries are deleted from the master catalog.

If EXPORT is coded to remove a user catalog connector entry, DISCONNECT is a required parameter. The VVDS volume and the BCS volume can be physically moved to the system to which the catalog is connected.

To make a user catalog available in other systems and in the original system, code the IMPORT CONNECT command to connect the user catalog to each system to which it is to be available, but do not EXPORT DISCONNECT the user catalog.

EXPORT DISCONNECT displays the volume serial number of the user catalog at the time of the disconnect. This volume serial number information is required to perform the IMPORT CONNECT.

**Abbreviation:** DCON

© Copyright IBM Corp. 1973, 2002

# EXPORT DISCONNECT Examples

The EXPORT DISCONNECT command can perform the functions shown in the following examples.

## Export Disconnect of a User Catalog from Another User Catalog: Example 1

The following example shows the EXPORT command being used to disconnect a user catalog from another user catalog.

```
//EXPDISC  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     EXPORT -
           RBLUCAT2 -
           DISCONNECT -
           CATALOG(RBLUCAT1)
/*
```

The EXPORT command removes the user catalog connector entry for RBLUCAT2 from user catalog RBLUCAT1. The parameters are:

- RBLUCAT2 identifies the object to be disconnected.
- DISCONNECT identifies the object as a user catalog.
- CATALOG names the user catalog (RBLUCAT1) containing the connector entry being disconnected.

## Export Disconnect of a User Catalog: Example 2

In this example, the user catalog 387UCAT1 is disconnected from the system. Its cataloged objects are no longer available to users of the system.

```
//EXPORT3  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     EXPORT -
           G87UCAT -
           DISCONNECT
/*
```

The EXPORT command removes the user catalog connector entry for G87UCAT from the master catalog. The catalog becomes unavailable to system users until the system programmer reconnects it to the system, using an IMPORT CONNECT command. The parameters are:

- G87UCAT identifies the object to be disconnected.
- DISCONNECT identifies the object as a user catalog. When a user catalog's connector entry is to be deleted, DISCONNECT is required.

# Chapter 25. IMPORT

The IMPORT command moves or restores a cluster or alternate index, or restores an integrated catalog facility catalog. The syntax of the IMPORT command is:

```
IMPORT   {INFILE(ddname)|INDATASET(entryname)}
         {OUTFILE(ddname)| OUTDATASET(entryname)}
         [ALIAS|NOALIAS]
         [ERASE|NOERASE]
         [INTOEMPTY]
         [LOCK|UNLOCK]
         [OBJECTS
           ((entryname
             [FILE(ddname)]
             [MANAGEMENTCLASS(class)]
             [NEWNAME(newname)]
             [STORAGECLASS(class)]
             [VOLUMES(volser[ volser...])])
          [(entryname...)...])]
         [PURGE|NOPURGE]
         [SAVRAC|NOSAVRAC]
         [CATALOG(catname)]
```

IMPORT can be abbreviated: IMP

**Restrictions:**

1. The original version of the catalog is **always** deleted when you use IMPORT.
2. You can export a KSDS with extended addressability to a system that does not support extended addressability if the data set is smaller than 4GB. If it is larger, the EXPORT and IMPORT commands appear to complete successfully, but when the data set tries to extend beyond 4GB, a message is issued. You can use REPRO—specifically FROMKEY and TOKEY, or COUNT parameters—to reduce the size of the data set to less than 4GB before using IMPORT.
3. VSAM record-level sharing (RLS) information is lost when the IMPORT is done on a DFSMS/MVS 1.2 or lower system. For the correct procedure to use when copying or moving data sets with pending recovery, see “Using VSAM Record-Level Sharing” in *z/OS DFSMS: Using Data Sets* and *CICS Recovery and Restart Guide* .
4. Access Method Services does not use RLS. If an RLS keyword is specified on the DD statement of a file to be opened by AMS, the keyword will be ignored and the file will be opened and accessed in non-RLS mode.

## IMPORT Parameters

The IMPORT command uses the following parameters.

## Required Parameters

**INFILE(*ddname*)|INDATASET(*entryname*)**
specifies the name of a DD statement or names the portable data set that contains a copy of the cluster, alternate index, or user catalog to be imported.

© Copyright IBM Corp. 1973, 2002

When importing into a nonexistent or an existing nonempty data set or catalog, the names specified for management class and storage class in the IMPORT command override the management class and storage class names from the portable data set. The class specifications and other attributes of the exported object are used to determine the SMS class specifications.

**INFILE(*ddname*)**
specifies the name of a DD statement that identifies the portable copy of the cluster, alternate index, or user catalog to be imported.

If a nonlabeled tape or a direct access data set created by DOS/VS access method services contains the copy, the following DCB parameters must be specified on the referenced DD statement:

- BLKSIZE. If you specified BLKSIZE when the cluster or alternate index was exported, you must specify the same block size value for IMPORT. If you did not specify a block size for EXPORT, a default value of 2048 was used. Consequently, if you do not specify BLKSIZE for IMPORT, IMPORT sets the block size to 2048.
- LRECL. LRECL is based on the maximum record size of the exported VSAM data set. Maximum record size is determined by the value given by the maximum subparameter of the RECORDSIZE parameter of the DEFINE CLUSTER or DEFINE ALTERNATEINDEX command when the data set was defined.
- RECFM. Must be VBS.

**Abbreviation:** IFILE

**INDATASET(*entryname*)**
specifies the name of the portable data set that contains a copy of the cluster, alternate index, or user catalog to be imported.

If INDATASET is specified, the portable data set is dynamically allocated. The entryname must be cataloged in a catalog that is accessible by the system into which the entry is to be imported.

**Abbreviation:** IDS

**OUTFILE(*ddname*)|OUTDATASET(*entryname*)**
specifies the name of a DD statement or the name of a cluster, alternate index, or user catalog to be imported.

When you use OUTFILE or OUTDATASET to describe the data set, you must identify the data set's catalog with a JOBCAT or STEPCAT DD statement unless one of the following is true:

- The data set's entry is in the master catalog.
- The qualifiers in the data set's qualified name are the catalog's name or alias.
- The data set is SMS-managed.
- You are importing a non-SMS-managed integrated catalog facility user catalog:
  - When importing a cluster that was permanently exported, the OUTFILE parameter should be used.
  - If a concatenated JOBCAT or STEPCAT DD statement is used, the catalog to be imported must be identified in the first DD statement unless the catalog is specified with the CATALOG parameter.
  - If you are importing to a volume other than the original volume, the OBJECTS(VOLUMES) parameter must also be specified.

**OUTFILE(*ddname*)**
specifies the name of a DD statement that identifies the data set name and volumes of the cluster, alternate index, or user catalog that is to be imported.

If the object was permanently exported or you are importing to a volume other than the original volume, the DD statement specifies the name of the cluster or alternate index as DSNAME, the volume serial number, the device type, DISP=OLD, and AMP='AMORG'.

If the object has its data and index components on different device types, specify OUTDATASET instead of OUTFILE.

If the NEWNAME parameter is specified for the cluster or alternate index entry, the data set name on the DD statement must be the same as the new name. Failure to do so will result in the deletion of the original cluster.

**Abbreviation:** OFILE

**OUTDATASET(*entryname*)**
specifies the name of the cluster, alternate index, or user catalog that is to be imported. If you select OUTDATASET, the VSAM data set you identify is dynamically allocated.

You can use concatenated DD statements if the object was permanently exported and its data and index components are on different device types. The first DD statement specifies the name of the cluster or alternate index as the DSNAME, the volume serial numbers and device type of the data component, DISP=OLD, and AMP=AMORG. The second DD statement specifies the name of the index component as the DSNAME, the volume serial numbers and device type of the index component, DISP=OLD, and AMP=AMORG.

If NEWNAME is specified for the cluster or alternate index entry, *entryname* must be the same as the new name. Also, this should be the same name as declared on the NEWNAME parameter. Failure to do so will result in the deletion of the original cluster.

**Abbreviation:** ODS

## Optional Parameters

The IMPORT command can use the following optional parameters.

**ALIAS|NOALIAS**
specifies whether any aliases are defined for the imported catalog. ALIAS causes the IMPORT command to retrieve the aliases that were exported and define them for the catalog being imported. The default, NOALIAS, will result in no aliases being imported.

If ALIAS is specified, and the catalog

- exists but is empty, any aliases that exist in the system for that catalog are not deleted. Any aliases that exist on the portable data set are defined for the imported catalog if the aliases do not exist on the system. Any duplicate aliases will produce a duplicate alias message. IMPORT will print the list of aliases that were defined for the imported catalog.
- exists and is not empty, the catalog is deleted and redefined from the portable data set. Any aliases that exist in the system are not deleted. Aliases on the portable data set are not defined but a list of the alias names from the portable data set are printed.

- does not exist, the catalog is defined along with its aliases from the portable data set. The catalog will then be loaded from the portable data set.

**Hint:** Before restoring the catalog, you might want to run LISTCAT to determine the status of the catalog and its aliases.

**Abbreviations:** ALS NALS

**CATALOG(*catname*)**
specifies the name of the catalog in which the imported object is to be cataloged.

*catname*
is the name of the catalog into which to define the entry being imported.

**Abbreviation:** CAT

**ERASE|NOERASE**
specifies whether the component of the cluster or alternate index is to be erased (that is, overwritten with binary zeros). Use this parameter when you are importing the object into the system from which it was previously exported with the TEMPORARY option. This parameter overrides whatever was specified when the object was defined or last altered.

**ERASE**
overwrites the component with binary zeros when the cluster or alternate index is deleted. If ERASE is specified, the volume that contains the data component must be mounted.

If the cluster is protected by a RACF generic or discrete profile and the cluster is cataloged in an integrated catalog facility catalog, use RACF commands to specify an ERASE attribute as part of this profile so that the component is automatically erased upon deletion.

**Abbreviation:** ERAS

**NOERASE**
specifies that the component is not to be overwritten with binary zeros when the cluster or alternate index is deleted.

NOERASE resets only the indicator in the catalog entry that was created from a prior DEFINE or ALTER command. If the cluster is protected by a RACF generic or discrete profile that specifies the ERASE attribute and if the cluster is cataloged in an integrated catalog facility catalog, it is erased upon deletion. Use RACF commands to alter the ERASE attribute in a profile.

**Abbreviation:** NERAS

**INTOEMPTY**
specifies that you are importing from the portable data set into an empty data set. If this parameter is not specified, an attempt to import into an empty data set is unsuccessful. If you import into an empty SMS-managed data set or catalog, the SMS class specifications in effect are not changed. MANAGEMENTCLASS and STORAGECLASS from the portable data set are ignored.

The RACF profiles associated with the empty non-SMS-managed data set are retained.

You can use INTOEMPTY to import a previously SMS-managed data set into a predefined empty non-SMS-managed data set.

When importing into an empty data set, the SAVRAC|NOSAVRAC parameter applies only to the paths imported and successfully defined over the empty data set. If the define of an exported path is unsuccessful because a catalog entry with the same name already exists, the path on the portable data set is ignored.

**Abbreviation:** IEMPTY

**LOCK|UNLOCK**
controls the setting of the catalog lock attribute, and therefore checks access to a catalog. LOCK or UNLOCK can be specified only when entryname identifies an integrated catalog facility catalog. If the LOCK|UNLOCK parameter is not specified, the catalog being imported will be unlocked. Before you lock a catalog, review the information on locking catalogs in *z/OS DFSMS: Managing Catalogs*. Locking a catalog makes it inaccessible to all users without read authority to RACF facility class profile IGG.CATLOCK (including users sharing the catalog on other systems).

**LOCK**
specifies that the catalog being imported is to be locked. Nonexisting catalogs are defined as locked. Existing unlocked catalogs are locked. Locking the catalog restricts catalog access to authorized personnel. Specification of this parameter requires read authority to the profile name, IGG.CATLOCK, with class type FACILITY. Integrated catalog facility catalogs are usually locked only during catalog recovery operations.

**UNLOCK**
specifies that the catalog being imported is to be unlocked. Nonexisting catalogs are defined as unlocked. Existing locked catalogs are unlocked. If LOCK|UNLOCK is not specified, the catalog is unlocked.

**OBJECTS**

**((***entryname*

**[FILE(***ddname***)]**

**[MANAGEMENTCLASS(***class***)]**

**[NEWNAME(***newname***)]**

**[STORAGECLASS(***class***)]**

**[VOLUMES(***volser***[** *volser***...])])**

**[(***entryname***...)...])**
specifies the new or changed attributes for the cluster, alternate index, any associated paths, or user catalog to be imported. Access method services matches each *entryname* you specify against the name of each object on the portable data set. When a match is found, the information specified by OBJECTS overrides the information on the portable data set.

If you specify NEWNAME when importing an integrated catalog facility catalog, an error message is issued and processing ends.

*entryname*
specifies the name of the data component, index component, cluster, alternate index, path, or user catalog for which attributes are being specified. The entryname must appear on the portable data set; otherwise, the parameter list is ignored.

**Abbreviation:** OBJ

**FILE(***ddname***)**
specifies the name of a DD statement that identifies the volumes allocated

to the data and index components of a key-sequenced cluster, an alternate index, or user catalog. This parameter is used when the data and index components reside on different device types. FILE can be coded twice within the OBJECTS parameter: once in the parameter set for the index component and once in a second parameter set for the data component. If you do not specify FILE, the required volumes are dynamically allocated. The volumes must be mounted as permanently resident and reserved.

**MANAGEMENTCLASS(***class***)**
specifies a 1-to-8 character management class name to be associated with the data set or catalog being imported. It must be associated with the entry name of the CLUSTER or the AIX.

**Abbreviation:** MGMTCLAS

**NEWNAME(***newname***)**
specifies the new name of an imported cluster or alternate index or its components, or an associated path. If you use NEWNAME, only the name specified as *entryname* is changed.

**Restriction:** The NEWNAME parameter is not valid when importing an integrated catalog facility catalog.

If you are specifying a new name for a cluster or alternate index that was exported with the TEMPORARY option and it is being imported back into the original system, you must also rename each of its components. If you are specifying NEWNAME for an SMS-managed cluster or alternate index, you must also rename each of its components, so that each component orients to the same user catalog.

**Abbreviation:** NEWNM

**ORDERED|UNORDERED**
ORDERED|UNORDERED is no longer supported; if it is specified, it will be ignored and no error message will be issued.

**STORAGECLASS(***class***)**
specifies a 1-to-8 character storage class name to be associated with the data set or catalog being imported. It must be associated with the entry name of the CLUSTER or the AIX.

**Abbreviation:** STORCLAS

**VOLUMES(***volser***[** *volser***...])**
specifies either the volumes on which the cluster, alternate index, or user catalog is to reside, or the volume on which the user catalog resides. If VOLUMES is not coded, the original volume is the receiving volume.

SMS might not use candidate volumes for which you request specific volsers. In some cases, a user-specified volser for an SMS-managed data set can result in an error. To avoid candidate volume problems with SMS, you can request that SMS choose the specific volser used for a candidate volume. To do this, you can code an * for each volser that you request. If, however, you request both specified and unspecified volsers in the same command, you must enter the specified volsers first in the command syntax. The default is one volume. For SMS-managed data sets, you can specify up to 59 volume serial numbers.

Catalogs can only be on one volume, so only one volume should be specified when importing a user catalog.

If you use VOLUMES, you can specify the cluster or alternate index name, the data component name or the index component name as entryname with the following results:

- If VOLUMES is specified with the cluster or alternate index name, the specified volume list is defined for the data component. For a key-sequenced cluster or alternate index, the specified volume list is also defined for the index component.
- If VOLUMES is specified with the data component name, the specified volume list is defined for the data component. Any specification of VOLUMES with the cluster or alternate index name is overridden.
- For a key-sequenced cluster or alternate index, if VOLUMES is specified with the index component name, the specified volume list is defined for the index component. Any specification of VOLUMES with the cluster or alternate index name is overridden.

If a guaranteed space storage class is assigned to the data sets (cluster) and volume serial numbers are specified, space is allocated on all specified volumes if the following conditions are met:

- All volumes specified are in the same Storage Group.
- The storage group to which these volumes belong is in the list of storage groups selected by the ACS routines for this allocation.

For clusters or alternate indexes, if multiple volumes are specified, they must be of the same device type. By repeating the OBJECTS parameter set for each component and including VOLUMES in each parameter set, you can have the data and index components on different volumes. Although the index and data components can reside on different device types, each volume of a multivolume component must be of the same type.

If the receiving volume is different from that which originally contained the cluster or alternate index, the job might end because of allocation problems. Each space allocation quantity is recorded in a catalog entry as an amount of cylinders or tracks even if RECORDS was specified in the DEFINE command.

When a cluster or alternate index is imported, the number of cylinders or tracks in the catalog entry is not modified, even though the object might be imported to reside on a device type other than that it was exported from. If an object is exported from a smaller DASD and imported to a larger DASD, more space is allocated than the object needs. Conversely, if an attempt is made to import an object that previously resided on a larger DASD to a smaller DASD, it might be unsuccessful.

You can avoid space allocation problems by defining an empty cluster or alternate index and identifying it as the target for the object being imported as described below:

- Use the DEFINE command to define a new entry for the cluster or alternate index in the catalog to which it is to be moved. If space was allocated in RECORDS, you can specify the same quantity; if it was allocated in TRACKS or CYLINDERS, you must adjust the quantity for the new device type. If an entry already exists in the catalog for the object, you must delete that entry or use a different name in the DEFINE command.

- Use the IMPORT command to load the portable data set into the newly defined cluster or alternate index. When IMPORT encounters an empty target data set, the exported catalog information is bypassed and only the data records are processed.

**Abbreviation:** VOL

**PURGE|NOPURGE**
specifies whether the original cluster, alternate index, or catalog is to be deleted and replaced, regardless of the retention time specified in the TO or FOR parameter. Use this parameter when you are importing the object into the original system from which it was exported with the TEMPORARY option.

**PURGE**
specifies that the object is to be deleted even if the retention period has not expired.

**Abbreviation:** PRG

**NOPURGE**
specifies that the object is not to be deleted unless the retention period has expired.

**Abbreviation:** NPRG

**SAVRAC|NOSAVRAC**
specifies, for a RACF-protected object, whether existing profiles are to be used or whether new profiles are to be created. This option applies only to discrete profiles. Generic profiles are not affected.

**Exception:** The SAVRAC|NOSAVRAC parameters are ignored if the INTOEMPTY parameter has been specified and the target data set exists and is empty.

**SAVRAC**
specifies that RACF data set profiles that already exist for objects being imported from the portable data set are to be used. Typically, you would specify this option when replacing a data set with a portable copy made with an EXPORT TEMPORARY operation. SAVRAC causes the existing profiles to be saved and used, rather than letting the system delete old profiles and create new, default profiles.

The profiles will actually be redefined by extracting information from existing profiles and adding caller attributes. You should ensure these added attributes are acceptable.

The ownership creation group and access list are altered by the caller of the SAVRAC option.

**Attention:** Ensure that valid profiles do exist for clusters being imported when SAVRAC is specified. If this is not done, a not valid and improper profile might be “saved” and used inappropriately.

**NOSAVRAC**
specifies that new RACF data set profiles are to be created. This is usually the situation when importing a permanently exported cluster.

If the automatic data set protection option has been specified for you or if the exported cluster had a RACF indication in the catalog when it was exported, a profile is defined for the imported clusters.

If you import into a catalog containing a component with a duplicate name that is marked as having been temporarily exported, it, and any associated profiles, is deleted before the portable data set is imported.

## IMPORT Examples

The IMPORT command can perform the functions in the following examples.

## Import a Cluster Utilizing SMS Keywords: Example 1

In this example, the IMPORT command is used with the SMS keyword STORAGECLASS to import an entry-sequenced cluster, HRB.EXAMPLE.ESDS1, from a portable file, TAPE1. The cluster and data components are renamed. The clusters storage class is derived by the storage class selection routines using the specified value as input. If the storage class selection routine assigns a storage class name, the management class is derived by the management class selection routines using the value in the portable data set as input.

```
//IMPORT   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SOURCE   DD   DSN=TAPE1,UNIT=(TAPE,,DEFER),DISP=OLD,
//   VOL=SER=022585,LABEL=(1,SL)
//SYSIN    DD   *
        IMPORT -
           INFILE(SOURCE) -
           OUTDATASET(K83.EXAMPLE.SMS.ESDS1) -
           OBJECTS( -
             (HRB.EXAMPLE.ESDS1 -
              NEWNAME(K83.EXAMPLE.SMS.ESDS1)) -
             (HRB.EXAMPLE.ESDS1.DATA -
              NEWNAME(K83.EXAMPLE.SMS.ESDS1.DATA) -
              STORAGECLASS(FAST)))
/*
```

**Requirement:** The ALIAS entries for HRB and K83 must point to the same user catalog.

Job control statement:

- SOURCE DD describes the portable file, TAPE1. which, resides on a magnetic tape file that are not mounted by the operator until access method services opens TAPE1 for processing.

The IMPORT command moves the contents of TAPE1 into the system. Access method services reorganizes the data records. The parameters are:

- INFILE points to the SOURCE DD statement.
- OUTDATASET gives the name of the renamed cluster. In this example, the data set either might not exist or, if it does exist, it must not be empty because INTOEMPTY is not specified.
- OBJECTS changes some of the attributes for the object being imported:
  - HRB.EXAMPLE.ESDS1 identifies the entry-sequenced cluster as it is currently named on TAPE1.
  - NEWNAME specifies that the cluster's name is to be changed to K83.EXAMPLE.SMS.ESDS1.
  - STORAGECLASS specifies that the data set requires the storage class, FAST. If the data set K83.EXAMPLE.SMS.ESDS1 existed at the time of the import and was not empty, it would be deleted and redefined. If the data set is redefined, the storage class used for redefinition is derived by the storage class selection routines using FAST as input. The management class used for

redefinition is derived by the management class selection routines using the management class in effect when the object was exported.

– HRB.EXAMPLE.ESDS1.DATA identifies the data component as it is currently named.
– NEWNAME specifies that the data component's name it to be changed to K83.EXAMPLE.SMS.ESDS1.DATA.

## Import an Integrated Catalog Facility Catalog: Example 2

In this example, an integrated catalog facility catalog, USERCAT4, that was previously exported, is imported. (See the EXPORT example, "Export an Integrated Catalog Facility Catalog: Example 1" on page 243.) The original copy of USERCAT4 is replaced with the imported copy, from the portable file copy in CATBACK. Access method services finds and deletes the duplicate name, USERCAT4. Any aliases in the master catalog for USERCAT4 are preserved. (A duplicate name exists because the catalog was exported with the TEMPORARY attribute.) Access method services then redefines USERCAT4, using the catalog information from the portable file, CATBACK. USERCAT4 is locked to prevent access to anyone except authorized recovery personnel.

**Requirement:** Before you can lock a catalog, you must have read authority to the profile name, IGG.CATLOCK, with class type FACILITY.

```
//IMPRTCAT JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SOURCE   DD    DSNAME=CATBACK,UNIT=3380,
//   VOL=SER=327409,DISP=OLD
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     IMPORT -
           INFILE(SOURCE) -
           OUTDATASET(USERCAT4) -
           ALIAS -
           LOCK -
           CATALOG(ICFMAST1)
/*
```

Job control language statement:

- SOURCE DD describes the portable data set, CATBACK.

The IMPORT command copies the portable data set, CATBACK, into the system. Access method services reorganizes the data records so that deleted records are removed and control intervals and control areas contain the specified free space percentages. The original copy of the cluster is deleted and replaced with the data records from the CATBACK portable file. The IMPORT command's parameters are:

- INFILE points to the SOURCE DD statement.
- OUTDATASET gives the name of the catalog being imported. Access method services dynamically allocates the catalog.
- ALIAS specifies that aliases that already exist in the master catalog for USERCAT4 are to be preserved and that the aliases on the portable file are to be listed. However, if USERCAT4 had not existed in the system when importing, the aliases on the portable file would have been defined for USERCAT4.
- LOCK specifies that the catalog being imported is locked.
- CATALOG identifies the master catalog, ICFMAST1.

## Import a Key-Sequenced Cluster: Example 3

In this example, a key-sequenced cluster, BCN.EXAMPLE.KSDS1, that was previously exported, is imported. (See the EXPORT example, “Export a Key-Sequenced Cluster: Example 2” on page 244.) OUTFILE and its associated DD statement are provided to allocate the data set.

The original copy of BCN.EXAMPLE.KSDS1 is replaced with the imported copy, TAPE2. Access method services finds and deletes the duplicate name, BCN.EXAMPLE.KSDS1, in the catalog, VCBUCAT1. (A duplicate name exists because TEMPORARY was specified when the cluster was exported.) Access method services then redefines BCN.EXAMPLE.KSDS1, using the catalog information from the portable file, TAPE2.

```
//IMPORT2  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SOURCE   DD    DSNAME=TAPE2,UNIT=(TAPE,,DEFER),
//   VOL=SER=003030,DISP=OLD,
//   DCB=(BLKSIZE=6000,LRECL=479,DEN=3),LABEL=(1,SL)
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     IMPORT -
           INFILE(SOURCE) -
           OUTDATASET(BCN.EXAMPLE.KSDS1) -
           CATALOG(VCBUCAT1)
/*
```

Job control language statement:

- SOURCE DD describes the portable data set, TAPE2, which resides on a magnetic tape file, that is not mounted by the operator until access method services opens TAPE2 for processing. The block size parameter is included (even though it need not be, because the tape has a standard label and the information is contained in the data set header label) to illustrate the fact that the information specified when the data set is imported is required to be the same as was specified when the data set was exported. The LRECL parameter is not required, because the maximum record size is 475 bytes and the default (block size minus 4) is adequate. However, by specifying a record size, the default is overridden and virtual storage is more efficiently used. To specify a record size, specify the size of the largest record + 4.

The IMPORT command copies the portable data set, TAPE2, into the system and assigns it the name BCN.EXAMPLE.KSDS1. When TAPE2 is copied, access method services reorganizes the data records so that deleted records are removed and control intervals and control areas contain the specified free space percentages. The original copy of the cluster is deleted and replaced with the data records from the TAPE2 portable file. The parameters are:

- INFILE points to the SOURCE DD statement, which describes the portable file, TAPE2, to be imported.
- OUTDATASET gives the name of the data set being imported. Because the high-level qualifier of the data set is the name of the alias of the user catalog VCBUCAT2, access method services can dynamically allocate the cluster without specifying a JOBCAT or STEPCAT DD statement.
- CATALOG identifies the catalog, VCBUCAT1, in which the imported cluster is to be defined.

## Import an Entry-Sequenced Cluster in a Catalog: Example 4

In this example, an entry-sequenced cluster, X98.EXAMPLE.ESDS1, is imported from a portable file, TAPE1. This example is associated with EXPORT example,

"Export an Entry-Sequenced Cluster: Example 3" on page 244. The cluster is defined in a different catalog than the one from which it was exported, assigned a new name, and imported to a different volume.

```
//IMPORT3  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SOURCE   DD   DSNAME=TAPE1,UNIT=(TAPE,,DEFER),DISP=OLD,
//   VOL=SER=001147,LABEL=(1,SL)
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     IMPORT -
           INFILE(SOURCE) -
           OUTDATASET(BCN.EXAMPLE.ESDS3) -
           OBJECTS -
             ((X98.EXAMPLE.ESDS1 -
               NEWNAME(BCN.EXAMPLE.ESDS3) -
               VOLUMES(VSER02))) -
            CATALOG(VCBUCAT1)
/*
```

Job control language statement:

- SOURCE DD describes the portable file, TAPE1, which resides on a magnetic tape file, that is not mounted by the operator until access method services opens TAPE1 for processing.

The IMPORT command moves the contents of TAPE1 into the system. Access method services reorganizes the data records. The parameters are:

- INFILE points to the SOURCE DD statement.
- OUTDATASET gives the name of the renamed cluster. Because the high-level qualifier of the data set name is the name of the alias of the catalog, VCBUCAT1, the data set can be dynamically allocated without specifying a JOBCAT or STEPCAT DD statement.
- OBJECTS changes some of the attributes for the object being imported:
  - X98.EXAMPLE.ESDS1 identifies the entry-sequenced cluster as it is currently named on TAPE1.
  - NEWNAME specifies that the cluster's name is to be changed to BCN.EXAMPLE.ESDS3.
  - VOLUMES identifies the new volume on which the cluster is to reside.
- CATALOG identifies the catalog, VCBUCAT1, to contain the cluster's catalog entry.

## Import a Cluster to a Volume Other Than One on Which It Was Originally Defined: Example 5

In this example, a key-sequenced cluster, MPS.IMPORT.CLUSTER, is imported from a portable file, CLUSBACK. The cluster is imported to a volume other than the one in which it was originally defined.

```
//IMPORT   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//PORTDS   DD   DSNAME=CLUSBACK,UNIT=3380,VOL=SER=338001,DISP=OLD
//SYSIN    DD   *
      IMPORT -
           INFILE(PORTDS) -
           OUTDATASET(MPS.IMPORT.CLUSTER) -
           OBJECTS((MPS.IMPORT.CLUSTER -
             VOLUMES(338002))) -
           CATALOG(ICFUCAT1)
/*
```

Job control language statement:

- PORTDS DD describes the portable data set, CLUSBACK.

The IMPORT command's parameters are:

- INFILE points to the PORTDS DD statement.
- OUTDATASET gives the name of the data set being imported.
- OBJECTS changes some of the attributes for the object being imported:
  - MPS.IMPORT.CLUSTER identifies the key-sequenced cluster.
  - VOLUMES identifies the new volume on which the cluster is to reside.
- CATALOG identifies the catalog, ICFUCAT1, to contain the cluster's catalog entry.

# Chapter 26. IMPORT CONNECT

The IMPORT CONNECT command connects a user catalog or a tape volume catalog to a master catalog. The syntax of the IMPORT CONNECT command is:

| | |
|---|---|
| **IMPORT** | **CONNECT**<br>**OBJECTS((***catname*<br>  **DEVICETYPE(***devtype***)**<br>  **VOLUMES(***volser***)))**<br>**[ALIAS]**<br>**[VOLCATALOG]**<br>**[CATALOG(***catname***)]** |

## IMPORT CONNECT Parameters

The IMPORT CONNECT command uses the following parameters.

## Required Parameters

**CONNECT**
specifies that a user catalog or volume catalog is to be connected to the master catalog in the receiving system. When you use CONNECT, you must also use OBJECTS to provide the user or tape volume catalog's name, DASD volser, and DASD volume device type.

**Abbreviation:** CON

**OBJECTS((***catname*

**DEVICETYPE(***devtype***)**

**VOLUMES(***volser***)))**
specifies the user or tape volume catalog to be connected.

**Abbreviation**: OBJ

*catname*
specifies the name of the user or tape volume catalog being connected.

**DEVICETYPE(***devtype***)**
specifies the device type of the volume that contains the user or tape volume catalog that is to be connected. You can specify a device type for any direct access device that is supported.

**Abbreviation:** DEVT

**VOLUMES(***volser***)**
specifies the volume containing the user or tape volume catalog.

**Abbreviation**: VOL

## Optional Parameters

**ALIAS**
specifies that alias associations for the already connected user catalog are to be retained.

**Exception:** The ALIAS parameter is ignored when the VOLCATALOG parameter is specified.

© Copyright IBM Corp. 1973, 2002

The specification of ALIAS during an IMPORT CONNECT operation is intended for cases in which the volume serial information, or device type, or both, of the user catalog has changed since the DEFINE or previous IMPORT CONNECT operation. Specifying ALIAS results in an operation that is similar to an EXPORT DISCONNECT/IMPORT CONNECT sequence, except that any aliases that are of the user catalog are preserved.

**Abbreviation:** ALS

**CATALOG(*catname*)**
specifies the name of the catalog into which to define the catalog you are connecting. This parameter is required when you want to direct the catalog's entry to a particular catalog other than the master catalog.

To specify catalog names for SMS-managed data sets, you must have authority from the RACF STGADMIN.IGG.DIRCAT facility class. See "Storage Management Subsystem (SMS) Considerations" on page 2 for more information.

*catname*
is the name of the catalog in which to define the catalog being imported. If you are import connecting a user catalog, the specified catalog is usually the master catalog. If the specified catalog is not the master catalog, the catalog must be identified by a JOBCAT or STEPCAT DD statement.

**Abbreviation:** CAT

**VOLCATALOG**
specifies that a volume catalog is to be connected.

**Abbreviation:** VOLCAT

## Import Connect Example

## Import to Connect a User Catalog

In this example, a user catalog, VCBUCAT1, is connected to the system's master catalog, AMAST1. This example reconnects the user catalog, VCBUCAT1, that was disconnected in the EXPORT DISCONNECT example.

```
//IMPORT1  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     IMPORT -
           OBJECTS -
             ((VCBUCAT1 -
             VOLUME(VSER02) -
             DEVICETYPE(3380))) -
           CONNECT -
           CATALOG(AMAST1)
/*
```

The IMPORT command builds a user catalog connector entry that identifies the user catalog VCBUCAT1 in the master catalog AMAST1. The parameters are:

- OBJECTS is required when a user catalog is being imported. The subparameters of OBJECTS identify the user catalog, VCBUCAT1; the user catalog's volume, VSER02; and the device type of the user catalog's volume, 3380.

- CONNECT specifies that the user catalog connector entry is to be built and put in the master catalog to connect the user catalog to the master catalog. CONNECT is required when a user catalog is being reconnected.
- CATALOG identifies the master catalog, AMAST1.

# Chapter 27. LISTCAT

The LISTCAT command lists catalog entries. The syntax of the LISTCAT command is:

| | |
|---|---|
| **LISTCAT** | **[ALIAS]**<br>**[ALTERNATEINDEX]**<br>**[CLUSTER]**<br>**[DATA]**<br>**[GENERATIONDATAGROUP]**<br>**[INDEX]**<br>**[LIBRARYENTRIES(***libent***)]**<br>**[NONVSAM]**<br>**[PAGESPACE]**<br>**[PATH]**<br>**[USERCATALOG]**<br>**[VOLUMEENTRIES(***volent***)]**<br>**[CREATION(***days***)]**<br>**[ENTRIES(***entryname* **[** *entryname***...])\|**<br>**LEVEL(***level***)]**<br>**[EXPIRATION(***days***)]**<br>**[FILE(***ddname***)]**<br>**[LIBRARY***(libname)***]**<br>**[NAME\|HISTORY\|VOLUME\| ALLOCATION\|ALL]**<br>**[OUTFILE(***ddname***)]**<br>**[CATALOG(***catname***)]** |

LISTCAT can be abbreviated: LISTC

## LISTCAT Parameters

The LISTCAT command uses the following parameters.

## Required Parameters

The LISTCAT command has no required parameters.

When the LISTCAT command is entered as a job step (that is, not through TSO) and no parameters are specified, an entire catalog is listed. See “Catalog Search Order for LISTCAT” on page 12 for a description of how the catalog to be listed is selected.

LISTCAT lists data sets, both SMS and non-SMS, from a catalog specified in a JOBCAT or STEPCAT statement.

Catalog management does not maintain the statistics of a catalog’s own cluster entry. While LISTCAT will record statistics for data sets defined into a catalog, most of the statistics for the catalog’s own entries are not accurately recorded.

For TSO users, when LISTCAT is invoked from a TSO terminal and no operands are specified, the prefix (the TSO userid) becomes the highest level of entryname qualification and only those entries with a matching highest level of qualification are listed. It is as though you specified:

© Copyright IBM Corp. 1973, 2002

```
LISTCAT LEVEL(TSO user prefix)
```

For RACF users, in a non-SMS environment, LISTCAT checks the authorization at the catalog level before checking the authorization at the data set level. For a SMS environment, LISTCAT checks the authorization at the data set level before checking the authorization at the catalog level.

## Optional Parameters

**[ALIAS][ALTERNATEINDEX][CLUSTER][DATA]**

**[GENERATIONDATAGROUP][INDEX][LIBRARYENTRIES]**

**[NONVSAM][PAGESPACE][PATH][USERCATALOG]**

**[VOLUMEENTRIES]**
specifies that certain types of entries are to be listed. Only those entries whose type is specified are listed. For example, if you specify CLUSTER but not DATA or INDEX, the cluster's entry is listed and its associated data and index entries are not listed.

If you use ENTRIES and also specify an entry type, the entryname is not listed unless it is of the specified type. You can specify as many entry types as desired. When you want to completely list a catalog, do not specify any entry type.

**ALIAS**
specifies that alias entries are to be listed.

**ALTERNATEINDEX**
specifies that entries for alternate indexes are to be listed. If ALTERNATEINDEX is specified and DATA and INDEX are not also specified, entries for the alternate index's data and index components are not listed.

**Abbreviation:** AIX

**CLUSTER**
specifies that cluster entries are to be listed. If CLUSTER is specified and DATA and INDEX are not also specified, entries for the cluster's data and index components are not listed.

**Abbreviation:** CL

**DATA**
specifies that entries for data components of clusters and alternate indexes are to be listed.

If a VSAM object's name is specified and DATA is coded, only the object's data component entry is listed. When DATA is the only entry type parameter coded, the catalog's data component is not listed.

**GENERATIONDATAGROUP**
specifies that entries for generation data groups are to be listed. GDSs in the active state, existing at the time the LISTCAT command is entered, are identified as such when ALL is specified.

**Abbreviation:** GDG

**INDEX**
specifies that entries for index components of key-sequenced clusters and alternate indexes are to be listed. If a VSAM object's name is specified and

INDEX is coded, only the object's index component entry is listed. When INDEX is the only entry type parameter coded, the catalog's index component is not listed.

**Abbreviation:** IX

**LIBRARYENTRIES(*libent*)**
specifies that tape library entries are to be listed.

*libent*
specifies the name of the tape library entry.

**Abbreviation:** LIBENTRIES|LIBENT

**NONVSAM**
specifies that entries for non-VSAM data sets are to be listed. If a generation data group's name and non-VSAM are specified, GDSs in the deferred, active, or rolled-off state, associated with the GDG are listed by specifying the ALL option.

**Abbreviation:** NVSAM

**PAGESPACE**
specifies that entries for page spaces are to be listed.

**Abbreviation:** PGSPC

**PATH**
specifies that entries for paths are to be listed.

**USERCATALOG**
specifies that catalog connectors are to be listed. The user catalog connector entries are in the master catalog. (User catalog connector entries can also be in a user catalog, but the operating system does not recognize them when searching for a user catalog.)

**Abbreviation:** UCAT

**VOLUMEENTRIES(*volent*)**
specifies that tape library volume entries are to be listed. Prefix the name of the tape volume with the letter 'V'. For example 'Vxxxxx', where xxxxx equals the volume name. You can specify the CATALOG parameter to list tape volume entries from a specific catalog.

*volent*
specifies the name of the tape volume entry to be listed.

**Abbreviation:** VOLENTRIES|VOLENT

**CATALOG(*catname* )**
specifies the name of the catalog that contains the entries that are to be listed. If CATALOG is coded, only entries from that catalog are listed. See "Catalog Search Order for LISTCAT" on page 12 for information about the order in which catalogs are searched.

*catname*
is the name of the catalog.

If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved.

**Abbreviation:** CAT

**CREATION(*days*)**
specifies that entries of the indicated type (CLUSTER, DATA, and so on,) are to be listed only if they were created the specified number of days ago or earlier.

*days*
specifies the number of days ago. The maximum number that can be specified is 9999; zero indicates that all entries are to be listed.

**Abbreviation:** CREAT

**ENTRIES(*entryname*[ *entryname*...]) |**

**LEVEL(*level*)**
specifies the names of entries to be listed.

Unexpired GDSs that have been rolled off and recataloged can be displayed using LISTCAT ENTRIES(*gdg.**), LISTCAT LEVEL(*gdg*) where *gdg* is the original name of the GDG, LISTCAT HISTORY, and LISTCAT ALL. Current and deferred generations are displayed as well as those that have been rolled off, in alphabetical order.

For TSO users, TSO will prefix the user ID to the specified data set name when the ENTRIES parameter is unqualified. The userid is not prefixed when the LEVEL parameter is specified.

**Exception::** You can use LISTCAT ENTRY LEVEL command only to list the cluster information of the entry. To list the data component information for the entry, you must use the LISTCAT ENTRY ALL command.

**ENTRIES(*entryname* [ *entryname*...])**
specifies the name or generic name of each entry to be listed. (See the generic examples following the description of the LEVEL parameter.) When you want to list the entries that describe a user catalog, the catalog's volume must be physically mounted. You then specify the catalog's name as the entryname.

**Abbreviation:** ENT

**LEVEL(*level*)**
specifies that all entries that match the level of qualification specified by (level) are to be listed irrespective of the number of additional qualifiers. If a generic level name is specified, only one qualifier replaces the *. The * must not be the last character specified in the LEVEL parameter. LEVEL(A.*) will give you an error message.

LEVEL can result in more than one user catalog being searched if the multilevel alias search level is greater than 1. For example if TEST is an alias for UCAT.ONE and TEST.PROD is an alias for UCAT.TWO, and the multilevel alias search level is 2, LEVEL(TEST) results in both catalogs being searched, and data sets satisfying both aliases are listed. If TEST and TEST.PROD are not defined as aliases, and there are catalogs called TEST.UCAT1 and TEST.UCAT2, LEVEL(TEST) with a multilevel alias search level of 2 results in both catalogs, as well as the master catalog, being searched for data sets with a high-level qualifier of TEST. In this situation, where a level is being searched which is not also an alias, the master catalog and all user catalogs with the same high-level qualifier *and a number of qualifiers equal to the multilevel alias search level* are searched for data sets matching the level requested. This situation should not occur if proper aliases are defined for user catalogs.

**Abbreviation:** LVL

*Examples of ENTRIES and LEVEL specifications*:

Suppose a catalog contains the following names:
1. A.A.B
2. A.B.B
3. A.B.B.C
4. A.B.B.C.C
5. A.C.C
6. A.D
7. A.E
8. A

If ENTRIES(A.*) is specified, entries 6 and 7 are listed.
If ENTRIES(A.*.B) is specified, entries 1 and 2 are listed.
If LEVEL(A.*.B) is specified, entries 1, 2, 3, and 4 are listed.
If LEVEL(A) is specified, entries 1, 2, 3, 4, 5, 6, and 7 are listed.

When using a generic name with the ENTRIES parameter, entries must have one qualifier in addition to those specified in the command.

When using the LEVEL parameter, associated entries (for example, data and index entries associated with a cluster) are not listed unless their names match the level of qualification.

If the specified cluster name is fully qualified and the data set name is the maximum length of 44 characters, more clusters than expected might be displayed. For fully qualified names use the ENTRIES parameter.

**Attention:** LISTCAT LEVEL has a restriction on the number of entries that can be displayed due to the amount of available storage below the line (24-bit addressing).

**EXPIRATION(***days***)**
specifies that entries of the indicated type (CLUSTER, DATA, and so on) are to be listed only if they will expire in the specified number of days or earlier.

*days*
specifies the number of days. The maximum number that can be specified is 9999 and it indicates that all entries are to be listed. Zero indicates that only entries that have already expired are to be listed.

**Abbreviation:** EXPIR

**FILE(***ddname***)**
specifies the name of a DD statement that identifies the devices and volumes that contain information in the VVDS that is to be listed. If FILE is not specified, the volumes are dynamically allocated. The volumes must be mounted as permanently resident or reserved.

**LIBRARY***(libname)*
specifies the name of the tape library entry for which tape volume entries are to be listed. Only those tape volumes that are entries in the specified tape library are listed.

*libname*
specifies a 1-to-8 character tape library name. You can use a partial tape library name followed by an * to list tape volume entries for more than one tape library.

**Abbreviation:** LIB

**NAME|HISTORY|VOLUME|ALLOCATION|ALL**
specifies the fields to be included for each entry listed. “Appendix B. Interpreting LISTCAT Output Listings” on page 327, shows the listed information that results when you specify nothing (which defaults to NAME), HISTORY, VOLUME, ALLOCATION, or ALL. For SMS-managed data sets and catalogs, the SMS class names and last backup date are listed in addition to the other fields specified. The class definitions are not displayed.

**Exception:** For tape library and tape volume entries, only the ALL parameter is functional. If the HISTORY, VOLUME, and ALLOCATION parameters are specified for tape library and tape volume entries, these parameters are ignored. If ALL is not specified, only the names of the tape library or tape volume entries are listed.

**NAME**
specifies that the name and entry type of the entries are to be listed. Some entry types are listed with their associated entries. The entry type and name of the associated entry follow the listed entry’s name. For details, see “ASN: Associations Group” in Appendix A.

For TSO users, only the name of each entry associated with the TSO user’s prefix is listed when no other parameters are coded.

**HISTORY**
specifies that only the following information is to be listed for each entry: name, entry type, ownerid, creation date, expiration date, and release. For GDG base and non-VSAM entries, status is also listed. For alternate indexes, “SMS-managed (YES/NO)” is also listed. For SMS-managed data sets, storage class, management class, data class, and last backup date are also listed. If the last backup date is unavailable, as in the case of migrated data sets, LISTCAT displays a field of all “X’s” instead of an actual date.

HISTORY can be specified for CLUSTER, DATA, INDEX, ALTERNATEINDEX, PATH, GENERATIONDATAGROUP, PAGESPACE, and NONVSAM. See examples, Figure 15 on page 375 and Figure 16 on page 378.

The OWNER-IDENT field in the HISTORY subset has been renamed DATASET-OWNER. This displays the contents of the data set owner field in the BCS. The ACCOUNT information is listed when the HISTORY or ALL parameter is specified.

**Abbreviation:** HIST

**VOLUME**
specifies that the information provided by specifying HISTORY, plus the volume serial numbers and device types allocated to the entries, are to be listed. Volume information is only listed for data and index component entries, non-VSAM data set entries, and user catalog connector entries.

For TSO users, only the name and volume serial numbers associated with the TSO user’s prefix are listed when no other parameters are coded.

**Abbreviation:** VOL

**ALLOCATION**
specifies that the information provided by specifying VOLUME plus detailed information about the allocation are to be listed. The information about allocation is listed only for data and index component entries.

**Abbreviation:** ALLOC

**ALL**
specifies that all fields are to be listed.

**OUTFILE(*ddname*)**
specifies a data set, other than the SYSPRINT data set, to receive the output produced by LISTCAT (that is, the listed catalog entries). Completion messages produced by access method services are sent to the SYSPRINT data set, along with your job's JCL and input statements.

ddname identifies a DD statement that describes the alternate target data set. If OUTFILE is not specified, the entries are listed in the SYSPRINT data set. If an alternate data set is specified, it must meet the requirements in "JCL DD Statement for an Alternate Target Data Set" on page 4.

**Abbreviation:** OFILE

## LISTCAT Examples

The LISTCAT command can perform the functions shown in the following examples.

## List an SMS-Managed Data Set: Example 1

In this example, the HISTORY parameter is used to list an SMS-managed data set.

```
//LISTCAT1 JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
      LISTCAT -
           ENTRIES(USER01.DATA1.EXAMPL) -
           CLUSTER -
           HISTORY
/*
```

The CLUSTER parameter specifies that only the cluster component of the entry identified in the ENTRIES parameter are listed. The HISTORY parameter causes the display of the HISTORY information along with the SMS classes and last backup date. The SMS information appears in the following format:

```
SMSDATA
  STORAGECLASS-----------SC4      MANAGEMENTCLASS-MGTCL004
  DATACLASS-----------DCL021      LBACKUP----1993.221.0255
```

If the last backup date had been unavailable, LISTCAT would have displayed:

```
LBACKUP------XXXX.XXX.XXXX
```

An example of the complete output resulting from this LISTCAT command is shown in Figure 15 on page 375.

## List a Key-Sequenced Cluster's Entry in a Catalog: Example 2

In this example, a key-sequenced cluster entry is listed.

```
//LISTCAT1 JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     LISTCAT -
           ENTRIES(LCT.EXAMPLE.KSDS1) -
           CLUSTER -
           ALL
/*
```

The LISTCAT command lists the cluster's catalog entry. It is assumed that the high level of the qualified cluster name is the same as the alias of the catalog STCUCAT1; this naming convention directs the catalog search to the appropriate catalog. The parameters are:

- ENTRIES identifies the entry to be listed.
- CLUSTER specifies that only the cluster entry is to be listed. If CLUSTER had not been specified, the cluster's data and index entries would also be listed.
- ALL specifies that all fields of the cluster entry are to be listed.

## Alter a Catalog Entry, Then List the Modified Entry: Example 3

In this example, the free space attributes for the data component (LCT.KSDATA) of cluster LCT.MYDATA are modified. Next, the cluster entry, data entry, and index entry of LCT.MYDATA are listed to determine the effect, if any, the modification has on the cluster's other attributes and specifications.

```
//LISTCAT2 JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALTER -
           LCT.KSDATA -
           FREESPACE(10 10)
     IF LASTCC = 0 -
           THEN -
           LISTCAT -
             ENTRIES(LCT.MYDATA) -
             ALL
  /*
```

The ALTER command modifies the free space specifications of the data component of the key-sequenced VSAM cluster LCT.MYDATA. Its parameters are:

- LCT.KSDATA is the *entryname* of the data component being altered. LCT.KSDATA identifies the data component of a key-sequenced VSAM cluster, LCT.MYDATA. To alter a value that applies only to the cluster's data component, such as FREESPACE does, you must specify the data component's entryname.
- FREESPACE specifies the new free space percentages for the data component's control intervals and control areas.

The IF ... THEN command sequence verifies that the ALTER command completed successfully before the LISTCAT command runs. The LISTCAT command lists the cluster's entry and its data and index entries. The parameters are:

- ENTRIES specifies the *entryname* of the object being listed. Because LCT.MYDATA is a key-sequenced cluster, the cluster entry, its data entry, and its index entry are listed.
- ALL specifies that all fields of each entry are to be listed.

## List Catalog Entries: Example 4

This example illustrates how all catalog entries with the same generic name are listed.

```
//LISTCAT3   JOB   ...
//STEP1      EXEC  PGM=IDCAMS
//SYSPRINT   DD    SYSOUT=A
//SYSIN      DD    *
     LISTCAT -
            ENTRIES(GENERIC.*.BAKER) -
            ALL
/*
```

The LISTCAT command lists each catalog entry with the generic name GENERIC.*.BAKER, where * is any 1-to-8 character simple name. The parameters are:

- ENTRIES specifies the *entryname* of the object to be listed. Because GENERIC.*.BAKER is a generic name, more than one entry can be listed.
- ALL specifies that all fields of each entry are to be listed.

## List Catalog Entries: Example 5

This example shows the LISTCAT command being used with the HISTORY parameter.

```
//LISTCAT4   JOB   ...
//STEP1      EXEC  PGM=IDCAMS
//SYSPRINT   DD    SYSOUT=A
//SYSIN      DD    *
     LISTCAT -
          ENTRIES(USER01.DATA1.EXMPL) -
          DATA -
          HISTORY
/*
```

The LISTCAT command's parameters are:

- ENTRIES specifies the name of the entry to be listed.
- DATA specifies that only the data component of the entry identified in the ENTRIES parameter are listed.
- HISTORY specifies that the HISTORY information is displayed.

## List a Tape Library Entry: Example 6

This example lists the tape library entry named ATLLIB1.

```
//LISTCLIB    JOB     ...
//STEP1       EXEC   PGM=IDCAMS
//SYSPRINT    DD     SYSOUT=A
//SYSIN       DD     *
     LISTCAT  -
          LIBRARYENTRIES(ATLLIB1) -
          ALL
/*
```

This command's parameters are:

- LIBRARYENTRIES identifies ATLLIB1 as the entry to be listed.
- ALL specifies that all information associated with the tape library entry ATLLIB1 is to be listed.

The tape library entry information is listed in the following format:

```
                         LISTING FROM CATALOG -- SYS1.VOLCAT.VGENERAL

LIBRARY-ENTRY------------ATLLIB1
     DATA-LIBRARY
     LIBRARY-ID----------12345    DEVICE-TYPE---------3495-L50    MAX-SLOTS--------------0     SCRATCH-VOLUME---------0
       CONSOLE-NAME-----CONSOLE     LOGICAL-TYPE--------AUTOMATED SLOTS-EMPTY------------0     SCR-VOL-THRESHOLD------0
                         MEDIA1     MEDIA2   MEDIA3   MEDIA4
       SCRATCH-VOLUME---------0          0        0        0
       SCR-VOL-THRESHOLD------0          0        0        0
       DESCRIPTION---(NULL)
```

## List Tape Volume Entries: Example 7

This example lists all the tape volume entries whose names begin with the letters ‘VA’ in the tape library named ATLLIB1.

```
//LISTCLIB    JOB     ...
//STEP1       EXEC   PGM=IDCAMS
//SYSPRINT    DD     SYSOUT=A
//SYSIN       DD     *
      LISTCAT -
           VOLUMEENTRIES(VA*) -
           LIBRARY(ATLLIB1) -
           ALL
/*
```

This command’s parameters are:

- VOLUMEENTRIES specifies that information relating to tape volume entries whose names begin with the letters ‘VA’ are be listed.
- LIBRARY specifies that only tape volume entries associated with the tape library named ATLLIB1 are to be listed.
- ALL requires that all information associated with the specified tape volume entries are to be listed.

The tape volume entries information is listed in the following format:

```
                         LISTING FROM CATALOG -- SYS1.VOLCAT.VGENERAL

 VOLUME-ENTRY-----------VAL0001
     DATA-VOLUME
       LIBRARY----------ATLLIB1     LOCATION-----------LIBRARY       CREATION-DATE---1991-01-01   ENT-EJ-DATE---1991-01-01
       RECORDING--------UNKNOWN     MEDIATYPE----------MEDIA2       COMPACTION------------NO     SPEC-ATTRIBUTE -----NONE
       STORAGE-GROUP---*SCRTCH*     USE-ATTRIBUTE------SCRATCH       EXPIRATION------2000-12-31   LAST-MOUNTED--1991-01-01
       CHECKPOINT-------------Y     ERROR-STATUS-------NOERROR       WRITE-PROTECTED----------N   LAST-WRITTEN--1991-01-01
       SHELF-LOCATION------------(NULL)
       OWNER---------------(NULL)
```

# Chapter 28. PRINT

The PRINT command prints VSAM data sets, non-VSAM data sets, and catalogs. The syntax of the PRINT command is:

| | |
|---|---|
| **PRINT** | **{INFILE(***ddname***)\|**<br>**INDATASET(***entryname***)}**<br>**[CHARACTER\|DUMP\|HEX]**<br>**[DBCS]**<br>**[FROMKEY(***key***)\|FROMADDRESS(***address***)\|**<br>**FROMNUMBER(***number***)\|SKIP(***number***)]**<br>**[INSERTSHIFT((***offset1 offset2***)[(***offset1 offset2*** )...])\|INSERTALL]**<br>**[OUTFILE(***ddname***)]**<br>**[SKIPDBCSCHECK((***offset1 offset2***)[(***offset1 offset2***)...])\|NODBCSCHECK]**<br>**[TOKEY(***key***)\|TOADDRESS(***address***)\|**<br>**TONUMBER(***number***)\|COUNT(***number***)]** |

**Restriction**:Access Method Services does not use RLS. If the RLS keyword is specified on the DD statement of a file to be opened by Access Method Services, the keyword is ignored and the file is opened and accessed in non-RLS mode.

## PRINT Parameters

## Required Parameters

**INFILE(***ddname***)|**

**INDATASET(***entryname***)**
identifies the data set or component to be printed. If the logical record length of a non-VSAM source data set is greater than 32,760 bytes, your PRINT command ends with an error message.

**INFILE(***ddname***)**
specifies the name of the DD statement that identifies the data set or component to be printed. You can list a base cluster in alternate-key sequence by specifying a path name as the data set name in the DD statement.

**Abbreviation:** IFILE

**INDATASET(***entryname***)**
specifies the name of the data set or component to be printed. If INDATASET is specified, the *entryname* is dynamically allocated.

You can list a base cluster in alternate-key sequence by specifying a path name as *entryname*.

If you are printing a member of a non-VSAM partitioned data set, the *entryname* must be specified in the format:

```
pdsname(membername)
```

**Abbreviation:** IDS

## Optional Parameters

The PRINT command can use the following optional parameters.

© Copyright IBM Corp. 1973, 2002

**CHARACTER|DUMP|HEX**
specifies the format of the listing.

For the CHARACTER and DUMP parameters, setting the GRAPHICS parameter of the PARM command determines which bit patterns print as characters. See “PARM Command” on page 19 for more information.

**CHARACTER**
specifies that each byte in the logical record is to be printed as a character. Bit patterns not defining a character are printed as periods. Key fields are listed in character format (see Figure 2). CHARACTER must be specified if data contains DBCS characters. DUMP and HEX cannot be specified with DBCS.

**Abbreviation:** CHAR

```
IDCAMS   SYSTEM SERVICES                        TIME:  19:26:13     03/08/88  PAGE 6
LISTING OF DATA SET -EXAMPLE.LISTC.DATA
RBA OF RECORD - 0

ABCD000000000001ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ012345
6789ABCDEFGHIJKLMNOPQRSTRUVWYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGH
IJKLMNOPQRSTUVWYZY0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRST
UVWXYZ0123456789ABCDEFGHIJKLMNOPQRST....
```

*Figure 2. Example of Character Format*

**DUMP**
specifies that each byte in the logical record is to be printed in both hexadecimal and character format. In the character portion of the listing, bit patterns not defining a character are printed as periods. Key fields are listed in hexadecimal format (see Figure 3).

```
IDCAMS   SYSTEM SERVICES                        TIME:  19:26:13     03/08/88  PAGE 2
LISTING OF DATA SET -EXAMPLE.LISTC
KEY OF RECORD - F0F0F0F0F0F0F0F0F0F0F0F1
0000   C1C2C3C4 F0F0F0F0 F0F0F0F0 F0F0F0F1  C1C2C3C4 C5C6C7C8 C9D1D2D3 D4D5D6D7   *ABCD000000000001ABCDEFGHIJKLMNOP*
0020   D8D9E2E3 D4D5D6D7 D8D9F0F1 F2F3F4F5  F6F7F8F9 C1C2C3C4 C5C6C7C8 C9D1D2D3   *QRSTRUVWYZ0123456789ABCDEFGHIJKL*
0040   D4D5D6D7 D8D9E2E3 E4E5E6E7 F8F9F0F1  F2F3F4F5 F6F7F8F9 C1C2C3C4 C5C6C7C8   *MNOPQRSTRUVWXYZ0123456789ABCDEFG*
0060   C9D1D2D3 D4D5D6D7 D8D9E2E3 E4E5E6E7  E8E9F0F1 F2F3F4F5 F6F7F8F9 C1C2C3C4   *IJKLMNOPQRSTUVWXYZ0123456789ABCD*
0080   C5C6C7C8 C9D1D2D3 D4D5D6D7 D8D9E2E3  E4E5E6E7 E8E9F0F1 F2F3F4F5 F6F7F8F9   *EFGHIJKLMNOPQRSTUVWXYZ0123456789*
00A0   C1C2C3C4 C5C6C7C8 C9D1D2D3 D4D5D6D7  D8D9E2E3 E4E5E6E7 E8E9F0F1 F2F3F4F5   *ABCDEFGHIJKLMNOPQRSTUVWXYZ012345*
00C0   F6F7F8F9 C1C2C3C4 C5C6C7C8 C9D1D2D3  D4D5D6D7 D8D9E2E3 E4E5E6E7 E8E9F0F1   *6789ABCDEFGHIJKLMNOPQRSTUVWXYZ01*
00E0   F2F3F4F5 F6F7F8F9 C1C2C3C4 C5C6C7C8  C9D1D2D3 D4D5D6D7 D8D9E2E3 E4E5E6E7   *23456789ABCDEFGHIJKLMNOPQRSTUVWX*
0100   E8E9F0F1 F2F3F4F5 F6F7F8F9 C1C2C3C4  C5C6C7C8 C9D1D2D3 D4D5D6D7 D8D9E2E3   *YZ0123456789ABCDEFGHIJKLMNOPQRST*
0120   E4E5E6E7 E8E9F0F1 F2F3F4F5 F6F7F8F9  C1C2C3C4 C5C6C7C8 C9D1D2D3 D4D5D6D7   *UVWXYZ0123456789ABCDEFGHIJKLMNOP*
0140   D8D9E2E3 E4E5E6E7 E8E9F0F1 F2F3F4F5  F6F7F8F9 C1C2C3C4 C5C6C7C8 C9D1D2D3   *QRSTRVWXYZ0123456789ABCDEFGHIJKL*
0160   D4D5D6D7 D8D9E2E3 E4E5E6E7 E8E9F0F1  F2F3F4F5 F6F7F8F9 C1C2C3C4 C5C6C7C8   *MNOPQRSTUVWXYZ0123456789ABCDEFGH*
0180   C9D1D2D3 D4D5D6D7 D8D9E2E3 00000000                                       *IJKLMNOPQRST....               *
```

*Figure 3. Example of Dump Format*

**HEX**
specifies that each byte in the logical record is to be printed as two hexadecimal digits. Key fields are listed in hexadecimal format (see Figure 4 on page 279).

```
IDCAMS   SYSTEM SERVICES                        TIME:  19:26:13     03/08/88  PAGE 4
LISTING OF DATA SET -EXAMPLE.LISTC.DATA
RBA OF RECORD - 0
C1C2C3C4F0F0F0F0F0F0F0F0F0F0F0F1C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E5E6E7E8F9F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8
C9D1D2D3D4D5D6D7D8D9E2D3D4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5
F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3
E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8
C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C7C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5
F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3
E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E300000000
```

*Figure 4. Example of Hexadecimal Format*

**DBCS**
specifies that the data to be printed includes DBCS characters. Bytes from the logical record are printed in their respective characters (that is, SBCS or DBCS format). Bit patterns not defining a character are printed as periods. When DBCS is specified, PRINT checks during printing to ensure that the DBCS data meets DBCS criteria unless SKIPDBCSCHECK or NODBCSCHECK is also specified. For more information on DBCS support, see *z/OS DFSMS: Using Data Sets*.

**FROMKEY(*key*)|FROMADDRESS(*address*)|**

**FROMNUMBER(*number*)|SKIP(*number*)**
locates the data set being listed where listing is to start. If you do not specify a value, the listing begins with the first logical record in the data set or component.

The only value that can be specified for a SAM data set is SKIP.

Use FROMADDRESS and TOADDRESS to specify a partial print range for a linear data set cluster. If required, printing is rounded up to 4096-byte boundaries.

The starting delimiter must be consistent with the ending delimiter. For example, if FROMADDRESS is specified for the starting location, use TOADDRESS to specify the ending location. The same is true for FROMKEY and TOKEY, and FROMNUMBER and TONUMBER.

**FROMKEY(*key*)**
specifies the key of the first record you want listed. You can specify generic keys (that is, a portion of the key followed by *). If you specify generic keys, listing begins at the first record with a key matching that portion of the key you specified.

You cannot specify a key longer than that defined for the data set. If you do, the listing is not done. If the specified key is not found, the next higher key is used as the starting point for the listing.

FROMKEY can be specified only when an alternate index, a key-sequenced VSAM data set, a catalog, or an indexed sequential (ISAM) non-VSAM data set is being printed.

*key*
can contain 1 to 255 EBCDIC characters. A key ending in X'5C' is processed as a generic key.

**Abbreviation:** FKEY

**FROMADDRESS(*address*)**
specifies the relative byte address (RBA) of the first record you want printed. The RBA value must be the beginning of a logical record. If you

specify this parameter for a key-sequenced data set, the listing is in physical sequential order instead of in logical sequential order.

FROMADDRESS can be specified only for VSAM key-sequenced, linear, or entry-sequenced data sets or components. FROMADDRESS cannot be specified when the data set is accessed through a path or for a key-sequenced data set with spanned records if any of those spanned records are to be accessed.

*address*
Can be specified in decimal (n) or hexadecimal (X'n'). The specification cannot be longer than one fullword when specified in decimal.

The largest address you can specify in decimal is 4,294,967,295. If a higher value is required, specify it in hexadecimal.

**Abbreviation:** FADDR

**FROMNUMBER(*number*)**
specifies the relative record number of the first record you want printed. FROMNUMBER can only be specified for VSAM relative record data sets.

*number*
Can be specified in decimal (n), hexadecimal (X'n'), or binary (B'n'). The specification cannot be longer than one fullword.

The largest address you can specify in decimal is 4,294,967,295. If a higher value is required, specify it in hexadecimal.

**Abbreviation:** FNUM

**SKIP(*number*)**
specifies the number of logical records you want to skip before the listing of records begins. For example, if you want the listing to begin with record number 500, you specify SKIP(499). SKIP should not be specified when you are accessing the data set through a path; the results are unpredictable.

**INSERTSHIFT((*offset1 offset2*)[(*offset1 offset2*)...])|INSERTALL**
If DBCS is specified without INSERTSHIFT nor INSERTALL, the logical record is assumed to already contain SO and SI characters. PRINT will check during printing to ensure that the DBCS data meets DBCS criteria.

**INSERTSHIFT((*offset1 offset2*)[(*offset1 offset2*)...])**
indicates that SO and SI characters are to be inserted in the logical record during PRINT command processing. This action has no effect on the data set referenced by PRINT. This keyword cannot be specified unless DBCS is also specified.

*offset1*
Indicates the byte offset in the logical record to be printed before which a SO character is to be inserted.

*offset2*
Indicates the byte offset in the logical record to be printed after which an SI character is to be inserted. *offset2* must be greater than *offset1* and the difference must be an even number.

Offset pairs cannot overlap ranges.

The maximum number of offset pairs that can be specified is 255.

**Abbreviation**: ISHFT

**INSERTALL**
indicates the logical record is assumed to contain only DBCS characters. An SO character is inserted at the beginning of the record and an SI character is inserted at the end of the record.

**Abbreviation**: ISALL

**OUTFILE(*ddname*)**
identifies a target data set other than SYSPRINT. For ddname, substitute the name of the JCL statement that identifies the alternate target data set.

The access method services target data set for listings, identified by the ddname SYSPRINT, is the default. The target data set must meet the requirements stated in “JCL DD Statement for a Target Data Set” on page 4.

**Abbreviation:** OFILE

**SKIPDBCSCHECK((*offset1 offset2*)[(*offset1 offset2*)...])|NODBCSCHECK**

**SKIPDBCSCHECK((*offset1 offset2*)[(*offset1 offset2*)...])**
indicates that characters between *offset1* and *offset2* are not to be checked for DBCS criteria during PRINT command processing. This keyword cannot be specified unless DBCS is also specified.

*offset1*
Indicates the byte offset in the logical record to be printed at which checking is to cease until *offset2* is reached.

*offset2*
Indicates the byte offset in the logical record after which checking is to resume. *offset2* must be greater than *offset1*.

Offset pairs cannot overlap ranges.

The maximum number of offset pairs that can be specified is 255.

**Abbreviation**: SKDCK

**NODBCSCHECK**
specifies that DBCS validity checking not be performed.

**Abbreviation**: NODCK

**TOKEY(*key*)|TOADDRESS(*address*)| TONUMBER(*number*)|COUNT(*number*)**
locates the data set being listed where you want the listing to stop. If you do not use this, the listing ends with the logical end of the data set or component.

The only value that can be specified for a SAM data set is COUNT.

Use FROMADDRESS and TOADDRESS to specify a partial print range for a linear data set cluster. The location where the listing is to stop must follow the location where the listing is to begin.

The ending delimiter must be consistent with the starting delimiter. For example, if FROMADDRESS is specified for the starting location, use TOADDRESS to specify the ending location. The same is true for FROMKEY and TOKEY, and FROMNUMBER and TONUMBER.

**TOKEY(*key*)**
specifies the key of the last record to be listed. You can specify generic keys (that is, a portion of the key followed by *). If you specify generic keys,

listing stops after the last record is listed whose key matches that portion of the key you specified. If you specify a key longer than that defined for the data set, the listing is not done.

If the specified key is not found, the next lower key is used as the stopping point for the listing.

TOKEY can be specified only when an alternate index, a key-sequenced VSAM data set, a catalog, or an indexed sequential (ISAM) non-VSAM data set is being printed.

*key*
Can contain 1 to 255 EBCDIC characters. A key ending in X'5C' is processed as a generic key.

**TOADDRESS(***address***)**
specifies the relative byte address (RBA) of the last record you want listed.

Unlike FROMADDRESS, the RBA value does not need to be the beginning of a logical record. The entire record containing the specified RBA is printed. If you specify this parameter for a key-sequenced data set, the listing is in physical sequential order instead of in logical sequential order.

TOADDRESS can be specified only for VSAM key-sequenced, linear or entry-sequenced data sets or components. TOADDRESS cannot be specified when the data set is accessed through a path. TOADDRESS cannot be specified for a key-sequenced data set with spanned records if any of those spanned records are to be accessed.

*address*
Can be specified in decimal (n) or hexadecimal (X'n').The specification cannot be longer than one fullword when specified in decimal.

The largest address you can specify in decimal is 4,294,967,295. If a higher value is required, specify it in hexadecimal.

**Abbreviation:** TADDR

**TONUMBER(***number***)**
specifies the relative record number of the last record you want printed. TONUMBER can be specified only for a VSAM relative record data set.

*number*
Can be specified in decimal (n), hexadecimal (X'n'), or binary (B'n'). The specification cannot be longer than one fullword.

The largest address you can specify in decimal is 4,294,967,295. If a higher value is required, specify it in hexadecimal.

**Abbreviation:** TNUM

**COUNT(***number***)**
specifies the number of logical records to be listed. COUNT should not be specified when you are accessing the data set through a path; the results are unpredictable.

*address or number*
can be specified in decimal (n), hexadecimal (X'n'), or binary (B'n'); the specification cannot be longer than one fullword.

The largest address you can specify in decimal is 4,294,967,295. If a higher value is required, specify it in hexadecimal.

## PRINT Examples

The PRINT command can perform the functions shown in the following examples.

## Print a Catalog: Example 1

This example shows how to print a catalog. You might find this function of the PRINT command helpful in the event of a problem with your catalog.

```
//PRINT3   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
 /* PRINT THE ENTIRE CATALOG */
    PRINT -
          INDATASET(USERCAT4)
/*
```

The PRINT command prints the entire catalog, because there are no delimiting parameters specified.

## Print a Key-Sequenced Cluster's Data Records: Example 2

In this example, the data records of a key-sequenced cluster, BRD.EXAMPLE.KSDS1, are printed in dump format. That is, each character of the record is printed in its hexadecimal and alphanumeric forms.

```
//PRINT1   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     PRINT -
           INDATASET(BRD.EXAMPLE.KSDS1)
/*
```

The PRINT command prints data records of the key-sequenced cluster, BRD.EXAMPLE.KSDS1. Its parameter INDATASET, names the data set to be printed. Because neither FROMADDRESS, FROMKEY, SKIP, TOKEY, TOADDRESS, or COUNT is specified, access method services prints all the cluster's data records. Because neither HEX nor CHAR was specified, access method services prints each record in the DUMP format. An example of the printed record is shown in Figure 5.

```
KEY OF RECORD - 00F0F0F0F0F1C9E240C4C1405CC6C9
0000 00F0F0F0 F0F1C9E2 40C4C140 5CC6C9D3 C549C8F0 C6F8F05C 40F5F040 D9C5C3D6 *.000011S DA *FILE 10380* 50 RECD*
0020 D9C4E240 D6C640F6 F940C3C8 C1D9E240 E6C9E3C8 40D2C5E8 40C9D540 D7D6E240 *RDS OF 69 CHARS WITH KEY IN POS *
0040 F160F1F1 4B000000 00000000 00000000                                     *1-11............
```

*Figure 5. Example of the Printed Record in DUMP Format*

## Copy Records from a Non-VSAM Data Set into an Entry-Sequenced VSAM Cluster, Then Print the Records: Example 3

The first 15 records from a non-VSAM data set, EXAMPLE.NONVSAM, are copied into an entry-sequenced cluster, KRL.EXAMPLE.ESDS1. If the records are copied correctly, the cluster's records are printed in hexadecimal format. If the records are not copied correctly, the non-VSAM data set's first 15 records are printed in character format.

```
//PRINT2   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//VSDSET2  DD   DSNAME=KRL.EXAMPLE.ESDS1,DISP=OLD
//SYSPRINT DD   SYSOUT=A
```

```
//SYSIN    DD   *
     REPRO -
           INDATASET(EXAMPLE.NONVSAM) -
           OUTFILE(VSDSET2) -
           COUNT(15)
     IF LASTCC = 0 -
           THEN -
           PRINT -
                 INFILE(VSDSET2) -
                 HEX
       PRINT -
           INDATASET(EXAMPLE.NONVSAM) -
           COUNT(15) -
           CHARACTER
/*
```

Job control language statement:

- VSDSET2 DD identifies the entry-sequenced VSAM cluster, KRL.EXAMPLE.ESDS1, that the records are copied into.

**Hint:** If the AMP=(BUFND=n) parameter were specified, performance would improve when the data set's records were accessed. In this example, the BUFND default is taken because only 15 records are being processed.

The REPRO command copies the first 15 records from the source data set, EXAMPLE.NONVSAM, into the target entry-sequenced cluster, KRL.EXAMPLE.ESDS1. Its parameters are:

- INDATASET identifies the source data set, EXAMPLE.NONVSAM.
- OUTFILE points to the VSDSET2 DD statement. The VSDSET2 DD statement identifies the output data set, KRL.EXAMPLE.ESDS1.
- COUNT specifies that 15 records are to be copied. Because the SKIP parameter was not specified, access method services assumes the first 15 records are to be copied. The records are always added after the last record in the output data set.

The IF ... THEN command sequence verifies that the REPRO command completed successfully before the first PRINT command runs.

The IF ... THEN command sequence ends with the HEX parameter because no continuation character follows this parameter. If you want two or more access method services commands to run only when the IF statement is satisfied, enclose the commands in a DO...END command sequence.

The first PRINT command prints the records in the entry-sequenced cluster, KRL.EXAMPLE.ESDS1. Its parameters are:

- INFILE points to the VSDSET2 DD statement. The VSDSET2 DD statement identifies the cluster, KRL.EXAMPLE.ESDS1.
- HEX specifies that each record is to be printed as a group of hexadecimal characters. An example of the printed record is shown in Figure 6.

```
RBA OF RECORD = 0
C2D6E8C5D9404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040
4040404040404040404040404040404040404040404040
```

*Figure 6. Example of the Printed Record in Hexadecimal*

The second PRINT command, which runs even if the REPRO command was unsuccessful, prints the first 15 records of the non-VSAM data set, EXAMPLE.NONVSAM. Its parameters are:

- INDATASET identifies the non-VSAM data set, EXAMPLE.NONVSAM.
- COUNT specifies that 15 records are to be printed. Because SKIP was not specified, access method services prints the first 15 records.
- CHARACTER specifies that each record is to be printed as a group of alphanumeric characters. Figure 7 shows an example of the printed record.

```
RECORD SEQUENCE NUMBER - 3
CLARK
```

*Figure 7. Example of a Printed Alphanumeric Character Record*

## Print a Linear Data Set Cluster: Example 4

A linear data set cluster is partially printed.

```
//PRINTLDS  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT  DD   SYSOUT=A
//SYSIN     DD   *
      PRINT  -
            INDATASET(EXAMPLE.LDS01) -
            FROMADDRESS(4096) -
            TOADDRESS(8191)
/*
```

The PRINT command produces a partial printout of the data set from relative byte address (RBA) 4096 up to an RBA of 8191. This is the second 4K-byte page of the linear data set. The parameters are:

- INDATASET identifies the source data set EXAMPLE.LDS01.
- FROMADDRESS specifies that printing is to start at offset 4096 in the data set.
- TOADDRESS specifies that printing is to stop at offset 8191.

## Print a Data Set that Contains DBCS Data: Example 5

Use the PRINT command to print data set USER.PRTSOSI.EXAMPLE that contains SO and SI characters surrounding DBCS data.

```
//PRINT    JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
         PRINT -
           INDATASET(USER.PRTSOSI.EXAMPLE) -
           DBCS -
           CHARACTER
/*
```

The parameters are:

- INDATASET specifies the name of the data set to be printed, USER.PRTSOSI.EXAMPLE.
- DBCS specifies that each logical record is to be printed as a group of alphanumeric characters and the logical record is assumed to already contain SO and SI characters. The bytes from the logical record are printed in their respective characters (that is, SBCS or DBCS character format).

# Chapter 29. REPRO

The REPRO command performs the following functions:

- Copies VSAM and non-VSAM data sets
- Copies catalogs
- Copies or merges tape volume catalogs
- Splits integrated catalog facility catalog entries between two catalogs
- Splits entries from an integrated catalog facility master catalog into another integrated catalog facility catalog
- Merges integrated catalog facility catalog entries into another integrated catalog facility user catalog.

**Restriction**:Access Method Services does not use RLS. If an RLS keyword is specified on the DD statement of a file to be opened by AMS, the keyword will be ignored and the file will be opened and accessed in non-RLS mode.

The syntax of the REPRO command is:

| | |
|---|---|
| **REPRO** | **{INFILE(***ddname* **[ ENVIRONMENT(DUMMY)])\|**<br>**INDATASET(***entryname* **[ ENVIRONMENT(DUMMY)])}**<br>**{OUTFILE(***ddname***)\| OUTDATASET(***entryname***)}**<br>**[DBCS]**<br>**[ENTRIES(***entryname* **[***entryname***...])\|**<br>**LEVEL(***level***)]**<br>**[ERRORLIMIT(***value***)]**<br>**[FILE(***ddname***)]**<br>**[FROMKEY(***key***)\|FROMADDRESS(***address***)\|**<br>**FROMNUMBER(***number***)\|SKIP(***number***)]**<br>**[INSERTSHIFT((***offset1 offset2***)[(***offset1 offset2* **)...])\| INSERTALL]**<br>**[MERGECAT\|NOMERGECAT]**<br>**[REPLACE\|NOREPLACE]**<br>**[REUSE\|NOREUSE]**<br>**[SKIPDBCSCHECK((***offset1 offset2***)[(***offset1 offset2* **)...])\|**<br>**NODBCSCHECK]**<br>**[TOKEY(***key***)\|TOADDRESS(***address***)\|**<br>**TONUMBER(***number***)\|COUNT(***number***)]**<br>**[VOLUMEENTRIES(***entryname***)]**<br>**[ENCIPHER**<br>**({EXTERNALKEYNAME(***keyname***)\|**<br>**INTERNALKEYNAME(***keyname***)\|PRIVATEKEY}**<br>**[CIPHERUNIT(***number***\|1)]**<br>**[DATAKEYFILE(***ddname***)\|DATAKEYVALUE(***value***)]**<br>**[SHIPKEYNAMES(***keyname***[** *keyname***...])]**<br>**[STOREDATAKEY\|NOSTOREDATAKEY]**<br>**[STOREKEYNAME(***keyname***)]**<br>**[USERDATA(***value***)])]**<br>**[DECIPHER**<br>**({DATAKEYFILE(***ddname***)\| DATAKEYVALUE(***value***)\|SYSTEMKEY}**<br>**[SYSTEMDATAKEY(***value***)]**<br>**[SYSTEMKEYNAME(***keyname***)])]** |

The parameters ENCIPHER and DECIPHER apply only to the:

© Copyright IBM Corp. 1973, 2002

- IBM Programmed Cryptographic Facility (5740-XY5)
- IBM Cryptographic Unit Support (5740-XY6)
- or the OS/390 Integrated Cryptographic Service Facility (5647-A01).

For more information, see *z/OS DFSMS: Using Data Sets* and *z/OS ICSF System Programmer's Guide* .

Partial copying of linear data set clusters is not permitted. The entire linear data space must be copied. A linear data set cluster can be copied to or from a physical sequential data set if the control interval size of the linear data set is equal to the logical record length of the physical sequential data set.

A data set with a DSORG of PO cannot be used with REPRO as either a source or target.

REPRO cannot be used as source or target tape data set if the blocksize is larger than 32760 bytes.

Attributes of data sets created by REPRO follow the same rules as those created by the JCL or by other utilities. Specifications of blocksize, logical record length, and blocking factors must be consistent with the type of data set and its physical characteristics. For more information, see the appropriate manuals: such as *z/OS MVS JCL Reference*, *z/OS MVS JCL User's Guide*, and *z/OS DFSMS: Using Data Sets*.

Because REPRO does not copy the control information stored in the directories, program libraries (in either the PDS or PDSE format) should not be copied. While the PDS as a whole cannot be copied, individual PDS members can be copied.

If you want to copy a KSDS that is larger than 4GB to a system that does not support extended addressability, you must use the FROMKEY and the TOKEY, or COUNT parameters to reduce the size of the data set or to create several smaller data sets.

REPRO copies the records of a VSAM recoverable data set. However, the locks used for VSAM record-level sharing (RLS) are not transferred. For the correct procedure to use when copying or moving data sets with pending recovery, please see "Using VSAM Record-Level Sharing" in *z/OS DFSMS: Using Data Sets* and *CICS Recovery and Restart Guide*.

## REPRO Parameters

The REPRO command uses the following parameters.

## Required Parameters

**INFILE(***ddname***[ ENVIRONMENT(DUMMY)])|**

**INDATASET(***entryname***[ ENVIRONMENT(DUMMY)] )**
Identifies the source data set to be copied. If the logical record length of a non-VSAM source data set is greater than 32760 bytes, your REPRO command will end with an error message. The keys in the source data set must be in ascending order.

**INFILE(***ddname***)**
specifies the name of the DD statement that identifies the data set to be

copied or the user catalog to be merged. You can copy a base cluster in alternate-key sequence by specifying a path name as the data set name in the DD statement.

**Abbreviation:** IFILE

**INDATASET(*entryname*)**
specifies the name of the entry to be copied or user catalog to be merged. If INDATASET is specified, the entryname is dynamically allocated. You can copy a base cluster in alternate-key sequence by specifying a path name for the entryname.

If you are copying a member of a non-VSAM partitioned data set, the entryname must be specified in the format: *pdsname(membername)*

**Abbreviation:** IDS

**ENVIRONMENT (DUMMY)**
specifies that dummy ISAM records are to be copied. Dummy records are records with hexadecimal 'FF' in the first byte.

If you do not code the ENVIRONMENT parameter, dummy records are ignored during the copy operation and are not copied. For further information, see *z/OS DFSMS: Using Data Sets*.

**Abbreviations:** ENV and DUM

**OUTFILE(*ddname*)| OUTDATASET(*entryname*)**
identifies the target data set. ISAM data sets cannot be specified as target data sets. If a VSAM data set defined with a record length greater than 32760 bytes is to be copied to a sequential data set, your REPRO command will end with an error message.

**Note:** To avoid picking up incorrect volume information from the original DD statement on a data set that has previously been deleted and redefined in this invocation of IDCAMS and the FILE parameter was specified on the delete, you must specify the OUTDATASET keyword instead of OUTFILE. Alternatively, you can issue the REPRO command in a different step from the step that invoked the delete and define commands.

**OUTFILE(*ddname*)**
specifies the name of a DD statement that identifies the target data set. For VSAM data sets, the data set name can name a path. If the DD statement identifies a SYSOUT data set, the attributes must match those specified in "JCL DD Statement for a Target Data Set" on page 4.

**Abbreviation:** OFILE

**OUTDATASET(*entryname*)**
specifies the name of the target data set. If OUTDATASET is specified, the entryname is dynamically allocated. For VSAM data sets, *entryname* can be that of a path.

**Abbreviation:** ODS

## Optional Parameters

**DBCS**
specifies that bytes in the logical record contain DBCS characters. REPRO

checks to ensure the DBCS data meets DBCS criteria. For more information about DBCS support, see *z/OS DFSMS: Using Data Sets*. This parameter cannot be specified with MERGECAT.

**ENTRIES(***entryname [ entryname***...])| LEVEL(***level***)]**
specifies the names of the entries in the source catalog to be merged when MERGECAT is specified.

For TSO users, TSO will prefix the userid to the specified data set name when ENTRIES is specified with an entry name without the userid. The userid is not prefixed when the LEVEL parameter is specified.

**ENTRIES(***entryname* **[** *entryname***...])**
specifies the name or generic name of each entry to be merged. (See the examples of generic entries following the description of the LEVEL parameter.) When using a generic name with the ENTRIES parameter, entries must have one qualifier in addition to those specified in the command.

**Note:** For information concerning RACF authorization levels, see "Appendix A. Security Authorization Levels" on page 321. RACF applies to both SMS- and non-SMS-managed data sets and catalogs.

**Abbreviation:** ENT

**LEVEL(***level***)**
specifies that all entries matching the level of qualification you indicated with the LEVEL parameter are to be merged irrespective of the number of additional qualifiers. If a generic level name is specified, only one qualifier can replace the *. The * must not be the last character specified in the LEVEL parameter. LEVEL(A.*) will give you an error message.

The LEVEL parameter is not allowed when merging tape volume catalogs. For tape volume catalogs, see "Access Method Services Tape Library Support" on page 7 and "VOLUMEENTRIES parameter" on page 296.

**Abbreviation:** LVL

*Examples of ENTRIES and LEVEL*:

Suppose a catalog contains the following names:
1. A.A.B
2. A.B.B
3. A.B.B.C
4. A.B.B.C.C
5. A.C.C
6. A.D
7. A.E
8. A

If ENTRIES(A.*) is specified, entries 6 and 7 are merged.
If ENTRIES(A.*.B) is specified, entries 1 and 2 are merged.
If LEVEL(A.*.B) is specified, entries 1, 2, 3, and 4 are merged.
If LEVEL(A) is specified, entries 1, 2, 3, 4, 5, 6, 7 are merged.

**ERRORLIMIT(***value***)**
lets you select a failure limit. Use this parameter to set a limit to the number of errors REPRO copy will tolerate. The default is four, but any number from 1 to 2,147,483,647 can be used.

**Abbreviation:** ELIMIT

**FILE(*ddname*)**
specifies the name of a DD statement that identifies all the volumes that contain the VVDSs to be updated. If you do not specify FILE, VSAM will try to dynamically allocate the required volumes.

**FROMKEY(*key*)|FROMADDRESS(*address*)|**

**FROMNUMBER(*number*)|SKIP(*number*)**
specifies the location in the source data set where copying is to start. If no value is coded, the copying begins with the first logical record in the data set. You can use only one of the four choices.

Use the SKIP parameter for a SAM data set.

None of these parameters can be specified if you are copying a catalog or a linear data set. You must copy the entire catalog or linear data set.

The starting delimiter must be consistent with the ending delimiter. For example, if FROMADDRESS is specified for the starting location, use TOADDRESS to specify the ending location. The same is true for FROMKEY and TOKEY, and FROMNUMBER and TONUMBER.

**FROMKEY(*key*)**
specifies the key of the first record you want copied. You can specify generic keys (a portion of the key followed by *). If you specify generic keys, copying begins at the first record with a key matching the specified portion of the key. If you specify a key longer than that defined for the data set, the data set is not copied. If the specified key is not found, copying begins at the next higher key.

FROMKEY can be specified only when copying an alternate index, a KSDS, or an ISAM data set.

*key*
Can contain 1-to-255 EBCDIC characters. A key ending in X'5C' is processed as a generic key.

**Abbreviation:** FKEY

**FROMADDRESS(*address*)**
specifies the relative byte address (RBA) of the first record you want copied. The RBA value must be the beginning of a logical record. If you specify this parameter for key-sequenced data, the records are copied in physical sequential order instead of in logical sequential order. FROMADDRESS:

- Can be coded only for key-sequenced or entry-sequenced data sets or components.
- Cannot be specified when the data set is being accessed through a path.
- Cannot be specified for a key-sequenced data set with spanned records if any of those spanned records are to be accessed.

*address*
Can be specified in decimal (n) or hexadecimal (X'n'). The specification cannot be longer than one fullword when specified in decimal.

The largest address you can specify in decimal is 4,294,967,295. If you require a higher value, specify it in hexadecimal.

**Abbreviation:** FADDR

**FROMNUMBER(*number*)**
specifies the relative record number of the first record you want copied. FROMNUMBER can be specified only when you copy a relative record data set.

*number*
Can be specified in decimal (n), hexadecimal (X'n'), or binary (B'n'). The specification cannot be longer than one fullword.

The largest address you can specify in decimal is 4,294,967,295. If you require a higher value, specify it in hexadecimal.

**Abbreviation:** FNUM

**SKIP(*number*)**
specifies the number of logical records to skip before beginning to copy records. For example, if you want to copy beginning with record number 500, specify SKIP(499).

SKIP should not be specified when you access the data set through a path; the results are unpredictable.

**INSERTSHIFT((*offset1 offset2*)[(*offset1 offset2*)...])|INSERTALL**
If DBCS is specified without INSERTSHIFT or INSERTALL, the logical record is assumed to already contain SO and SI characters, and REPRO will check during copying to ensure that the DBCS data meets DBCS criteria. INSERTSHIFT or INSERTALL can be specified only if DBCS is also specified, and the data set being copied is **not** a catalog.

**INSERTSHIFT[((*offset1 offset2*)[(*offset1 offset2*)...])**
indicates that SO and SI characters are to be inserted in the logical record during REPRO command processing. This action has a permanent effect on the target data set.

*offset1*
Indicates the byte offset in the logical record to be copied before which a SO character is to be inserted.

*offset2*
Indicates the byte offset in the logical record to be copied after which a SI character is to be inserted. *offset2* must be greater than *offset1* and the difference must be an odd number.

Offset pairs cannot overlap ranges.

The maximum number of offset pairs that can be specified is 255.

**Abbreviation**: ISHFT

**INSERTALL**
specifies the entire logical record is assumed to contain only DBCS characters. An SO character is inserted at the beginning of the record and an SI character is inserted at the end of the record.

**Abbreviation**: ISALL

**MERGECAT|NOMERGECAT**
specifies whether entries from the source catalog are to be merged with the entries of the target catalog. When merging catalogs ensure that data sets whose entries are merged can still be located after the merge operation. This parameter cannot be specified with the DBCS parameter.

MERGECAT merges deferred generations if specified with the GDG base during a merge of the entire catalog. Deferred generations retain their deferred status in the target catalog. Rolled-off generations are also merged during a merge of all entries.

MERGECAT can also be specified for tape volume catalogs or VOLCATS. For more information on REPRO MERGECAT of VOLCATS, refer to “Access Method Services Tape Library Support” on page 7

**MERGECAT**

specifies that the source catalog entries are to be merged with the target catalog entries and that the merged entries are to be deleted from the source catalog upon a successful merge operation.

The merge operation can be restarted if an error occurs, because the target catalog does not have to be empty. A LISTCAT and DIAGNOSE should be run before the MERGE is restarted. If the MERGE ended while processing a generation data group, it might be necessary to delete that generation data group from the target catalog because of duplicate data set names in the source and target catalogs.

Candidate volumes are preserved. MERGECAT retains candidate volume information when moving an entry from one catalog to another.

For some duplicate key errors, the merge does not end, and the processing of the next entry continues. However, some alias associations might not be merged because of the duplicate key error.

MERGECAT performs a series of DELETE NOSCRATCH and DEFINE RECATALOG requests to move entries from one catalog to another. For information concerning security authorization levels, see “Appendix A. Security Authorization Levels” on page 321.

During MERGECAT, if the target catalog name is found in the VVDS, the cluster entry for the VVDS is not recreated in the target catalog. You must use DEFINE CLUSTER RECATALOG to create the VVDS cluster entry in the target catalog.

See *z/OS DFSMS: Managing Catalogs* for additional information on the integrity of RACF discrete profiles after using MERGECAT.

**Abbreviation:** MRGC

**NOMERGECAT**

specifies that the source catalog is to be completely copied into an empty target catalog.

MERGECAT can also be specified for tape volume catalogs or VOLCATS. For more information on REPRO MERGECAT of VOLCATS, refer to “Access Method Services Tape Library Support” on page 7.

The empty target catalog implies that the copy operation cannot be restarted if an error occurs. Before the copy operation can be restarted, the target catalog must be redefined and all volumes that contain objects must be restored.

After a REPRO of one catalog to another, the VVRs are changed to point to the target catalog, and all subsequent processing must be done under the target catalog.

**Attention:** Performing REPRO on a catalog while data sets are open in the source catalog might result in a loss of information if any of those data sets extend, or other catalog updates are made. The changes might not be copied to the target catalog, resulting in a mismatch between the information contained in the VVDS and the new target BCS. This might cause the data sets to be inaccessible or receive errors.

**Abbreviation:** NOMRGC

**REPLACE|NOREPLACE**
specifies whether a record in the source cluster (INFILE or INDATASET) is to replace a record in the target cluster (OUTFILE or OUTDATASET) when the source cluster is copied into the target cluster.

When the source cluster is copied, its records might have keys or relative record numbers identical to the keys or relative record numbers of data records in the target cluster. In this case, the source record replaces the target record.

REPLACE|NOREPLACE is not used when copying integrated catalog facility catalogs because these catalogs do not use the catalog unload and reload functions.

REPLACE|NOREPLACE is not applicable for VSAM targets.

**REPLACE**
When a key-sequenced data set (other than a catalog) is copied, each source record with a key matching a target record's key replaces the target record. Otherwise, the source record is inserted into its appropriate place in the target cluster.

When a relative record data set is copied, each source record with a relative record number that identifies a data record (rather than an empty slot) in the target data set replaces the target data record. Otherwise, the source data record is inserted into the empty slot its relative record number identifies.

REPLACE cannot be used if the target data set is identified as a path through an alternate index, or if the target data set is a base cluster whose upgrade data set includes an alternate index defined with the unique-key attribute.

**Abbreviation:** REP

**NOREPLACE**
When a key-sequenced data set (other than a catalog) is copied, target records are not replaced by source records. For each source record whose key matches a target record's key, a "duplicate record" message is issued.

When a relative record data set is copied, target records are not replaced by source records. For each source record whose relative record number identifies a target data record instead of an empty slot, a "duplicate record" message is issued.

When copying something other than a VSAM data set to a sequential data set, the error limit parameter allows more than four mismatches or errors.

**Abbreviation:** NREP

**REUSE|NOREUSE**
specifies if the target data set is to be opened as a reusable data set. This parameter is valid only for VSAM data sets.

**REUSE**
specifies that the target data set, specified with OUTFILE or OUTDATASET, is opened as a reusable data set whether or not it was defined as reusable with the REUSE parameter. (See the DEFINE CLUSTER command description.) If the data set was defined with REUSE, its high-used relative byte address (RBA) is reset to zero (that is, the data set is effectively empty) and the operation proceeds. When you open a reusable data set with the reset option, that data set cannot be shared with other jobs.

If REUSE is specified and the data set was originally defined with the NOREUSE option, the data set must be empty; if not, the REPRO command ends with an error message.

**Abbreviation:** RUS

**NOREUSE**
specifies that records are written at the end of an entry-sequenced data set. (OUTFILE or OUTDATASET must identify a nonempty data set.)

**Abbreviation:** NRUS

**SKIPDBCSCHECK((***offset1 offset2***)[(***offset1 offset2***)...])|NODBCSCHECK**
SKIPDBCSCHECK and NODBCSCHECK cannot be specified unless DBCS is also specified.

**SKIPDBCSCHECK((***offset1 offset2***)[(***offset1 offset2***)...])**
indicates that characters between *offset1* and *offset2* are not to be checked for DBCS criteria during REPRO command processing.

*offset1*
Indicates the byte offset in the logical record to be copied at which checking is to cease until *offset2* is reached.

*offset2*
Indicates the byte offset in the logical record after which checking is to resume. *offset2* must be greater than *offset1*.

Offset pairs cannot overlap ranges.

The maximum number of offset pairs that can be specified is 255.

**Abbreviation**: SKDCK

**NODBCSCHECK**
indicates DBCS verification checking will not be done.

**Abbreviation**: NODCK

**TOKEY(***key***)|TOADDRESS(***address***)|TONUMBER(***number***)|COUNT(***number***)**
specifies where copying is to end in the data set being copied. You can specify only one of these parameters for a copy operation. The location where copying is to end must follow the location where it is to begin. If no value is coded, copying ends at the logical end of the data set or component.

COUNT is the only parameter that can be specified for a SAM data set.

None of these parameters can be specified when copying a catalog or a linear data set; the entire catalog or linear data space must be copied.

The ending delimiter must be consistent with the starting delimiter. For example, if FROMADDRESS is specified for the starting location, use TOADDRESS to specify the ending location. The same is true for FROMKEY and TOKEY, and FROMNUMBER and TONUMBER.

**TOKEY(*key*)**
specifies the key of the last record you want copied. You can specify generic keys (a portion of the key followed by *). If you specify generic keys, copying stops after the last record whose key matches that portion of the key you specified is copied . If you specify a key longer than the one defined for the data set, the data set is not copied. If the specified key is not found, copying ends at the next lower key. TOKEY can be specified only when copying an alternate index, a KSDS, or an ISAM data set. A key ending in X'5C' is processed as a generic key.

*key*
Can contain 1-to-255 EBCDIC characters.

**TOADDRESS(*address*)**
specifies the relative byte address (RBA) of the last record you want copied. Unlike FROMADDRESS, the RBA value does not need to be the beginning of a logical record. The entire record containing the specified RBA is copied.

If you specify this parameter for a KSDS, the records are copied in physical sequential order instead of in logical sequential order.

Use TOADDRESS with VSAM key-sequenced or entry-sequenced data sets or components. TOADDRESS cannot be specified when the data set is accessed through a path. TOADDRESS cannot be specified for a key-sequenced data set with spanned records if any of those spanned records are to be accessed.

*address*
can be specified in decimal (n) or hexadecimal (X'n'). The specification cannot be longer than one fullword.

The largest address you can specify in decimal is 4,294,967,295. If you require a higher value, specify it in hexadecimal.

**Abbreviation:** TADDR

**TONUMBER(*number*)**
specifies the relative record number of the last record you want copied. TONUMBER can be specified only when you copy a relative record data set.

*number*
Can be specified in decimal (n), hexadecimal (X'n'), or binary (B'n'). The specification cannot be longer than one fullword.

The largest address you can specify in decimal is 4,294,967,295. If you require a higher value, specify it in hexadecimal.

**Abbreviation:** TNUM

**COUNT(*number*)**
specifies the number of logical records you want copied. COUNT should not be specified when you access the data set through a path; results are unpredictable.

**VOLUMEENTRIES(*entryname*)**
specifies the tape volume catalogs to be merged or copied. The LEVEL parameter is not allowed when merging tape volume catalogs. When a tape volume catalog is copied, REPRO verifies that the target is a tape volume catalog.

**Abbreviation:** VOLENTRIES|VOLENT

## Cryptographic Parameters

**ENCIPHER**
specifies that the source data set is to be enciphered as it is copied to the target data set.

**Abbreviation:** ENCPHR

**EXTERNALKEYNAME(*keyname*) |INTERNALKEYNAME(*keyname*) |PRIVATEKEY**
specifies whether you, the Programmed Cryptographic Facility, the Cryptographic Unit Support, or the OS/390 Integrated Cryptographic Service Facility manage keys privately.

**EXTERNALKEYNAME(*keyname*)**
specifies that the Programmed Cryptographic Facility, or the Cryptographic Unit Support, or the OS/390 Integrated Cryptographic Service Facility manages keys. This parameter also supplies the 1-to-8 character key name of the external file key that is used to encipher the data encrypting key. The key is known only by the deciphering system. The key name and its corresponding enciphered data encrypting key are listed in SYSPRINT only if NOSTOREDATAKEY is specified.

**Abbreviation:** EKN

**INTERNALKEYNAME(*keyname*)**
specifies that the Programmed Cryptographic Facility, or the Cryptographic Unit Support, or the OS/390 Integrated Cryptographic Service Facility manages keys. This parameter also supplies the 1-to-8 character key name of the internal file key that is used to encipher the data encrypting key. The key is retained by the key-creating system. The key name and its corresponding enciphered data encrypting key will only be listed in SYSPRINT if NOSTOREDATAKEY is specified.

**Abbreviation:** IKN

**PRIVATEKEY**
specifies that the key is to be managed by you.

**Abbreviation:** PRIKEY

**CIPHERUNIT(*number* |1)**
specifies that multiple logical source records are to be enciphered as a unit. *Number* specifies the number of records that are to be enciphered together. By specifying that multiple records are to be enciphered together, you can improve your security (chaining is done across logical record boundaries) and also improve your performance. However, there is a corresponding increase in virtual storage requirements. The remaining records in the data set, after the last complete group of multiple records, are enciphered as a group. (If *number* is 5 and there are 22 records in that data set, the last 2 records are enciphered as a unit.)

The value for *number* can range from 1 to 255.

**Abbreviation:** CPHRUN

**DATAKEYFILE(*ddname*)|DATAKEYVALUE(*value*)**
specifies that you are supplying a plaintext (not enciphered) data encrypting key. If one of these parameters is not specified, REPRO will generate the data encrypting key. These parameters are valid only when EXTERNALKEYNAME or PRIVATEKEY is specified. If INTERNALKEYNAME and DATAKEYVALUE or

DATAKEYFILE are specified, REPRO will generate the data encrypting key and DATAKEYVALUE or DATAKEYFILE are ignored by REPRO.

The plaintext data encrypting key will not be listed in SYSPRINT unless PRIVATEKEY is specified and REPRO provides the key.

**DATAKEYFILE(*ddname*)**
identifies a data set that contains the plaintext data encrypting key. For ddname, substitute the name of the JCL statement that identifies the data encrypting key data set.

**Abbreviation:** DKFILE

**DATAKEYVALUE(*value*)**
specifies the 8-byte value to be used as the plaintext data encrypting key to encipher the data.

*Value* can contain 1-to-8 EBCDIC characters or 1-to-16 hexadecimal characters coded X'n'. *Value* must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark must be coded as two single quotation marks. With either EBCDIC or hexadecimal representation, *value* is padded on the right with blanks (X'40') if it is fewer than 8 characters.

**Abbreviation:** DKV

**SHIPKEYNAMES(*keyname*[ *keyname*...])**
supplies the 1-to-8 character key name of one or more external file keys to be used to encipher the data encrypting key. Each key name and its corresponding enciphered data encrypting key is listed in SYSPRINT, but is not stored in the target data set header. The primary use for this parameter is to establish multiple enciphered data encrypting keys to be transmitted to other locations for use in deciphering the target enciphered data set. This parameter is valid only when INTERNALKEYNAME or EXTERNALKEYNAME is specified.

**Abbreviation:** SHIPKN

**STOREDATAKEY|NOSTOREDATAKEY**
specifies whether the enciphered data encrypting key is to be stored in the target data set header. The key used to encipher the data encrypting key is identified by INTERNALKEYNAME or EXTERNALKEYNAME. This parameter is valid only when INTERNALKEYNAME or EXTERNALKEYNAME is specified. If the enciphered data encrypting key is stored in the data set header, it does not have to be supplied by the user when the data is deciphered.

**Restriction:** A data encrypting key enciphered under the keys identified by SHIPKEYNAMES cannot be stored in the header. Therefore, you might want to avoid using STOREDATAKEY and SHIPKEYNAMES together because this could result in storing header information unusable at some locations.

**STOREDATAKEY**
specifies that the enciphered data encrypting key is to be stored in the target data set header.

**Abbreviation:** STRDK

**NOSTOREDATAKEY**
specifies that the enciphered data encrypting key is not to be stored in the target data set header. The keyname and its corresponding enciphered data encrypting key is listed in SYSPRINT.

**Abbreviation:** NSTRDK

**STOREKEYNAME(*keyname*)**
specifies whether to store a keyname for the key used to encipher the data encrypting key in the target data set header. The specified keyname must be the name the key is known by on the system where the REPRO DECIPHER is to be performed. This keyname must be the same one specified in INTERNALKEYNAME if REPRO DECIPHER is to be run on the same system. If REPRO DECIPHER is run on a different system, the specified keyname can be different from the one specified in INTERNALKEYNAME or EXTERNALKEYNAME.

This parameter is valid only when INTERNALKEYNAME or EXTERNALKEYNAME is specified. If the keyname is stored in the data set header, it does not have to be supplied by the user when the data is deciphered.

**Restriction:** Keyname values identified by the SHIPKEYNAMES parameter cannot be stored in the header. Therefore, you might want to avoid using STOREKEYNAME and SHIPKEYNAMES together because this could result in storing header information unusable at some locations.

**Abbreviation:** STRKN

**USERDATA(*value*)**
specifies 1-to-32 characters of user data to be placed in the target data set header. For example, this information can be used to identify the security classification of the data.

*Value* can contain 1-to-32 EBCDIC characters. If *value* contains a special character, enclose the *value* in single quotation marks (for example, USERDATA('*CONFIDENTIAL*')). If the *value* contains a single quotation mark, code the embedded quotation mark as two single quotation marks (for example, USERDATA('COMPANY''S')).

You can code *value* in hexadecimal form, where two hexadecimal characters represent one EBCDIC character. For example, USERDATA(X'C3D6D4D7C1D5E8') is the same as USERDATA(COMPANY). The string can contain up to 64 hexadecimal characters when expressed in this form, resulting in up to 32 bytes of information.

**Abbreviation:** UDATA

**DECIPHER**
specifies that the source data set is to be deciphered as it is copied to the target data set. The information from the source data set header is used to verify the plaintext deciphered data encrypting key supplied, or deciphered from the information supplied, as the correct plaintext data encrypting key for the decipher operation.

**Abbreviation:** DECPHR

**DATAKEYFILE(*ddname*) | DATAKEYVALUE(*value*) | SYSTEMKEY**
specifies whether you, the Programmed Cryptographic Facility, Cryptographic Unit Support, or the OS/390 Integrated Cryptographic Service Facility manages keys privately.

**DATAKEYFILE(*ddname*)**
specifies that the key is to be managed by you, and identifies a data set that contains the private data encrypting key that was used to encipher the

data. For *ddname*, substitute the name of the JCL statement that identifies the data set containing the private data encrypting key.

**Abbreviation:** DKFILE

**DATAKEYVALUE(*value*)**
specifies that the key is to be managed by you, and supplies the 1- to 8-byte value that was used as the plaintext private data encrypting key to encipher the data.

*Value* can contain 1-to-8 EBCDIC characters, and must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark contained within *value* must be coded as two single quotation marks. You can code *value* in hexadecimal form, (X'n'), *value* can contain 1-to-16 hexadecimal characters, resulting in 1 to 8 bytes of information. With either EBCDIC or hexadecimal representation, *value* is padded on the right with blanks (X'40') if it is less than 8 characters.

**Abbreviation:** DKV

**SYSTEMKEY**
specifies that the Programmed Cryptographic Facility, the Cryptographic Unit Support, or the OS/390 Integrated Cryptographic Service Facility manages keys.

**Abbreviation:** SYSKEY

**SYSTEMDATAKEY(*value*)**
specifies the 1- to 8-byte value representing the enciphered system data encrypting key used to encipher the data. This parameter is valid only if SYSTEMKEY is specified. If SYSTEMDATAKEY is not specified, REPRO obtains the enciphered system data encrypting key from the source data set header. In this case, STOREDATAKEY must have been specified when the data set was enciphered.

*value* can contain 1-to-8 EBCDIC characters and must be enclosed in single quotation marks if it contains commas, semicolons, blanks, parentheses, or slashes. A single quotation mark must be coded as two single quotation marks. You can code value in hexadecimal form, (X'n'). *value* can contain 1-to-16 hexadecimal characters, resulting in 1-to-8 bytes of information. With either EBCDIC or hexadecimal representation, *value* is padded on the right with blanks (X'40') if it is fewer than 8 characters.

**Abbreviation:** SYSDK

**SYSTEMKEYNAME(*keyname*)**
specifies the 1-to-8 character key name of the internal key that was used to encipher the data encrypting key. This parameter is only valid if SYSTEMKEY is specified. If SYSTEMKEYNAME is not specified, REPRO obtains the key name of the internal key from the source data set header. In this case, STOREKEYNAME must have been specified when the data set was enciphered.

**Abbreviation:** SYSKN

## REPRO Examples

The REPRO command can perform the functions shown in the following examples.

## Copy Records into a Key-Sequenced Data Set: Example 1

In this example, records from an indexed sequential data set, ISAMDSET, are copied into key-sequenced VSAM cluster, RPR.EXAMPLE.KSDS1.

```
//REPRO1   JOB    ...
//STEP1    EXEC PGM=IDCAMS
//INDSET1  DD   DSNAME=ISAMDSET,DISP=OLD,
//         DCB=(DSORG=IS,BUFNO=6)
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
           INFILE(INDSET1) -
           OUTDATASET(RPR.EXAMPLE.KSDS1)
/*
```

Job control language statement:

- INDSET1 DD describes the indexed sequential data set, ISAMDSET, that contains the source records. The BUFNO parameter specifies the number of buffers assigned to the ISAM data set. This improves performance when the ISAM data set's records are accessed.

The REPRO command copies all records from the source data set, ISAMDSET, to the target data set, RPR.EXAMPLE.KSDS1. Its parameters are:

- INFILE points to the INDSET1 DD statement, which identifies the source data set: ISAMDSET.
- OUTDATASET identifies the key-sequenced VSAM cluster into which the source records are to be copied. This data set is dynamically allocated by access method services.

## Copy Records into a VSAM Data Set: Example 2

In this two-part example, data records are copied from the non-VSAM data set SEQ.DRGV, a sequential data set, into a key-sequenced VSAM data set, RPR.MYDATA. Next, records are copied from the key-sequenced data set, RPR.MYDATA, into an entry-sequenced data set, ENTRY.

```
//REPRO2   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//INPUT    DD   DSNAME=SEQ.DRGV,DISP=SHR,DCB=(BUFNO=6)
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
           INFILE(INPUT) -
           OUTDATASET(RPR.MYDATA)-
           ERRORLIMIT(6)
/*
//STEP2    EXEC PGM=IDCAMS
//INPUT    DD   DSNAME=RPR.MYDATA,DISP=OLD
//OUTPUT   DD   DSNAME=ENTRY,DISP=OLD
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
           INFILE(INPUT) -
           OUTFILE(OUTPUT) -
           FROMKEY(DEAN) -
           TOKEY(JOHNSON)
/*
```

### STEP1

Access method services copies records from a sequential data set, SEQ.DRGV, into a key-sequenced data set, RPR.MYDATA. STEP1's job control language statement:

- INPUT DD identifies the sequential data set, SEQ.DRGV, that contains the source records. The BUFNO parameter specifies the number of buffers assigned to the sequential data set. This improves performance when the data set's records are accessed.

STEP1's REPRO command copies all records from the source data set, SEQ.DRGV, to the target data set, RPR.MYDATA. Its parameters are:

- INFILE points to the INPUT DD statement, which identifies the source data set.
- OUTDATASET identifies the key-sequenced data set into which the source records are to be copied. The data set is dynamically allocated by access method services.
- ERRORLIMIT identifies the number of errors REPRO will tolerate.

### STEP2

Access method services copies some of the records of the key-sequenced data set RPR.MYDATA into an entry-sequenced data set, ENTRY. STEP2's job control language statements:

- INPUT DD identifies the key-sequenced cluster, RPR.MYDATA, that contains the source records.
- OUTPUT DD identifies the entry-sequenced cluster, ENTRY, that the records are to be copied into.

STEP2's REPRO command copies records from the source data set, RPR.MYDATA, to the target data set, ENTRY. Only those records with key values from DEAN to, and including, JOHNSON are copied.

The parameters are:

- INFILE points to the INPUT DD statement, which identifies the source key-sequenced data set.
- OUTFILE points to the OUTPUT DD statement, which identifies the entry-sequenced data set into which the source records are to be copied.
- FROMKEY and TOKEY specify the lower and upper key boundaries.

If ENTRY already contains records, VSAM merges the copied records with ENTRY's records. A subsequent job step could resume copying the records into ENTRY, beginning with the records with key greater than JOHNSON. If you subsequently copied records with key values less than DEAN into ENTRY, VSAM merges them with ENTRY's records.

## Merge an Integrated Catalog Facility User Catalog into Another Integrated Catalog Facility User Catalog: Example 3

This example shows how integrated catalog facility user catalog entries are merged into another integrated catalog facility user catalog. This function effectively combines entries from two catalogs into one catalog.

```
//MERGE6   JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//DD1      DD    VOL=SER=VSER01,UNIT=DISK,DISP=OLD
//         DD    VOL=SER=VSER02,UNIT=DISK,DISP=OLD
//         DD    VOL=SER=VSER03,UNIT=DISK,DISP=OLD
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     REPRO -
           INDATASET(USERCAT4) -
```

```
          OUTDATASET(USERCAT5) -
          MERGECAT -
          FILE(DD1)
/*
```

The REPRO command moves all the entries from the source catalog, USERCAT4, and merges them into the target catalog, USERCAT5. All the entries moved are no longer accessible in the source catalog.

- INDATASET identifies the source catalog, USERCAT4.
- OUTDATASET identifies the target catalog, USERCAT5.
- MERGECAT specifies that entries from the source catalog are to be merged with entries of the target catalog.
- FILE specifies the ddname of a DD statement that describes all the volumes that contain VVDS entries for all the entries that are being merged.

## Merge Selected Entries (Split) from a User Catalog into Another User Catalog: Example 4

This example shows how selected entries from an integrated catalog facility user catalog are merged into another integrated catalog facility user catalog that is empty. This function effectively splits a catalog into two catalogs. However, the MERGECAT parameter allows the target catalog to be empty or nonempty.

```
//MERGE76   JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//DD1      DD    VOL=SER=VSER01,UNIT=DISK,DISP=OLD
//         DD    VOL=SER=VSER02,UNIT=DISK,DISP=OLD
//         DD    VOL=SER=VSER03,UNIT=DISK,DISP=OLD
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     REPRO -
           INDATASET(USERCAT4) -
           OUTDATASET(USERCAT5) -
           ENTRIES(VSAMDATA.*) -
           MERGECAT -
           FILE(DD1)
/*
```

The REPRO command moves selected entries from the source catalog, USERCAT4, and merges them into the empty target catalog, USERCAT5. All the entries moved are no longer accessible in the source catalog.

- INDATASET identifies the source catalog, USERCAT4.
- OUTDATASET identifies the target catalog, USERCAT5
- ENTRIES specifies a generic name, VSAMDATA.*. All the names of the entries cataloged in the source catalog that satisfy the generic name are selected to be merged.
- MERGECAT specifies that entries from the source catalog are to be merged with entries of the target catalog.
- FILE specifies the ddname of a DD statement that describes all the volumes that contain VVDS entries for all the entries that are being merged.

## Copy a Catalog: Example 5

In this example, a catalog is copied to illustrate the catalog copying procedure.

```
//COPYCAT  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
```

```
        DEFINE USERCATALOG -
           (NAME(COPYUCAT) -
           ICFCATALOG -
           FOR(365) -
           CYLINDERS(20 10) -
           VOLUME(338000) )
/*
//STEP2    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     REPRO NOMERGECAT -
           INDATASET(MYCAT) -
           OUTDATASET(COPYUCAT)
     EXPORT -
           MYCAT -
           DISCONNECT
/*
//STEP3    EXEC  PGM=IDCAMS
//STEPCAT  DD    DSNAME=COPYUCAT,DISP=OLD
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     LISTCAT NAMES
/*
//STEP4    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DEFINE ALIAS -
           (NAME(MYCAT) -
           RELATE(COPYUCAT) )
/*
```

## STEP 1

A user catalog, COPYUCAT, is defined on volume 338000 using the DEFINE USERCATALOG command. Its parameters are:

- NAME specifies the name of the new catalog, COPYUCAT.
- ICFCATALOG specifies the catalog format of COPYUCAT.
- FOR specifies that the catalog is to be retained for 365 days.
- CYLINDERS specifies that the catalog itself is initially to occupy 20 cylinders. When the catalog's data component is extended, it is to be extended in increments of 10 cylinders.
- VOLUME specifies that the catalog is to reside on volume 338000.

## STEP 2

The REPRO NOMERGECAT command copies the contents of MYCAT into COPYUCAT. Access method services treats each catalog as a key-sequenced data set and copies each record. The first three records of MYCAT, which describe MYCAT as an integrated catalog facility catalog, are not copied into COPYUCAT. Entries from MYCAT are written into COPYUCAT beginning with record 4 (that is, after the three self-describing records of COPYUCAT). The REPRO command's parameters are:

- INDATASET identifies the source data set, MYCAT. MYCAT is cataloged in the master catalog.
- OUTDATASET identifies the receiving data set, COPYUCAT. COPYUCAT is cataloged in the master catalog.

The EXPORT command removes MYCAT's user catalog connector entry from the master catalog. MYCAT's cataloged objects now are not available to the system. (STEP4 builds an alias entry that relates MYCAT to COPYUCAT, making the cataloged objects available to the system again.)

### STEP 3

The LISTCAT command lists the name of each entry in the new catalog, COPYUCAT. The STEPCAT DD statement identifies the catalog to be listed.

LISTCAT cannot run in a job step where the catalog is empty when it is opened. To ensure that the LISTCAT correctly reflects the contents of the catalog, the LISTCAT was run as a separate job step.

### STEP 4

Access method services builds an alias entry that relates MYCAT entries to COPYUCAT.

## Copy a DBCS Data Set: Example 6

In this example, the REPRO command is used with the DBCS and INSERTSHIFT parameters. The REPRO command copies the input data set to the output data set inserting SO and SI characters into each logical record of the output data set. It is assumed that the input data set's logical records contain DBCS characters and have an LRECL, for this example, of 100 bytes and the record format is fixed length records.

```
//REPRO    JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//OUTDS    DD   DSN=MY.DATA,DISP=(NEW,CATLG),VOL=SER=VSER01,
//            UNIT=3380,DCB=(LRECL=104,RECFM=F),SPACE=(TRK,(20,10))
//SYSIN    DD   *
         REPRO -
           INDATASET(USER.REPRO.EXAMPLE) -
           OUTFILE(OUTDS) -
           DBCS -
           INSERTSHIFT((11 30)(51 60))
/*
```

The parameters are:

- INDATASET specifies the name of USER.REPRO.EXAMPLE the data set to be copied. This data set might not contain SO and SI characters.
- OUTFILE specifies the name of the output data set, MY.DATA. This data set will have SO and SI characters inserted. Because four shift characters are being inserted, the LRECL must be 4 bytes larger than the input data set's LRECL.
- DBCS specifies that the data contains DBCS characters and should be criteria checked.
- INSERTSHIFT specifies that a SO character is inserted before offsets 11 and 51 of the logical record and a SI character is inserted after offsets 30 and 60 of the logical record.

## Encipher Using System Keys: Example 7

In this example, an enciphered copy of part of a VSAM relative record data set is produced using a tape as output. The enciphered data set is deciphered at a remote installation. The keys are managed by the Programmed Cryptographic Facility, the Cryptographic Unit Support, or the OS/390 Integrated Cryptographic Service Facility.

```
//ENSYS    JOB    ...
//STEP1    EXEC PGM=IDCAMS
//CLEAR    DD   DSN=RRDS1,DISP=SHR
//CRYPT    DD   DSN=RRDSEN,LABEL=(1,SL),DISP=NEW,
//              UNIT=3480,VOL=SER=TAPE01,
//              DCB=(DEN=3,RECFM=FB,LRECL=516,BLKSIZE=5160)
//SYSPRINT DD   SYSOUT=A
```

```
//SYSIN    DD   *
     REPRO -
           INFILE(CLEAR) -
           OUTFILE(CRYPT) -
           COUNT(50) -
           ENCIPHER -
             (EXTERNALKEYNAME(AKEY27) -
              STOREDATAKEY -
              CIPHERUNIT(4) -
              USERDATA(CONF))
/*
```

Job control language statements:

- CLEAR DD describes the relative record data set.
- CRYPT DD describes and allocates a magnetic tape file. LRECL is the relative record data set record size plus 4.

The REPRO command copies 50 records enciphered from a generated data encrypting key, from the source data set, RRDS1, to the output tape. The source records are enciphered in units of 4 records, except for the last 2 records, which are enciphered together. The enciphered data encrypting key is stored in the header of the target data set; therefore, REPRO will not list the key name or enciphered data encrypting key in SYSPRINT. The parameters of the command are:

- INFILE points to the CLEAR DD statement identifying the source data set to be enciphered, RRDS1.
- OUTFILE points to the CRYPT DD statement, identifying the target data set on tape.
- COUNT indicates that 50 records are to be copied.
- ENCIPHER indicates that the target data set is to contain an enciphered copy of the source data set.
- EXTERNALKEYNAME supplies the name, AKEY27, of the external file key to be used to encipher the data encrypting key.
- STOREDATAKEY indicates that the data encrypting key enciphered under the secondary file key is to be stored in the header of the target data set.
- CIPHERUNIT indicates that 4 source records at a time are to be enciphered as a unit.
- USERDATA specifies a character string, CONF, to be stored in the header of the target data set as user data.

## Decipher Using System Keys: Example 8

In this example, the enciphered data set produced by the job in Encipher Using System Keys: Example 7 is deciphered, using a VSAM relative record data set as the target for the plaintext (deciphered) data. The empty slots in the original data set are reestablished. Keys are managed by the Programmed Cryptographic Facility or the Cryptographic Support Unit.

```
//DESYS    JOB    ...
//STEP2    EXEC PGM=IDCAMS
//CRYPT    DD   DSN=RRDSEN,LABEL=(1,SL),DISP=OLD,
//              UNIT=3480,VOL=SER=TAPE01,
//              DCB=DEN=3
//CLEAR    DD   DSN=RRDS2,DISP=SHR
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
           INFILE(CRYPT) -
           OUTFILE(CLEAR) -
```

```
          DECIPHER -
            (SYSTEMKEY -
            SYSTEMKEYNAME(BKEY27))
/*
```

Job control language statements:

- CRYPT DD describes and allocates the magnetic tape containing the enciphered data.
- CLEAR DD describes the relative record data set.

The REPRO command copies and deciphers the enciphered data set from the source tape to the target data set RRDS2. The enciphered data encrypting key is obtained from the header of the source data set. Use the internal file key (BKEY27) to decipher the enciphered data encrypting key that is then used to decipher the data. The parameters of the REPRO command are:

- INFILE points to the CRYPT DD statement, identifying the tape containing the enciphered source data.
- OUTFILE points to the CLEAR DD statement, identifying the data set to contain the deciphered data, RRDS2. The defined record size must be the same as that of the original relative record data set.
- DECIPHER indicates that the source data set is to be deciphered as it is copied to the target data set.
- SYSTEMKEY indicates that keys are managed by the Program Cryptographic Facility, the Cryptographic Unit Support, or the OS/390 Integrated Cryptographic Service Facility.
- SYSTEMKEYNAME supplies the key name, BKEY27, of the internal file key that was used to encipher the system data encrypting key. The file key must be an internal file key in this system.

## Encipher Using Private Keys: Example 9

In this example, an enciphered copy of a SAM data set is produced by using an entry-sequenced data set as the target data set. The enciphered data set resides on a volume that is to be stored offline at the local installation. Each record in the target data set is enciphered separately, using a data encrypting key supplied by the user with a data encrypting key data set. Keys are managed privately by the user.

```
//ENPRI    JOB    ...
//STEP1    EXEC PGM=IDCAMS
//CLEAR    DD   DSN=SAMDS1,DISP=OLD,
//              VOL=SER=VOL005,UNIT=DISK
//CRYPT    DD   DSN=ESDS1,DISP=OLD
//KEYIN    DD   *
     X'53467568503A7C29'
/*
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
           INFILE(CLEAR) -
           OUTFILE(CRYPT) -
           REUSE -
           ENCIPHER -
             (PRIVATEKEY -
             DATAKEYFILE(KEYIN))
  /*
```

Job control language statements:

- CLEAR DD describes the SAM data set.

- CRYPT DD describes the entry-sequenced data set.
- KEYIN DD describes the data encrypting key data set consisting of a single record containing the data encrypting key.

The REPRO command copies all records enciphered under the supplied data encrypting key, from the source data set, SAMDS1, to the target data set, ESDS1. The plaintext private data encrypting keys is not listed on SYSPRINT, because the user manages the key. The parameters of the REPRO command are:

- INFILE points to the CLEAR DD statement, identifying the source data set to be enciphered, SAMDS1.
- OUTFILE points to the CRYPT DD statement, identifying the target data set, ESDS1. The defined maximum record size of the entry-sequenced data set must be large enough to accommodate the largest SAM record.
- REUSE indicates that the target data set is to be opened as a reusable data set. If the data set was defined as REUSE, it is reset to empty; otherwise, the REPRO command will end.
- ENCIPHER indicates that the target data set is to contain an enciphered copy of the source data set.
- PRIVATEKEY indicates that the key is to be managed by the user.
- DATAKEYFILE points to the KEYIN DD statement that supplies the plaintext data encrypting key, X'53467568503A7C29', to be used to encipher the data.

## Decipher Using Private Keys: Example 10

In this example, the enciphered data set produced by the job in Encipher Using Private Keys: Example 9 is deciphered at the same location, using an entry-sequenced data set as the target for the plaintext (deciphered) data. Keys are managed privately by the user.

```
//DEPRI    JOB    ...
//STEP1    EXEC PGM=IDCAMS
//CRYPT    DD   DSN=ESDS1,DISP=OLD
//CLEAR    DD   DSN=ESDS3,DISP=OLD
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
           INFILE(CRYPT) -
           OUTFILE(CLEAR) -
           DECIPHER -
             (DATAKEYVALUE(X'53467568503A7C29'))
/*
```

Job control language statements:

- CRYPT DD describes the enciphered source entry-sequenced data set.
- CLEAR DD describes the target entry-sequenced data set.

The REPRO command copies and deciphers the enciphered data set from the source data set, ESDS1, to the target data set, ESDS3. The supplied plaintext data encrypting key is used to decipher the data. The parameters of the REPRO command are:

- INFILE points to the CRYPT DD statement identifying the source data set, ESDS1.
- OUTFILE points to the CLEAR DD statement, identifying the target data set, ESDS3, which must be empty. The defined maximum record size of the target entry-sequenced data set must be large enough to accommodate the largest source entry-sequenced data set record.

- DECIPHER indicates that the source data set is to be deciphered as it is copied to the target data set.
- DATAKEYVALUE indicates that keys are to be managed by the user, and supplies the plaintext private data encrypting key, X'53467568503A7C29', used to encipher the data.

# Chapter 30. SHCDS

The SHCDS command lists and changes recovery information kept by the SMSVSAM server. It also resets VSAM record-level sharing (RLS) settings in catalogs, allowing for non-RLS access or fallback from RLS processing. This command can be used in batch and TSO foreground environments.

Use this command cautiously. Many of the situations requiring the use of the SHCDS command are described in the *CICS Recovery and Restart Guide*. See *z/OS DFSMSdfp Storage Administration Reference* for details about administering VSAM RLS. See “Appendix C. Interpreting SHCDS Output Listings” on page 383 for SHCDS output listings.

The syntax of the access method services SHCDS command is:

| | |
|---|---|
| **SHCDS** | {[**LISTDS(***base-cluster***)**]\|<br>[**LISTSUBSYS(***subsystem*\|**ALL)**]\|<br>[**LISTSUBSYSDS(***subsystem*\|**ALL)**]\|<br>[**LISTRECOVERY(***base-cluster***)**\|<br>[**LISTALL**]\|<br>**[FRSETRR(***base-cluster***)]\|**<br>[**FRUNBIND(***base-cluster***)**]\|<br>[**FRBIND(***base-cluster***)**]\|<br>[**FRRESETRR(***base-cluster***)**]\|<br>[**FRDELETEUNBOUNDLOCKS(***base-cluster***)**]\|<br>[**PERMITNONRLSUPDATE(***base-cluster***)**]\|<br>**[DENYNONRLSUPDATE(***base-cluster***)]\|**<br>[**REMOVESUBSYS(***subsystem***)**]\|<br>**[CFREPAIR(**{**INFILE(***ddname***)**\|**INDATASET(***dsname***)**}<br>[({**LIST**\|**NOLIST**})]]\|<br>[**CFRESET(**{**INFILE(***ddname***)**\|**INDATASET(***dsname***)**}]<br>[({**LIST**\|**NOLIST**})]]\|<br>[**CFRESETDS(***base-cluster***)**]}<br>[**OUTFILE(***ddname***)**] |

*Base-cluster* is a fully or partially qualified VSAM data set name. The high-level qualifier must be specified. You can use an asterisk (*) for a subsequent qualifier, but then no lower-level qualifiers are allowed. For example, this is allowed:

```
A.*
```

This is not allowed:

```
A.*.B
```

Please see “Appendix C. Interpreting SHCDS Output Listings” on page 383 for examples and explanations of the output from the list parameters.

*Subsystem* is the name of an online system, such as CICS, as registered to the SMSVSAM server.

**Requirements:**

1. Various levels of authority are required to use the SHCDS parameters. See “Appendix A. Security Authorization Levels” on page 321 for further information.

© Copyright IBM Corp. 1973, 2002

2. A program that calls the SHCDS command must be APF-authorized. See “Appendix D. Invoking Access Method Services from Your Program” on page 393 for more information.
3. To use the SHCDS command in the TSO foreground, SHCDS must be added to the authorized command list (AUTHCMD) in the SYS1.PARMLIB member IKJTSO*xx* or added to the CSECT IKJEGSCU. Please see *z/OS TSO/E Customization* for more information.

## SHCDS Parameters

The SHCDS parameters provide for:

- Listing information kept by the SMSVSAM server and the catalog as related to VSAM RLS. Use:
  - LISTDS
  - LISTSUBSYS
  - LISTSUBSYSDS
  - LISTRECOVERY
  - LISTALL
- Controlling forward recovery, as well as preserving retained locks when a data set is moved or copied; and, in rare cases when forward recovery fails, deleting the locks. Use:
  - FRSETRR
  - FRUNBIND
  - FRBIND
  - FRRESETRR
  - FRDELETEUNBOUNDLOCKS
- Allowing non-RLS updates when forward recovery is required. Use:
  - PERMITNONRLSUPDATE
  - DENYNONRLSUPDATE
- Removing the SMSVSAM server’s knowledge of an inactive subsystem, thus forcing a cold start of the online application. Use REMOVESUBSYS only when procedures provided by the application have failed or you have no intention of ever using the subsystem again.
  - REMOVESUBSYS
- Resetting VSAM RLS indicators in the catalog, allowing reconstruction of RLS information or fallback from VSAM RLS. (See *z/OS DFSMSdfp Storage Administration Reference* for the fallback procedure.) Use:
  - CFREPAIR
  - CFRESET
  - CFRESETDS

## Required Parameters

SHCDS has no required parameters, but you must specify one of the optional parameters. OUTFILE is a second optional parameter you can specify.

## Optional Parameters

**LISTDS(***base-cluster***)**
lists:

- The assigned coupling facility structure name

- The subsystem type and status:
  - Active for batch
  - Active or failed for online
- Whether the VSAM sphere accessed in RLS mode requires recovery and a summary of the recovery indicators:
  - Forward recovery required
  - Retained locks
  - Lost locks
  - Locks unbound
  - Non-RLS update permitted
  - Permit-first-time switch

**Abbreviation:** LDS

**LISTSUBSYS(***subsystem***|ALL)**
lists information about a specific subsystem or all subsystems known to the SMSVSAM server:

- Subsystem status
  - Active for batch
  - Active or failed for online
- A summary showing whether the subsystem's shared data sets have:
  - Lost locks
  - Retained locks
  - Non-RLS update permitted

For an active subsystem, LISTSUBSYS gives the number of held locks, waiting lock requests, and retained locks. For a failed subsystem, LISTSUBSYS shows the number of retained locks.

**Abbreviation:** LSS

**LISTSUBSYSDS(***subsystem***|ALL)**
lists information about a specific subsystem or all subsystems known to the SMSVSAM server, including data sets that it is sharing. For each subsystem, it lists:

- Sharing protocol (online or batch)
- The status (active or failed)
- Recovery information for each shared data set:
  - Whether it has retained locks owned by this subsystem
  - Whether it has lost locks owned by this subsystem
  - Whether there are locks not bound to the data set
  - If forward recovery is required
  - If non-RLS update is permitted
  - The permit-first-time switch setting

**Abbreviation:** LSSDSL

**LISTRECOVERY(***base-cluster***)**
lists data sets requiring recovery and the subsystems that share those data sets. Recovery indicators listed are:

- Lost locks
- Retained locks
- Non-RLS update permitted
- Forward recovery required

**Abbreviation:** LRCVY

**LISTALL**
Lists all information related to recovery for subsystems and VSAM spheres accessed in RLS mode. The output from this parameter can be quite large.

**Abbreviation:** LALL

**FRSETRR(***base-cluster***)**
This parameter sets the forward-recovery-required indicator. Until reset with the FRRESETRR parameter, access is prevented until forward recovery is complete.

If you use a forward recovery utility that supports RLS, such as CICSVR, do not use this parameter.

**Abbreviation:** SETRR

**FRUNBIND(***base-cluster***)**
This parameter unbinds the retained locks prior to restoring or moving the data set. These locks protect uncommitted changes and are needed for eventual backout. They must be rebound by using the FRBIND parameter.

If you use a forward recovery utility that supports RLS, such as CICSVR, do not use this parameter.

**Abbreviation:** UNB

**FRBIND(***base-cluster***)**
Use this parameter after BLDINDEX to rebind the associated locks to the restored data set.

**Attention:** Between the unbind and the bind, do not delete any clusters in the sphere or change their names.

If you use a forward recovery utility that supports RLS, such as CICSVR, do not use this parameter.

**Abbreviation:** BIND

**FRRESETRR(***base-cluster***)**
Use this parameter after forward recovery is complete and after locks have been bound to the new location of the data set using FRBIND. This allows access to the newly recovered data set by applications other than the forward recovery application.

If you use a forward recovery utility that supports RLS, such as CICSVR, do not use this parameter.

**Abbreviation:** RESET

**FRDELETEUNBOUNDLOCKS(***base-cluster***)**
The FRDELETEUNBOUNDLOCKS parameter lets you delete locks in the rare case when a successful forward recovery is not possible. Every attempt should be made to complete forward recovery, whether using a product such as CICSVR that supports VSAM RLS or using another forward recovery procedure.

If forward recovery does not successfully complete, locks cannot be re-associated (bound) to the new version of the data set, because these locks do not provide the protection that online backout requires.

Before using this parameter, check the documentation for your online application. For CICS, the procedure is documented in the *CICS Recovery and Restart Guide*.

**Abbreviation:** DUNBL

**PERMITNONRLSUPDATE(***base-cluster***)**
Allows a data set with pending RLS recovery to be opened for output in non-RLS mode. This command is used when it is necessary to run critical batch updates and RLS recovery cannot first be completed. This is reset the next time the data set is accessed for RLS. If after using PERMITNONRLSUPDATE, you do not run a non-RLS batch job, you must use DENYNONRLSUPDATE to prevent non-RLS updates.

**Abbreviation:** PERMT

**DENYNONRLSUPDATE(***base-cluster***)**
If you inadvertently issue PERMITNONRLSUPDATE, use this parameter to reset the effect of PERMITNONRLSUPDATE.

If recovery was pending, but you did not run a non-RLS batch job, you must use this parameter. If not reset, CICS takes action assuming the data set has been opened for update in non-RLS mode.

Do not use DENYNONRLSUPDATE if you do indeed run non-RLS work after specifying PERMITNONRLSUPDATE. The permit status is reset the next time the data set is opened in RLS mode.

**Abbreviation:** DENY

**REMOVESUBSYS(***subsystem***)**
Use this parameter to remove any knowledge of recovery owed to SMSVSAM by the named subsystem, including locks protecting uncommitted updates.

Normally, a failed online application would be restarted so that it can do the required backouts and release locks protecting uncommitted updates. However, sometimes it might be necessary to cold start the online application. For more information about cold starts, see *CICS Recovery and Restart Guide*.

Use of this parameter is equivalent to cold starting the named subsystem with respect to the SMSVSAM server. Use REMOVESUBSYS for the rare cases where either there is no intention of ever running the subsystem again or the application's cold start procedures cannot be used. An example of an appropriate use of REMOVESUBSYS would be removing a test system that is no longer needed.

If the removed subsystem is ever run again, every effort should be made to cold start the subsystem.

**Attention:** Use of REMOVESUBSYS can result in loss of data integrity.

**Abbreviation:** RSS

**CFREPAIR({INFILE***(ddname)***|INDATASET***(dsname)}*
Use this command to reconstruct the RLS indicators for all applicable data sets in a restored catalog. CFREPAIR uses information known to the VSAM RLS server at the time the SHCDS command is used. The catalog must be import-connected on all systems to the master catalog before the CFREPAIR parameter can be used.

**INFILE***(ddname)*
Indicates which DD statement defines the catalog to be processed.

**INDATASET***(dsname)*
Use this to specify the name of the catalog to be processed.

**({LIST|NOLIST})**
Optional subparameters, which control the information returned by the CFREPAIR parameter.

**LIST**
Requests a list of data sets for which CFREPAIR successfully restored the RLS information. If you do not specify this subparameter, CFREPAIR lists only those data sets whose RLS information could not be restored.

**NOLIST**
Only data sets whose information could not be restored are listed. Using this subparameter is the same as not specifying LIST or NOLIST.

**Abbreviation:** CFREP

**CFRESET({INFILE***(ddname)***|INDATASET(***dsname***)}**
Use this parameter if you've decided to fall back from using VSAM RLS. It clears VSAM RLS indicators in the catalog for all applicable data sets. A detailed fallback procedure is included in the *z/OS DFSMSdfp Storage Administration Reference*. Also, please see the *CICS Recovery and Restart Guide* for information specific to CICS.

If the catalog is later restored, use CFREPAIR to reconstruct critical information required by the SMSVSAM server.

**INFILE***(ddname)*
Specifies the DD name of the catalog to be processed.

**INDATASET***(dsname)*
Specifies the data set name of the catalog to be processed.

**({LIST|NOLIST})**
Optional subparameters, which control the information returned by the CFRESET parameter.

**LIST**
Requests a list of data sets for which CFRESET successfully processed the RLS indicators. If you do not specify this subparameter, CFRESET lists only those data sets whose indicators were not cleared.

**NOLIST**
Only data sets that were not successfully processed are listed. Using this subparameter is the same as not specifying LIST or NOLIST.

**Abbreviation:** CFRES

**CFRESETDS(***base-cluster***)**
Use this parameter if you've decided to fall back from using VSAM RLS. It clears VSAM RLS indicators in the catalog for all applicable data sets. CFRESETDS This parameter differs from CFRESET in that it lets you select one or more data sets for fallback. CFRESETDS lists all data sets processed, not just those with errors.

A detailed fallback procedure is included in the *z/OS DFSMSdfp Storage Administration Reference*. Also, please see the *CICS Recovery and Restart Guide* for information specific to CICS.

**Abbreviation:** CFRDS

**OUTFILE(*ddname*)**

Specifies a data set, other than the SYSPRINT data set, to receive the output produced by the SHCDS command.

*ddname* identifies the DD statement of the alternate target data set.

**Abbreviation:** OUTDD

## SCHDS Examples

The SCHDS command can perform the functions shown in the following examples.

### Using PERMITNONRLSUPDATE With a Generic Data Set Name Specification: Example 1

The following example shows using the SHCDS subparameter PERMITNONRLSUPDATE with a generic data set name specification.

```
  /* SET NONRLS UPDATE ON                                    */
  SHCDS PERMITNONRLSUPDATE(SYSPLEX.PERMIT.*)
IDC2917I NO RACF PROFILE ON STGADMIN.IGWSHCDS.REPAIR
IDC01885I NON-RLS UPDATE PERMITTED FOR SYSPLEX.PERMIT.CLUS2
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

### Listing Data Sets With the High-Level Qualifier SYSPLEX: Example 2

The following example lists the data sets with the high-level qualifier of SYSPLEX.

In general, when a base cluster name can be specified for the SHCDS command, a generic can be used.

```
  SHCDS LISTDS(SYSPLEX.*)
IDC2917I NO RACF PROFILE ON STGADMIN.IGWSHCDS.REPAIR
----- LISTING FROM SHCDS ----- IDCSH02
--------------------------------------------------------------------------------
DATA SET NAME----SYSPLEX.PERMIT.CLUS2
  CACHE STRUCTURE----CACHE01
  RETAINED LOCKS---------YES   NON-RLS UPDATE PERMITTED--------YES
  LOST LOCKS--------------NO   PERMIT FIRST TIME---------------YES
  LOCKS NOT BOUND---------NO   FORWARD RECOVERY REQUIRED--------NO
  RECOVERABLE------------YES

                           SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  RETLK05A     ONLINE--FAILED    YES         NO       YES
DATA SET NAME----SYSPLEX.RETAINED.CLUS1
  CACHE STRUCTURE----CACHE01
  RETAINED LOCKS---------YES   NON-RLS UPDATE PERMITTED---------NO
  LOST LOCKS--------------NO   PERMIT FIRST TIME----------------NO
  LOCKS NOT BOUND---------NO   FORWARD RECOVERY REQUIRED--------NO
  RECOVERABLE------------YES

                           SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  RETLK05A     ONLINE--FAILED    YES         NO       NO
DATA SET NAME----SYSPLEX.SHARED.CLUS4
  CACHE STRUCTURE----CACHE01
  RETAINED LOCKS---------YES   NON-RLS UPDATE PERMITTED---------NO
  LOST LOCKS--------------NO   PERMIT FIRST TIME----------------NO
  LOCKS NOT BOUND---------NO   FORWARD RECOVERY REQUIRED--------NO
  RECOVERABLE------------YES
```

```
                              SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  RETLK05A     ONLINE--FAILED    YES         NO       NO
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

# Chapter 31. VERIFY

The VERIFY command causes a catalog to correctly reflect the end of a VSAM data set after an error occurs while closing a VSAM data set. The error might have caused the catalog to be incorrect. The syntax of the VERIFY command is:

| | |
|---|---|
| **VERIFY** | **{FILE(***ddname***)\| DATASET(***entryname***)}** |

VERIFY can be abbreviated: VFY

**Exception:** If you use the VERIFY command on a linear data set, the explicit VERIFY function is bypassed. The linear data set is successfully opened and closed, without an error message, which resets the open indicator for the data set.

## VERIFY Parameters

The VERIFY command uses the following parameters.

### Required Parameter

**FILE(***ddname***)**
*ddname* names a DD statement identifying the cluster or alternate index being verified. For further information, see "Using VERIFY to Fix Improperly Closed Data Sets" in *z/OS DFSMS: Using Data Sets*. The data set is deallocated at the VERIFY job step termination.

**DATASET(***entryname***)**
specifies the name of the object being verified. If DATASET is specified, the object is dynamically allocated. The data set is deallocated dynamically at job termination.

**Abbreviation:** DS

You can use the VERIFY command following a system error that caused a component opened for update processing to be improperly closed. You can also use it to verify an entry-sequenced data set defined with RECOVERY that was open in create mode when the system error occurred. However, the entry-sequenced data set must contain records (not be empty) to successfully verify.

**Recommendation:** When sharing data sets between different processors, we recommend that you run VERIFY as the first step of a job stream to prevent job termination caused by an open access method control block (ACB) error code if the other processor already has the data set open.

## VERIFY Example

The VERIFY command can perform the following function.

### Upgrade a Data Set's End-of-File Information

When an improperly closed data set (a data set closed as a result of system error) is opened, the VSAM OPEN routines set a return code to indicate the data set's

© Copyright IBM Corp. 1973, 2002

cataloged information might not be accurate. You can upgrade the end of data (EOD) and end of key range (EOKR) information (so that it is accurate when the data set is next opened) by closing the data set [4] and issuing the VERIFY command:

```
//VERIFY   JOB  ...
//FIXEOD   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
           LISTCAT   ENTRIES(TAROUT) -
                     ALL
           VERIFY    DATASET(TAROUT)
           LISTCAT   ENTRIES(TAROUT) -
                     ALL
/*
```

The first LISTCAT command lists the data set's cataloged information, showing the data set's parameters as they were when the data set was last properly closed.

The VERIFY command updates the data set's cataloged information to show the data set's real EOD and EOKR values.

The second LISTCAT command lists the data set's cataloged information again. This time, the EOD and EOKR information shows the point where processing stopped because of system error. This information should help you determine how much data was added correctly before the system stopped.

VERIFY will update only the high-used RBA fields for the data set, not any record counts.

4. When the data set is opened (first OPEN after system error), VSAM OPEN sets a "data set improperly closed" return code. When the data set is closed properly, VSAM CLOSE resets the "data set improperly closed" indicator but does not upgrade erroneous catalog information that resulted from the system error. Subsequently, when the data set is next opened, its EOD and EOKR information might still be erroneous (until VERIFY is entered to correct it), but VSAM OPEN sets the "data set opened correctly" return code.

# Appendix A. Security Authorization Levels

This appendix contains tables that show the required Resource Access Control Facility (RACF) authorization levels for access method services commands. These tables include information for both non-SMS and SMS-managed data sets.

If no RACF profile exists for a data set, you are authorized to access that data set without further RACF checking. The catalog RACF profile is not checked, even if it exists.

The following tables are contained in this appendix:


| Table | Contents |
|---|---|
| **Table 5** | Required Security Authorization for Catalogs |
| **Table 6** | Required Security Authorization for VSAM Data Sets |
| **Table 7** | Required Security Authorization for Non-VSAM Data Sets |
| **Table 8** | Required Security Authorization for LISTCAT |
| **Table 9** | Required Security Authorization for Data Set Operations |
| **Table 10** | Required Security Authorization for VOLCAT Operations |
| **Table 11** | RACF Facility Class Authorization for IDCAMS Commands |
| **Table 12** | Required Authorization for SHCDS Parameters |


## Required RACF Authorization Tables

*Table 5. Required Security Authorization for Catalogs*

| Function Performed | Required RACF for User Catalog | Required RACF for Master Catalog | Comments |
|---|---|---|---|
| Alter UCAT | Alter | Alter | Either UCAT or MCAT authorization is sufficient, see note 1. |
| Define Alias of UCAT | None | Update | MCAT update authority is not checked if the user has authority for the facility class STGADMIN.IGG.DEFDEL.UALIAS. |
| Define UCAT/MCAT | Alter | Update | |
| Delete Alias of UCAT | Alter | Alter | UCAT/MCAT update authority is not checked if the user has authority for the facility class STGADMIN.IGG.DEFDEL.UALIAS. Either UCAT or MCAT authorization is sufficient, see note 1. |
| Delete UCAT | Alter | None | |
| Export Disconnect of UCAT | Alter | None | |
| Import Connect Alias of UCAT | Alter | Update | |
| Import Connect of UCAT | Alter | Update | |
| PRINT | Alter | Alter | |

**Notes:**

1. Alter is an ″OR″ function. Either alter to the user catalog or alter to the master catalog is required, but not both.

**Note:** If not indicated in the comments, the same authorization applies to both non-SMS and SMS.

© Copyright IBM Corp. 1973, 2002

## Security Authorizations

*Table 6. Required Security Authorization for VSAM Data Sets*

| Function Performed | Required RACF for Data Set | Required RACF for Catalog | Comments |
|---|---|---|---|
| Alter Cluster | Alter | None | • The same authorization applies to both non-SMS and SMS.<br>• See note 1. |
| Alter Cluster Component | Alter | None | • The same authorization applies to both non-SMS and SMS.<br>• See notes 1 and 2. |
| Alter Cluster Newname | Alter | None | • Alter is required to the new name.<br>• See note 1. |
| Alter Pagespace | Alter | None | • The same authorization applies to both non-SMS and SMS.<br>• See notes 1 and 2. |
| Define AIX | Alter | Update | See notes 2 and 3. |
| Define Cluster | Alter | Update | See note 3. |
| Define Cluster Model | Alter | Update | See note 3. |
| Define Pagespace | Alter | Update | See notes 2 and 3. |
| Define Path | Alter | Update | See notes 2 and 3. |
| Define Recatalog VSAM | Alter | Update | See notes 2 and 3. |
| Delete AIX | Alter | Alter | See notes 2 and 4. |
| Delete Cluster | Alter | Alter | See note 4. |
| Delete Cluster Noscratch | Alter | Alter | See note 4. |
| Delete NVR/VVR | None | Alter | |
| Delete Pagespace | Alter | Alter | See notes 2 and 4. |
| Delete Path | Alter | Alter | See notes 2 and 4. |
| Diagnose Catalog | Alter | None | The data set is the user catalog. |
| Diagnose VVDS | | Alter | |
| Examine Catalog | Alter | None | The data set is the user catalog. |
| Examine Data Set | Control | None | |
| Export Cluster | Alter | Alter | Alter authority to either the data set or the catalog is sufficient. |
| Export UCAT | Alter | None | The data set is the user catalog. |
| Import Into Empty | Read | Alter | The data set is the user catalog |

*Table 6. Required Security Authorization for VSAM Data Sets (continued)*

| Function Performed | Required RACF for Data Set | Required RACF for Catalog | Comments |
|---|---|---|---|
| **Notes:**<br>1. Alter is an ″OR″ function. Either alter to the data set or alter to the catalog is required, but not both.<br>2. Authorization is always to the cluster name for VSAM components cataloged with the integrated catalog facility. Integrated catalog facility does not check for individual component names such as data, index, path, or AIX.<br>3. No authority is required to the catalog for the define of SMS-managed data sets unless the catalog is the master catalog. Update authority is required if the catalog is a master catalog.<br>4. Delete is an “OR” function for both non-SMS- and SMS-managed data sets. Either alter authority to the data set or alter authority to the catalog is required to delete the data set, but not both. | | | |
| **Note:** If no profile exists for a data set, then the user is considered authorized. The catalog profile is not checked, even if it exists. | | | |

*Table 7. Required Security Authorization for Non-VSAM Data Sets*

| Function Performed | Required RACF for Data Set | Required RACF for Catalog | Comments |
|---|---|---|---|
| Alter Non-VSAM | Alter | None | • The same authorization applies to both non-SMS and SMS.<br>• See note 1. |
| Define Alias of a Non-VSAM | None | Update | |
| Define Alias of a SMS Non-VSAM | None | None | |
| Define GDG | Alter | Update | Although a GDG is not SMS, these authorities still apply if the catalog is SMS. |
| Define GDS | Alter | Update | See note 2. |
| Define GDS SMS | Alter | None | See note 2. |
| Define Non-VSAM Non-SMS | Alter | Update | See note 3. |
| Define Non-VSAM Recatalog Non-SMS | Alter | Update | |
| Define Non-VSAM SMS | Alter | None | Master catalog requires update authority. |
| Define Non-VSAM Recatalog SMS | Alter | Update | |
| Delete Alias of a Non-VSAM | Alter | Alter | See note 4. |
| Delete GDG | Alter | Alter | Alter authorization either to the data set or to the catalog is sufficient. |
| Delete Non-VSAM Scratch non-SMS | Alter | Alter | See notes 4 and 6. |
| Delete Non-VSAM Noscratch Non-SMS | Alter | Alter | Alter authorization either to the data set or to the catalog is sufficient. |
| Delete Non-VSAM SMS | Alter | Alter | See notes 4 and 5. |

*Table 7. Required Security Authorization for Non-VSAM Data Sets (continued)*

| Function Performed | Required RACF for Data Set | Required RACF for Catalog | Comments |
|---|---|---|---|
| **Notes:**<br>1. Alter is an ″OR″ function. Either alter to the data set or alter to the catalog is required, but not both.<br>2. To define a GDS, you must either have update authority to the GDG or alter authority to the catalog.<br>3. If this is a data set that resides on tape, SETROPTS TAPEDSN must be entered for RACF. If NOTAPEDSN (the default) is in effect, then update authority to the catalog is required to define or delete the data set.<br>4. Delete is an “OR” function for both non-SMS- and SMS-managed data sets. Either alter authority to the data set or alter authority to the catalog is required to delete the data set, but not both.<br>5. If the data set is cataloged in the master catalog you must have Update authority to the master catalog and Alter authority to the data set.<br>6. If the data set does not have a RACF profile we will require UPDATE authority to its catalog. | | | |

*Table 8. Required Security Authorization for LISTCAT*

| Function Performed | Required RACF for Data Set | Required RACF for Catalog | Comments |
|---|---|---|---|
| LISTCAT ALL | Read | None | Allows listing entries you have data set authority to. Passwords are not displayed. |
| LISTCAT ALL | None | Read | Allows listing all entries. Passwords are not displayed. |
| LISTCAT ALL | None | Alter | Allows listing all entries. Passwords are displayed. |
| LISTCAT Entry | Read | Read | Password is not displayed. |
| LISTCAT Entry | Alter | Alter | Password is displayed. |

*Table 9. Required Security Authorization for Data Set Operations*

| Function Performed | Required RACF for Input Data Set | Required RACF for Output Data Set | Comments |
|---|---|---|---|
| BLDINDEX | n/a | Update | Authority is to the base cluster. |
| CNVTCAT | Alter | Alter | |
| DCOLLECT | n/a | Update | |
| Export Data Set | Alter | Update | |
| REPRO | Read | Update | |

*Table 10. Required Security Authorization for VOLCAT Operations*

| Function Performed | Required RACF for LIB/VOL | Required RACF for VOLCAT Operations | Comments |
|---|---|---|---|
| Alter LIBENT | none | Alter | |
| Alter VOLENT | none | Alter | |
| Create LIBENT | none | Update | |
| Create VOLENT | none | Update | |
| Delete LIBENT | none | Alter | |

*Table 10. Required Security Authorization for VOLCAT Operations (continued)*

| Function Performed | Required RACF for LIB/VOL | Required RACF for VOLCAT Operations | Comments |
|---|---|---|---|
| Delete VOLENT | none | Alter | |
| Listc LIBENT | none | none | |
| Listc VOLENT | none | none | |

*Table 11. RACF Facility Class Authorization for IDCAMS Commands*

| IDCAMS Command | Required RACF Facility Class Authorization | Function Authorized |
|---|---|---|
| ALTER | STGADMIN.IGG.DIRCAT | Define a data set into a particular catalog which is not the one chosen according to a regular search for SMS-managed data sets. |
| ALTER LIBRARYENTRY | STGADMIN.IGG.LIBRARY | Alter a tape library entry. |
| ALTER VOLUMEENTRY | STGADMIN.IGG.LIBRARY | Alter a tape volume entry. |
| BUILD INDEX | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |
| CREATE LIBRARYENTRY | STGADMIN.IGG.LIBRARY | Create a tape library entry. |
| CREATE VOLUMEENTRY | STGADMIN.IGG.LIBRARY | Create a tape volume entry. |
| DCOLLECT | STGADMIN.IDC.DCOLLECT | Access the DCOLLECT function. |
| DEFINE ALIAS | STGADMIN.IGG.DEFDEL.UALIAS | Define an alias for a user catalog. |
| DEFINE ALTERNATEINDEX | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |
| DEFINE CLUSTER | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |
| DEFINE NONVSAM | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |
| DEFINE PAGESPACE | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |
| DELETE | STGADMIN.IGG.DEFDEL.UALIAS | Delete an alias for a user catalog. |
| DELETE GDG | STGADMIN.IGG.DELGDG.FORCE | Delete a GDG using the FORCE option. |
| DELETE GDG | STGADMIN.IGG.DELGDG.RECOVERY | DELETE a GDG using the RECOVERY option. |
| DELETE | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |
| DELETE LIBRARYENTRY | STGADMIN.IGG.LIBRARY | Delete a tape library entry or a tape volume entry. |
| DIAGNOSE | STGADMIN.IDC.DIAGNOSE.CATALOG | Open an ICF catalog without performing normal catalog security processing. |
| DIAGNOSE | STGADMIN.IDC.DIAGNOSE.VVDS | Open an ICF catalog without performing normal catalog security processing. |

*Table 11. RACF Facility Class Authorization for IDCAMS Commands (continued)*

| IDCAMS Command | Required RACF Facility Class Authorization | Function Authorized |
|---|---|---|
| EXAMINE | STGADMIN.IDC.EXAMINE.DATASET | Open an ICF catalog without performing usual catalog security processing. |
| EXPORT | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |
| EXPORT DISCONNECT | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |
| IMPORT | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |
| IMPORT CONNECT | STGADMIN.IGG.DIRCAT | Specify catalog names for SMS-managed data sets. |

*Table 12. Required Authorization for SHCDS Subcommands*

| SHCDS Parameter | Required Authority |
|---|---|
| CFREPAIR | Alter authority to the catalog and update authority to STGADMIN.IGWSHCDS.REPAIR. |
| CFRESET | Alter authority to the catalog and update authority to STGADMIN.IGWSHCDS.REPAIR. |
| CFRESETDS | Update authority to STGADMIN.IGWSHCDS.REPAIR and to the specified data sets. |
| DENYNONRLSUPDATE | Update authority to STGADMIN.IGWSHCDS.REPAIR and the base cluster. |
| FRSETRR | Update authority to STGADMIN.IGWSHCDS.REPAIR and the base cluster. |
| FRUNBIND | Update authority to STGADMIN.IGWSHCDS.REPAIR and the base cluster. |
| FRBIND | Update authority to STGADMIN.IGWSHCDS.REPAIR and the base cluster. |
| FRRESETRR | Update authority to STGADMIN.IGWSHCDS.REPAIR and the base cluster. |
| FRDELETEUNBOUNDLOCKS | Update authority to STGADMIN.IGWSHCDS.REPAIR and the base cluster. |
| LISTDS | Read authority to STGADMIN.IGWSHCDS.REPAIR |
| LISTSUBSYS | Read authority to STGADMIN.IGWSHCDS.REPAIR |
| LISTSUBSYSDS | Read authority to STGADMIN.IGWSHCDS.REPAIR |
| LISTRECOVERY | Read authority to STGADMIN.IGWSHCDS.REPAIR |
| LISTALL | Read authority to STGADMIN.IGWSHCDS.REPAIR |
| PERMITNONRLSUPDATE | Update authority to STGADMIN.IGWSHCDS.REPAIR and the base cluster. |
| REMOVESUBSYS | Update authority to STGADMIN.IGWSHCDS.REPAIR and the SUBSYSNM class. |

# Appendix B. Interpreting LISTCAT Output Listings

The various LISTCAT command options allow you to select the LISTCAT output that gives you the information you want. This appendix provides information on the structure of LISTCAT output if you use certain options. Fields that can be printed for each type of catalog entry are listed and described.

Each catalog entry is identified by its type (for example: cluster, non-VSAM, data) and by its entryname. Entries are listed in alphabetic order of the entrynames, unless the ENTRIES parameter is used. The entries are then listed in the order they are specified in the ENTRIES parameter.

An entry that has associated entries is immediately followed by the listing of each associated entry. That is, a cluster's data component (and, if the cluster is key-sequenced, its index component) is listed immediately following the cluster. The associated entry is excluded if type options (CLUSTER, DATA, SPACE, and so on) or a generic entryname list are specified.

On DFSMS/MVS 1.3 systems, LISTCAT returns information specific to VSAM record-level sharing (RLS). On 1.2 or lower systems, LISTCAT returns only an indication whether or not RLS information exists.

```
RLSDATA
  EXISTS
```

or

```
RLSDATA
  NODATA
```

This appendix has three parts:

- "LISTCAT Output Keywords" lists all field names that can be listed for each type of entry.
- "Description of Keyword Fields" on page 335 describes each field name within a group of related field names.
- "Examples of LISTCAT Output Listings" on page 346 describes and illustrates the LISTCAT output that results when various LISTCAT options are specified.

## LISTCAT Output Keywords

This section lists the field names associated with each type of catalog entry. Each field name is followed by an abbreviation that points to a group of related field descriptions in the next section. Keywords are listed in alphabetic order, not in the order of appearance in the LISTCAT output.

The group names and abbreviations are:

| Abbreviations | Group Names |
|---|---|
| **ALC** | allocation group |
| **ASN** | associations group |
| **ATT** | attributes group |
| **GDG** | generation data group base entry, special fields |
| **HIS** | history group |
| **NVS** | non-VSAM entry, special field |
| **PRT** | protection group |
| **STA** | statistics group |

© Copyright IBM Corp. 1973, 2002

**VLS** volumes group

## Alias Entry Keywords

ASSOCIATIONS (ASN)
entryname (HIS)
HISTORY (HIS)
RELEASE (HIS)

## Alternate-Index Entry Keywords

ASSOCIATIONS (ASN)
ATTEMPTS (PRT)
ATTRIBUTES (ATT)
CLUSTER (ASN)
CODE (PRT)
CONTROLPW (PRT)
DATA (ASN)
entryname (HIS)
HISTORY (HIS)
    CREATION (HIS)
    DATASET-OWNER(HIS)
    EXPIRATION (HIS)
    RELEASE (HIS)
    SMS-MANAGED
INDEX (ASN)
MASTERPW (PRT)
NOUPGRADE (ATT)
PATH (ASN)
PROTECTION (PRT)
RACF (PRT)
READPW (PRT)
UPDATEPW (PRT)
UPGRADE (ATT)
USAR (PRT)
USVR (PRT)

## Cluster Entry Keywords

AIX (ASN)
ASSOCIATIONS (ASN)
ATTEMPTS (PRT)
CODE (PRT)
CONTROLPW (PRT)
DATA (ASN)
entryname (HIS)
HISTORY (HIS)
    CREATION (HIS)
    DATASET-OWNER (HIS)
    EXPIRATION (HIS)
    RELEASE (HIS)
INDEX (ASN)
MASTERPW (PRT)
PATH (ASN)
PROTECTION (PRT)

RACF (PRT)
READPW (PRT)
RLSDATA
  FRLOG
  LOG
  LOGSTREAMID
  RECOVERY REQUIRED
  RECOVERY TIMESTAMP GMT
  RECOVERY TIMESTAMP LOCAL
  RLS IN USE
  VSAM QUIESCED
SMSDATA
    BWO
    BWO TIMESTAMP
    BWO STATUS
    DATACLASS
    LBACKUP
    MANAGEMENTCLASS
    STORAGECLASS
UPDATEPW (PRT)
USAR (PRT)
USVR (PRT)

## Data Entry Keywords

ACT-DICT-TOKEN (ATT)
ACCOUNT-INFO
AIX (ASN)
ALLOCATION (ALC)
ASSOCIATIONS (ASN)
ATTEMPTS (PRT)
ATTRIBUTES (ATT)
AVGLRECL (ATT)
AXRKP (ATT)
BINARY (ATT)
BIND (ATT)
BUFND (ATT)
BUFSPACE (ATT)
BYTES/TRACK (VLS)
CCSID (ATT)
CI/CA (ATT)
CISIZE (ATT)
CLUSTER (ASN)
CODE (PRT)
COMP-FORMT (ATT)
COMP-USER-DATA-SIZE (STA)
CONTROLPW (PRT)
CYLFAULT (ATT)
DDMEXIST (ATT)
DEVTYPE (VLS)
DSTGWAIT (ATT)
entryname (HIS)
ERASE (ATT)
EXCPEXIT (ATT)
EXCPS (STA)
EXT-ADDR (ATT)

EXTENT-NUMBER (VLS)
EXTENT-TYPE (VLS)
EXTENTS (STA)
EXTENTS (VLS)
    HIGH-CCHH (VLS)
    HIGH-RBA (VLS)
    LOW-CCHH (VLS)
    LOW-RBA (VLS)
    TRACKS (VLS)
FREESPACE-%CI (STA)
FREESPACE-%CA (STA)
FREESPC (STA)
HI-KEY-RBA (VLS)
HI-A-RBA (ALC)
HI-U-RBA (ALC)
HI-A-RBA (VLS)
HI-U-RBA (VLS)
HIGH-KEY (VLS)
HISTORY (HIS)
    CREATION (HIS)
    DATASET-OWNER (HIS)
    EXPIRATION (HIS)
    RELEASE (HIS)
ICFCATALOG (ATT)
INDEX (ASN)
INDEXED (ATT)
INH-UPDATE (ATT)
KEYLEN (ATT)
LINEAR (ATT)
LOW-KEY (VLS)
MASTERPW (PRT)
MAXLRECL (ATT)
MAXRECS (ATT)
NOERASE (ATT)
NONINDEXED (ATT)
NONSPANNED (ATT)
NONUNIQKEY (ATT)
NOREUSE (ATT)
NOSWAP (ATT)
NOTRKOVFL (ATT)
NOTUSABLE (ATT)
NOWRITECHK (ATT)
NUMBERED (ATT)
PGSPC (ASN)
PHYRECS/TRK (VLS)
PHYREC/SIZE (VLS)
PROTECTION (PRT)
RACF (PRT)
READPW (PRT)
RECOVERY (ATT)
REC-DELETED (STA)
REC-INSERTED (STA)
REC-RETRIEVED (STA)
REC-TOTAL (STA)
REC-UPDATED (STA)
RECORDS/CI (ATT)

RKP (ATT)
REUSE (ATT)
RECVABLE (ATT)
SHROPTNS (ATT)
SPACE-PRI (ALC)
SPACE-SEC (ALC)
SPACE-TYPE (ALC)
SPEED (ATT)
SPLITS-CA (STA)
SPLITS-CI (STA)
SPANNED (ATT)
STATISTICS (STA)
STRIPE-COUNT (ATT)
STRNO (ATT)
SWAP (ATT)
SYSTEM-TIMESTAMP (STA)
TEMP-EXP (ATT)
TEXT (ATT)
TRACKS/CA (VLS)
TRKOVFL (ATT)
UNORDERED (ATT)
UNIQUE (ATT)
UNIQUEKEY (ATT)
UPDATEPW (PRT)
USAR (PRT)
USER-DATA-SIZE (STA)
USVR (PRT)
VOLFLAG (VLS)
VOLSER (VLS)
VOLUMES (VLS)
WRITECHECK (ATT)

## Index Entry Keywords

AIX (ASN)
ALLOCATION (ALC)
ASSOCIATIONS (ASN)
ATTEMPTS (PRT)
ATTRIBUTES (ATT)
AVGLRECL (ATT)
BIND (ATT)
BUFNI (ATT)
BUFSPACE (ATT)
CI/CA (ATT)
CISIZE (ATT)
CLUSTER (ASN)
CODE (PRT)
CONTROLPW (PRT)
CYLFAULT (ATT)
DEVTYPE (VLS)
DSTGWAIT (ATT)
entryname (HIS)
ERASE (ATT)
EXCPEXIT (ATT)
EXCPS (STA)
EXTENTS (STA)

EXTENT-NUMBER (VLS)
EXTENT-TYPE (VLS)
EXTENTS (VLS)
    HIGH-CCHH (VLS)
    HIGH-RBA (VLS)
    LOW-CCHH (VLS)
    LOW-RBA (VLS)
    TRACKS (VLS)
FREESPACE-%CI (STA)
FREESPACE-%CA (STA)
FREESPC (STA)
HI-A-RBA (ALC)
HI-U-RBA (ALC)
HI-A-RBA (VLS)
HI-U-RBA (VLS)
HIGH-KEY (VLS)
HISTORY (HIS)
    CREATION (HIS)
    DATASET-OWNER (HIS)
    EXPIRATION (HIS)
    RELEASE (HIS)
INDEX (STA)
    ENTRIES/SECT (STA)
    HI-LEVEL-RBA (STA)
    LEVELS (STA)
    SEQ-SET-RBA (STA)
INH-UPDATE (ATT)
KEYLEN (ATT)
LOW-KEY (VLS)
MASTERPW (PRT)
MAXLRECL (ATT)
NOERASE (ATT)
NOREUSE (ATT)
NOTUSABLE (ATT)
NOWRITECHK (ATT)
PHYRECS/TRK (VLS)
PHYREC-SIZE (VLS)
PROTECTION (PRT)
RACF (PRT)
READPW (PRT)
RECOVERY (ATT)
REC-DELETED (STA)
REC-INSERTED (STA)
REC-RETRIEVED (STA)
REC-TOTAL (STA)
REC-UPDATED (STA)
RKP (ATT)
REUSE (ATT)
SHROPTNS (ATT)
SPACE-PRI (ALC)
SPACE-SEC (ALC)
SPACE-TYPE (ALC)
SPEED (ATT)
SPLITS-CA (STA)
SPLITS-CI (STA)
STATISTICS (STA)

SYSTEM-TIMESTAMP (STA)
TEMP-EXP (ATT)
TRACKS/CA (VLS)
UNIQUE (ATT)
UPDATEPW (PRT)
USAR (PRT)
USVR (PRT)
VOLFLAG (VLS)
VOLSER (VLS)
VOLUME (VLS)
WRITECHECK (ATT)

## Generation Data Group Base Entry Keywords

ASSOCIATIONS (ASN)
ATTRIBUTES (GDG)
    EMPTY (GDG)
    LIMIT (GDG)
    NOEMPTY (GDG)
    NOSCRATCH (GDG)
    SCRATCH (GDG)
entryname (HIS)
HISTORY (HIS)
    CREATION (HIS)
    DATASET-OWNER (HIS)
    EXPIRATION (HIS)
    RELEASE (HIS)
NONVSAM (ASN)

## Non-VSAM Entry Keywords

ACT-DICT-TOKEN (ATT)
ALIAS (ASN)
ASSOCIATIONS (ASN)
BINARY (ATT)
CCSID (ATT)
COMP-FORMT (ATT)
COMP-USER-DATA-SIZE (STA)
DDMEXIST (ATT)
DEVTYPE(VLS)
entryname (HIS)
FSEQN (NVS)
HISTORY (HIS)
    CREATION (HIS)
    DATASET-OWNER (HIS)
    EXPIRATION (HIS)
    RELEASE (HIS)
    STATUS
OAMDATA
    DIRECTORYTOKEN
SIZES-VALID (STA)
SMSDATA
    DATACLASS
    MANAGEMENTCLASS
    STORAGECLASS

LBACKUP
STRIPE-COUNT (ATT)
TEXT (ATT)
USER-DATA-SIZE (STA)
VOLSER(VLS)

## Page Space Entry Keywords

ASSOCIATIONS (ASN)
ATTEMPTS (PRT)
CODE (PRT)
CONTROLPW (PRT)
DATA (ASN)
entryname (HIS)
HISTORY (HIS)
CREATION (HIS)
DATASET-OWNER (HIS)
EXPIRATION (HIS)
RELEASE (HIS)
INDEX (ASN)
MASTERPW (PRT)
PROTECTION (PRT)
RACF (PRT)
READPW (PRT)
UPDATEPW (PRT)
USAR (PRT)
USVR (PRT)

## Path Entry Keywords

AIX (ASN)
ASSOCIATIONS (ASN)
ATTEMPTS (PRT)
ATTRIBUTES (ATT)
CLUSTER (ASN)
CODE (PRT)
CONTROLPW (PRT)
DATA (ASN)
entryname (HIS)
HISTORY (HIS)
CREATION (HIS)
DATASET-OWNER (HIS)
EXPIRATION (HIS)
RELEASE (HIS)
INDEX (ASN)
MASTERPW (PRT)
NOUPDATE (ATT)
PROTECTION (PRT)
RACF (PRT)
READPW (PRT)
UPDATE (ATT)
UPDATEPW (PRT)
USAR (PRT)
USVR (PRT)

## User Catalog Entry Keywords

ALIAS (ASN)
ASSOCIATIONS (ASN)
DEVTYPE(VLS)
entryname (HIS)
HISTORY (HIS)
    RELEASE (HIS)
SMSDATA
    DATACLASS
    MANAGEMENTCLASS
    STORAGECLASS
    LBACKUP
VOLFLAG (VLS)
VOLSER (VLS)

## Description of Keyword Fields

This section contains a description of each field name. The field names are in the following groups of related information:

| Abbreviations | Group Names |
|---|---|
| **ALC** | allocation group |
| **ASN** | associations group |
| **ATT** | attributes group |
| **GDG** | generation data group base entry, special fields |
| **HIS** | history group |
| **NVS** | non-VSAM entry, special field |
| **PRT** | protection group |
| **STA** | statistics group |
| **VLS** | volumes group. |

Groups are in alphabetic order. Field names within each group are in alphabetic order, not the order of appearance in the listed entry.

## ALC: Allocation Group

The fields in this group describe the space allocated to the data or index component defined by the entry.

**HI-A-RBA**—The highest RBA (plus 1) available within allocated space to store data.

**HI-U-RBA**—The highest RBA (plus 1) within allocated space that actually contains data. (The RBA of the next completely unused control interval.)

**SPACE-PRI**—Gives the number of units (indicated under TYPE) of space allocated to the data or index component when the cluster was defined. This amount of space is to be allocated whenever a data component, a key range within the data component, or the data component's associated sequence set (if IMBED is an attribute of the cluster) is extended onto a candidate volume.

**SPACE-SEC**—Gives the number of units (indicated under TYPE) of space to be allocated whenever a data set (or key range within it) is extended on the same volume.

**SPACE-TYPE**—Indicates the unit of space allocation:
**CYLINDER**—Cylinders

**KILOBYTE**—Kilobytes
**MEGABYTE**—Megabytes
**TRACK**—Tracks

## ASN: Associations Group

This group lists the type (cluster or data, for example) and entry names of the objects associated with the present entry. A cluster or alternate index entry will indicate its associated path entries and data and index (if a key-sequenced data set) entries. Similarly, an index or data entry will indicate its associated cluster or the alternate index of which it is a component.

- An alias entry points to:
  - Its associated non-VSAM data set entry. If the associated entry is a symbolic association (for example, defined with the SYMBOLICRELATE keyword), both the unresolved and resolved values will be listed. The resolved value will be the value resulting from using the symbols defined on the system on which the LISTCAT is run.
  - A user catalog entry. (All alias entries for a non-VSAM data set entry are chained together, as are alias entries for a user catalog entry.)
- An alternate index entry points to:
  - Its associated data and index entries.
  - Its base cluster's cluster entry.
  - Each associated path entry.
- An alternate index's data entry points to:
  - Its associated alternate index entry.
- An alternate index's index entry points to:
  - Its associated alternate index entry.
- A cluster entry points to:
  - Its associated data entry.
  - Each associated path entry.
  - For a key-sequenced cluster, its associated index entry.
  - For a cluster with alternate indexes, each associated alternate index entry.
- A cluster's data entry points to:
  - Its associated cluster entry.
- A cluster's index entry points to:
  - Its associated cluster entry.
- A generation data group base entry points to:
  - Its associated non-VSAM data set entries.
- A non-VSAM data set entry points to:
  - Its associated alias entry.
  - Its associated generation data group (for a G0000V00 non-VSAM).
- A page space entry points to:
  - Its associated data entry. The page space is cataloged as an entry-sequenced cluster with a cluster entry and an associated data entry.
- A path entry that establishes the connection between a base cluster and an alternate index points to:
  - Its associated alternate index entry, and the alternate index's associated data and index entries.
  - The data entry of its associated base cluster.
  - For a key-sequenced base cluster, the index entry of its associated base cluster.
- Path entry that is an alias for a cluster entry points to:

  - Its associated base cluster entry.
  - The data entry of its associated base cluster.
  - For a key-sequenced cluster, the index entry of its associated base cluster.
- A user catalog entry points to:
  - Its associated alias entry.

**AIX**—Identifies an alternate index entry. **ALIAS**—Identifies an alias entry. **CLUSTER**—Identifies a cluster entry. **DATA**—Identifies a data entry. **GDG**—Identifies a generation data group (GDG) base entry. **INDEX**—Identifies an index entry. **NONVSAM**—Identifies a non-VSAM data set entry. **PGSPC**—Identifies a page space entry. **PATH**—Identifies a path entry. **UCAT**—Identifies a user catalog entry.

## ATT: Attributes Group

The fields in this group describe the miscellaneous attributes of the entry. See the DEFINE command for further discussion of most of these attributes.

**ACT-DIC-TOKEN**—The active dictionary token or NULL. This attribute is only valid for compressed data sets.

**Note:** The following information is not an intended programming interface. It is provided for diagnostic purposes only.

The first byte of the dictionary token indicates the type of compression used for the data set.

**X'100. ....'** indicates compression has been rejected for the data set. No data is compressed.
**X'010. ....'** indicates generic DBB compression is used.
**X'011. ....'** indicates tailored compression is used.

**AVGLRECL**—The average length of data records, in bytes. AVGLRECL equals MAXLRECL when the records are fixed length. Do not, however, set AVGLRECL equal to MAXLRECL for variable-length relative records.

**Note:** For variable-length RRDSs, the AVLGLRECL shown in the LISTCAT output is 4 greater than the user-specified length, reflecting the system-increased record size.

**AXRKP**—Indicates, for an alternate index, the offset, from the beginning of the base cluster's data record, at which the alternate-key field begins.

**BUFND**—The number of buffers provided for catalog data records. The default for BUFND is taken at ICF catalog open and is not reflected in the output from LISTCAT.

**BUFNI**—The number of buffers provided for catalog index records. The default for BUFNI is taken at ICF catalog open and is not reflected in the output from LISTCAT.

**BUFSPACE**—The minimum buffer space, in bytes, in virtual storage to be provided by a processing program.

**CCSID**—The Coded Character Set Identifier attribute which identifies a specific set of encoding scheme identifier, character set identifier(s), code page identifier(s) or the additional coding required to uniquely identify the coded graphic used.

**CI/CA**—The number of control intervals per control area.

**CISIZE**—The size of a control interval, in bytes.

**COMP-FORMT**—The data is written to the data set in a format that allows data compression.

**ECSHARING**—Sharing via the coupling facility for this catalog is allowed.

**ERASE**—Records are to be erased (set to binary zeros) when deleted.

**EXCPEXIT**—The name of the object's exception exit routine.

**EXT-ADDR**—Extended addressability indicator.

**EXTENDED**—The extended format indicator.

**ICFCATALOG**—The object is part of a cluster for the catalog data set.

**INDEXED**—The data component has an index; it is key-sequenced.

**INH-UPDATE**—The data component cannot be updated. Either the data component was exported with INHIBITSOURCE specified, or its entry was modified by way of ALTER, with INHIBIT specified.

**KEYLEN**—The length of the key field in a data record, in bytes.

**LINEAR**—The cluster is a linear data set.

**MAXLRECL**—The maximum length of data or index records, in bytes. MAXLRECL equals AVGLRECL when the records are fixed length. Do not, however, set MAXLRECL equal to AVGLRECL for variable-length relative records.

**Note:** For variable-length RRDSs, the MAXLRECL shown in the LISTCAT output is 4 greater than the user-specified length, reflecting the system-increased record size.

**MAXRECS**—Identifies the highest possible valid relative record number, for a relative record data set. This value is calculated as follows: 2 to the 32nd power/CISIZE x number of records slots per control interval

**NOECSHARE**—Sharing via the coupling facility for this catalog is not allowed.

**NOERASE**—Records are not to be erased (set to binary 0's) when deleted.

**NONINDEXED**—The data component has no index; it is entry-sequenced.

**NONSPANNED**—Data records cannot span control intervals.

**NONUNIQKEY**—Indicates, for an alternate index, that more than one data record in the base cluster can contain the same alternate-key value.

**NOREUSE**—The data set cannot be reused.

**NOSWAP**—The page space is a conventional page space and cannot be used as a high speed swap data set.

**NOTRKOVFL**—The physical blocks of a page space data set cannot span a track boundary.

**NOUPDATE**—When the path is opened for processing, its associated base cluster is opened but the base cluster's upgrade set is not opened.

**NOUPGRADE**—The alternate index is not upgraded unless it is opened and being used to access the base cluster's data records.

**NOTUSABLE**—The entry is not usable because (1) the catalog could not be correctly recovered by RESETCAT, or (2) a DELETE SPACE FORCE was issued for a volume in the entry's volume list.

**NOWRITECHK**—Write operations are not checked for correctness.

**NUMBERED**—The cluster is a relative record data set.

**RECORDS/CI**—Specifies the number of records, or slots, in each control interval of a relative record data set.

**RECOVERY**—A temporary CLOSE is issued as each control area of the data set is loaded, so the whole data set will not have to be reloaded if a serious error occurs during loading.

**REUSE**—The data set can be reused (that is, its contents are temporary and its high-used RBA can be reset to 0 when it is opened).

**RKP**—The relative key position:the displacement from the beginning of a data record to its key field.

**SHROPTNS**—(n,m) The numbers n and m identify the types of sharing permitted. See SHAREOPTIONS in the DEFINE CLUSTER section for more details.

**SIZES-VALID**—Indicates if user data sizes are valid (YES) or are not valid (NO).

**SPANNED**—Data records can be longer than control interval length, and can cross, or span, control interval boundaries.

**SPEED**—CLOSE is not issued until the data set has been loaded.

**STRIPE-COUNT**—The number of stripes for the data set. This number will always be 1 for extended format VSAM KSDS.

**STRNO**—The number of concurrent RPLs the catalog is prepared to accommodate. The default for STRNO is taken at ICF catalog open and is not reflected in the output from LISTCAT.

**Note:** LISTCAT ALL indicates a value of 0 (the default) if no other value is specified, rather than the expected value of 2.

**SWAP**—The page space is a high speed swap data set used by Auxiliary Storage Management during a swap operation to store and retrieve the set of LSQA pages owned by an address space.

**TEMP-EXP**—The data component was temporarily exported.

**TRKOVFL**—The physical blocks of a page space data set can span a track boundary.

**UNIQUEKEY**—Indicates, for an alternate index, that the alternate-key value identifies one, and only one, data record in the base cluster.

**UPDATE**—When the path is opened, the upgrade set's alternate indexes (associated with the path's base cluster) are also opened and are updated when the base cluster's contents change.

**UPGRADE**—When the alternate index's base cluster is opened, the alternate index is also opened and is updated to reflect any changes to the base cluster's contents.

**WRITECHECK**—Write operations are checked for correctness.

## GDG: Generation Data Group Base Entry, Special Fields

The special fields for a generation data group base entry describe attributes of the generation data group.

**ATTRIBUTES**
This field includes the following fields:

**EMPTY**
All generation data sets in the generation data group are uncataloged when the maximum number (given under LIMIT) is reached and one more data set is to added to the group.

**LIMIT**
The maximum number of generation data sets allowed in the generation data group.

**NOEMPTY**
Only the oldest generation data set in the generation data group is uncataloged when the maximum number (given under LIMIT) is reached and one more data set is to be added to the group.

**NOSCRATCH**
Generation data sets are not to be scratched (see SCRATCH below) when uncataloged.

**SCRATCH**
Generation data sets are to be scratched (that is, the DSCB describing each one is removed from the VTOC of the volume where it resides) when uncataloged.

## NVS: Non-VSAM Entry, Special Field

The special field for a non-VSAM data set describes a non-VSAM data set stored on magnetic tape.

**FSEQN**—The sequence number (for the tape volume indicated under the "VOLUMES group" keyword VOLSER) of the file in which the non-VSAM data set is stored.

## HIS: History Group

The fields in this group identify the object's owner and give the object's creation and expiration dates.

**entryname**—The name of the cataloged object. The entryname can be specified with the ENTRIES parameter of LISTCAT to identify a catalog entry.

**HISTORY**—This field includes the following fields:

**CREATION**—The Julian date (YYYY.DDD) on which the entry was created.

**DATASET-OWNER**—Contents of the data set owner field in the BCS. Was formerly called **OWNER-IDENT** field.

**EXPIRATION**—The Julian date (YYYY.DDD) on or after which the entry can be deleted without specifying the PURGE parameter in the DELETE command. Julian dates 1999.365, 1999.366, and 9999.999 indicate that PURGE is always required to delete the object.

**RELEASE**—The release of VSAM under which the entry was created:

1 = OS/VS2 Release 3 and releases preceding Release 3

2 = OS/VS2 Release 3.6 and any later releases

**STATUS**—The possible values this field can contain are active, deferred, library, or rolled-off.

For generation data set entries, the status is indicated by active, deferred, or rolled-off.

For non-VSAM entries, a status of library indicates a partitioned data set extended (PDSE).

**OAMDATA**—For OAM entries, this field contains the following:

– **DIRECTORYTOKEN**—The OAM directory token (1-to-8 characters).

**RLSDATA**—For RLS/Recovery entries, this field has the following:

– **FRLOG**—Gives the value of the FRLOG parameter specified on the DEFINE CLUSTER.

– **LOG**—Gives the value of the LOG parameter specified on DEFINE CLUSTER.

– **LOGSTREAMID**—This gives you the value of the LOGSTREAMID parameter specified on the DEFINE CLUSTER.

– **RECOVERY REQUIRED**—Indicates whether the sphere is currently in the process of being forward recovered.

– **RECOVERY TIMESTAMP**—This gives the time the most recent backup was taken when the data set was accessed by CICS using VSAM RLS.

– **RLS IN USE**—Indicates whether the sphere is using RLS. A sphere uses RLS if:

- It was last opened for RLS processing.
- It is not opened for RLS processing but is recoverable and either has retained locks protecting updates or is in a lost locks state. For further information, see *CICS Recovery and Restart Guide*.

– **VSAM QUIESCED**—Indicates that the sphere has been quiesced for RLS. You cannot open the sphere using RLS.

**SMSDATA**—For SMS-managed data sets, this field has:

– **BWO**—Data set is enabled for backup-while-open.

– **BWO STATUS**—Indicates the status of the data set. Status can be:

- Data set is enabled for backup-while-open
- Control interval or control area split in progress
- Data set has been restored and is down level. It might need to be updated with forward recovery logs.

– **BWO TIMESTAMP**—A CICS timestamp that indicates the time from which forward recovery logs have to be applied to a restored copy of the data set.

– **STORAGECLASS**—The name of the storage class assigned to the cluster.

- **MANAGEMENTCLASS**—The name of the management class assigned to the cluster.
- **DATACLASS**—The name of the data class assigned to the cluster.
- **LBACKUP**—The last date that the cluster was backed up. If this date is unavailable, this field will contain an entry of all “Xs” rather than an actual date.

## PRT: Protection Group

The fields in this group describe how the alternate index, cluster, data component, index component, or path defined by the entry is password-protected or RACF protected. NULL or SUPPRESSED might be listed under password protection and YES or NO might be listed under RACF protection.

**NULL** indicates that the object defined by the entry has no passwords.

**SUPP** indicates that the master password of neither the catalog nor the entry was specified, so authority to list protection information is not granted.

**RACF**—Indicates if the entry is protected by the Resource Access Control Facility.
YES—Entry is RACF protected.
NO—Entry is not RACF protected.

**ATTEMPTS**—Gives the number of times the console operator is allowed to try to enter a correct password.

**CODE**—Gives the code used to tell the console operator which alternate index, catalog, cluster, path, data component, or index component requires a password to be entered. NULL is listed under CODE if a code is not used—the object requiring the password is identified with its full name.

**CONTROLPW**—The control interval password (that is, the password for control interval access). NULL indicates no control interval password.

**MASTERPW**—The master password.

**READPW**—The read-only password. NULL indicates no read-only password.

**UPDATEPW**—The update password. NULL indicates no update password.

**USAR**—The contents (1-to-255 bytes, in character format) of the USAR (user-security-authorization record). This is the information specified in the string subparameter of the AUTH subparameter of the DEFINE command.

**USVR**—The name of the USVR (user-security-verification routine) that is to be invoked to verify authorization of any access to the entry.

## STA: Statistics Group

The fields in this group give numbers and percentages that tell how much activity has taken place in the processing of a data or index component. The statistics in the catalog are updated when the data set is closed. Therefore, if an error occurs during CLOSE, the statistics might not be valid.

Statistic values can be any value from a minus to a large number, if an error occurs before or during close processing. VERIFY will not correct these statistics. To

correct these statistics, you can either use the EXPORT and IMPORT commands, or you can use REPRO to copy this information into a new data set. When using compressed VSAM data sets, a REPRO with the REPLACE option should be used to correct the statistics.

**COMP-USER-DATA-SIZE**—The total length of data after compression. If the data lengths are too large to be presented in decimal, they are presented in hexadecimal format.

**FREESPACE-%CI**—Percentage of space to be left free in a control interval for subsequent processing.

**FREESPACE-%CA**—Percentage of control intervals to be left free in a control area for subsequent processing.

**FREESPC**—Actual number of bytes of free space in the total amount of space allocated to the data or index component. Free space in partially used control intervals is not included in this statistic. Some of this space may not be accessible due to the current amount of key compression that can be performed in the index.

**INDEX**—This field appears only in an index entry. The fields under it describe activity in the index component.

**ENTRIES/SECT**—The number of entries in each section of entries in an index record.

**HI-LEVEL-RBA**—The RBA (relative byte address) of the highest-level index record.

**LEVELS**—The number of levels of records in the index. The number is 0 if no records have been loaded into the key-sequenced data set to which the index belongs.

**SEQ-SET-RBA**—The RBA (relative byte address), in decimal, of the first sequence-set record. The sequence set can be separated from the index set by some amount of RBA space.

The remaining fields in the statistics group (except for the system timestamp field), are updated only when the data set is closed.

**EXCPS**—EXCP (run channel program—SVC 0) macro instructions issued by VSAM against the data or index component.

**EXTENTS**—Extents in the data or index component.

**REC-DELETED**—The number of records that have been deleted from the data or index component. Statistics for records deleted are not maintained when the data set is processed in control interval mode.

**REC-INSERTED**—For a key-sequenced data set, the number of records that have been inserted into the data component before the last record; records originally loaded and records added to the end are not included in this statistic. For relative record data sets, it is the number of records inserted into available slots; the number of records originally loaded are included in this statistic. Statistics for records inserted are not maintained when the data set is processed in control interval mode.

**REC-RETRIEVED**—The number of records that have been retrieved from the data or index component, whether for update or not for update. Statistics for records retrieved are not maintained when the data set is processed in control interval mode.

**REC-TOTAL**—The total number of records actually in the data or index component. This statistic is not maintained when the data set is processed in control interval mode. For a variable-length RRDS, this is the count of slots in the data set.

**REC-UPDATED**—The number of records that have been retrieved for update and rewritten. This value does not reflect those records that were deleted, but a record that is updated and then deleted is counted in the update statistics. Statistics for records updated are not maintained when the data set is processed in control interval mode.

**SPLITS-CA**—Control area splits. Half the data records in a control area were written into a new control area and then were deleted from the old control area.

**SPLITS-CI**—Control interval splits. Half the data records in a control interval were written into a new control interval and then were deleted from the old control interval.

**SYSTEM-TIMESTAMP**—The time (system time-of-day clock value) the data or index component was last closed (after being opened for operations that might have changed its contents).

**USER-DATA-SIZE**—Displays the total length of data before compression. If the data lengths are too large to be presented in decimal, they are presented in hexadecimal format.

## VLS: Volumes Group

The fields in this group identify the volume on which a data component, index component, user catalog, or non-VSAM data set is stored. It also identifies candidate volumes for a data or index component. The fields describe the type of volume and give, for a data or index component, information about the space the object uses on the volume.

- If an entry-sequenced or relative record cluster's data component has more than one VOLUMES group, each group describes the extents that contain data records for the cluster on a specific volume.
- If a key-sequenced cluster's data component has more than one VOLUMES group, each group describes the extents that contain data records for the cluster, or one of its key ranges, on a specific volume.
- If a key-sequenced cluster's index component has more than one VOLUMES group, each group describes the extents that contain index records for the cluster, or one of its key ranges, on a specific volume. The first VOLUMES group describes the extent that contains the high-level index records (that is, index records in levels above the sequence set level). Each of the next groups describes the extents that contain sequence-set index records for the cluster, or one of its key ranges, on a specific volume. The index component of a key-sequenced data set with the IMBED attribute will have a minimum of two volume groups, one for the embedded sequence set, and one for the high-level index. The extents for the embedded sequence set are the same as those for the data component.

**BYTES/TRACK**—The number of bytes that VSAM can write on a track (listed for page spaces only).

**DEVTYPE**—The type of device to which the volume belongs.

**EXTENT-NUMBER**—The number of extents allocated for the data or index component on the volume.

**EXTENT-TYPE**—The type of extents:
- **00**—The extents are contiguous.
- **40**—The extents are not preformatted.
- **80**—A sequence set occupies a track adjacent to a control area.
- **FF**—A candidate volume.

**EXTENTS**—Gives the physical and relative byte addresses of each extent.
- **HIGH-CCHH**—The device address (that is, CC = cylinder and HH = track) of the end of the extent.
- **HIGH-RBA**—The RBA (relative byte address), in decimal, of the end of the extent.
- **LOW-CCHH**—The device address (that is, CC = cylinder and HH = track) of the beginning of the extent.
- **LOW-RBA**—The RBA (relative byte address), in decimal, of the beginning of the extent.
- **TRACKS**—The number of tracks in the extent, from low to high device addresses.

**HIGH-KEY** [5]—For a key-sequenced data set with the KEYRANGE attribute, the highest hexadecimal value allowed on the volume in the key field of a record in the key range. A maximum of 64 bytes can appear in HIGH-KEY.

**HI-KEY-RBA** [5]—For a key-sequenced data set, the RBA (relative byte address), in decimal, of the control interval on the volume that contains the highest keyed record in the data set or key range.

**LOW-KEY** [5]—For a key-sequenced data set with the KEYRANGE attribute, the lowest hexadecimal value allowed on the volume in the key field of a record in the key range. A maximum of 64 bytes can appear in LOW-KEY.

**PHYRECS/TRK**—The number of physical records (of the size indicated under PHYRECS-SIZE) that VSAM can write on a track on the volume.

**PHYREC-SIZE**—The number of bytes that VSAM uses for a physical record in the data or index component.

**HI-A-RBA**—The highest RBA (plus 1) available within allocated space to store data component, its key range, the index component, or the sequence set records of a key range.

**HI-U-RBA**—The highest RBA (plus 1) within allocated space that actually contains data component, its key range, the index component, or the sequence set records of a key range. (The RBA of the next completely unused control interval.)

5. Multiple key ranges can reside on a single volume; the volumes group is repeated for each such key range field.

**TRACKS/CA**—The number of tracks in a control area in the data component. (This value is computed when the entry is defined. This value reflects the optimum size of the control area for the given device type and the nature of the entry, whether indexed, nonindexed, or numbered.) For a key-sequenced data set with the imbedded attribute, this value includes the sequence set track.

**VOLFLAG**—Indicates if the volume is a candidate volume and if the volume is a prime or overflow volume on which data in a given key range is stored.

**CANDIDATE**—The volume is a candidate for storing the data or index component.

**CAND-SPACE**—The volume is a candidate for storing the data or index component, and it has a primary extent preallocated (the data set was defined with a guaranteed-space storage class).

**OVERFLOW**—The volume is an overflow volume on which data records in a key range are stored. The KEYRANGE begins on another (PRIME) volume.

**PRIME**—The volume is the first volume on which data records in a key range are stored.

**VOLSER**—The serial number of the volume.

## Device Type Translate Table

The following table lists the LISTCAT codes for supported device types.

*Table 13. Device Type Translate Table*

| Generic Name | LISTCAT Code | Device Type |
|---|---|---|
| 3380 | 3010 200E | 3380, all models |
| 3390 | 3010 200F | 3390, all models |
| 9345 | 3010 2004 | 9345, all models |
| 3400-2 | 30C0 8003 | 3420 Models 3, 5, and 7 |
| 3400-5 | 3200 8003 | 3420 Models 4, 6, and 8 (9 track, 6250 BPI) |
| 3400-6 | 3210 8003 | 3420 Models 4, 6, and 8 (9 track, 1600/6250 BPI) |
| 3400-9 | 3300 8003 | 3420C (3480 compatibility mode) |
| 3400-3 | 3400 8003 | 3430, 9 track, 1600/6250 BPI tape |
| 3480 | 7800 8080 | 3480 Magnetic Tape Unit |
| 3480 | 7800 8080 | 3490 Magnetic Tape Subsystem Models A01, A02, B02, B04, D31, and D32 |
| 3480X | 7804 8080 | 3480 Magnetic Tape Unit with IDRC enabled |
| 3480X | 7804 8080 | 3490 Magnetic Tape Subsystem Models A01, A02, B02, B04, D31, and D32 with IDRC enabled |
| 3490 | 7804 8081 | 3490 Magnetic Tape Subsystem Enhanced Capability Models A10, A20, B20, B40, D41, and D42 |
| 3590-1 | 7804 8083 | IBM 3590 High Performance Tape Subsystem Models A00, 3591 A01, A14, B11, B1A, C12 |

## Examples of LISTCAT Output Listings

This section illustrates the kind of output you can get when you specify LISTCAT parameters. It also describes the job control language you can specify and the output messages you get when the LISTCAT procedure runs successfully.

## Job Control Language (JCL) for LISTCAT Jobs

The job control language (JCL) statements that can be used to list a catalog's entries are:

```
//LISTCAT  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//OUTDD    DD    DSN=LISTCAT.OUTPUT,UNIT=3480,
//         VOL=SER=TAPE10,LABEL=(1,NL),DISP=(NEW,KEEP),
//         DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     LISTCAT -
          CATALOG(YOURCAT) -
          OUTFILE(OUTDD) -
          ...
/*
```

**Note:** Additional keywords can be included.

The JOB statement contains user and accounting information required for your installation.

The EXEC statement identifies the program to be run, IDCAMS (that is, the access method services program).

- OUTDD, which specifies an alternate output file, so that the LISTCAT output can be written onto an auxiliary storage device. The LISTCAT command's OUTFILE parameter points to the OUTDD DD statement. Only the LISTCAT output is written to the alternate output device. JCL statements, system messages, and job statistics are written to the SYSPRINT output device.
  - DSN=LISTCAT.OUTPUT specifies the name for the magnetic tape file.
  - UNIT=3480 and VOL=SER=TAPE10 specifies that the file is to be contained on magnetic tape volume TAPE10.
  - LABEL=(1,NL) specifies that this is the first file on a nonlabeled tape. You can also use a standard labeled tape by specifying LABEL=(1,SL). If subsequent job steps produce additional files of LISTCAT output on the same tape volume, you should increase the file number in each job step's LABEL subparameter (that is, LABEL=(2,NL) for the second job step, LABEL=(3,NL) for the third job step, etc.)
  - DISP=(NEW,KEEP) specifies that this is a new tape file and is to be rewound when the job finishes. If a subsequent job step prints the tape, DISP=(NEW,PASS) should be specified. If your job step contains more than one LISTCAT command, use DISP=(MOD,KEEP) or DISP=(MOD,PASS) to concatenate all of the LISTCAT output in one sequential file.
  - DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629) specifies that the LISTCAT output records are variable-length, blocked 5-to-1, and are preceded by an ANSI print control character.
- SYSPRINT DD, which is required for each access method services job step. It identifies the output queue, SYSOUT=A, on which all LISTCAT output and system output messages are printed (unless the OUTFILE parameter and its associated DD statement is specified—see OUTDD above).

**Note:** If you want *all* output to be written to an auxiliary storage device, replace X'OUTDD' with X'SYSPRINT' in the OUTDD DD statement and omit the SYSPRINT DD SYSOUT=A statement.

- SYSIN DD, which specifies, with an asterisk (*), that the statements that follow are the input data statements. A '/*' ends the input data statements.

### LISTCAT Output

The LISTCAT command parameters shown in the preceding example are common to the LISTCAT examples that follow. Other LISTCAT parameters are coded with each example and the output that results is illustrated. These two parameters are optional:

- CATALOG, which identifies YOURCAT as the catalog whose entries are to be listed.
- OUTFILE, which points to the OUTDD DD statement. The OUTDD DD statement allocates an alternate output file for the LISTCAT output.

If you want to print the LISTCAT output that is contained on an alternate output file, you can use the IEBGENER program. The following shows the JCL required to print the alternate output file, LISTCAT.OUTPUT, that was allocated previously:

```
//PRINTOUT  JOB   ...
//STEP1     EXEC  PGM=IEBGENER
//SYSUT1    DD    DSN=LISTCAT.OUTPUT,UNIT=2400-3,
//          VOL=SER=TAPE10,LABEL=(1,NL),DISP=(OLD,KEEP),
//          DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)
//SYSUT2    DD    SYSOUT=A
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD    DUMMY
/*
```

**Note:** If you have the DFSORT product installed, consider using ICEGENER as an alternative to IEBGENER when making an unedited copy of a data set or member. It is usually faster than IEBGENER. It might already be installed on your system using the name IEBGENER.

## LISTCAT and Access Method Services Output Messages

When the LISTCAT job completes, access method services provides messages and diagnostic information. If an error occurred, an analysis of the error message can be found in *z/OS TSO/E User's Guide*. When your LISTCAT job completes successfully, access method services provides messages that follow the entry listing (see Figure 8):

```
                                                  LISTING FROM CATALOG -- ICFUCAT1

         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------1
                   ALIAS -----------------1
                   CLUSTER ---------------4
                   DATA ------------------5
                   GDG -------------------1
                   INDEX -----------------4
                   NONVSAM ---------------9
                   PAGESPACE -------------0
                   PATH ------------------2
                   SPACE -----------------0
                   USERCATALOG -----------0
                   TOTAL ----------------27

         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0

IDC0002I IDCAMS PROCESSING COMPLETE, MAXIMUM CONDITION CODE WAS 0
```

*Figure 8. Messages That Follow the Entry Listing*

The first line identifies the catalog that contains the listed entries. The next group of lines specify the number of each entry type, and the total number of entries, that were listed. This statistical information can help you determine the approximate size, in records, of your catalog. The next line specifies the number of entries that could not be listed because the appropriate password was not specified. The last two messages indicate that the LISTCAT command (FUNCTION) and the job step (IDCAMS) completed successfully. When LISTCAT is invoked from a TSO terminal, IDC0001I is not printed.

## LISTCAT Output Listing

If you use LISTCAT with no parameters, the entryname and type of each entry are listed (see Figure 9 on page 350). You can use this type of listing to list the name of each cataloged object and to determine the number of entries in the catalog. The total number of entries is an approximate size, in records, of your catalog.

## LISTCAT Output

```
  LISTCAT -
          CATALOG(ICFUCAT1)           /* IN CATALOG ICFUCAT1       */

                              LISTING FROM CATALOG -- ICFUCAT1
CLUSTER ------- 0000000000000000000000000000000000000000000000
   DATA ------- ICFUCAT1
   INDEX ------ ICFUCAT1.CATINDEX
CLUSTER ------- SYS1.VVDS.V1P0101
   DATA ------- SYS1.VVDS.V1P0101
CLUSTER ------- SYS1.VVDS.V1P0201
   DATA ------- SYS1.VVDS.V1P0201
CLUSTER ------- SYS1.VVDS.V1P0202
   DATA ------- SYS1.VVDS.V1P0202
CLUSTER ------- SYS1.VVDS.V1P0301
   DATA ------- SYS1.VVDS.V1P0301
CLUSTER ------- SYS1.VVDS.V1P0302
   DATA ------- SYS1.VVDS.V1P0302
CLUSTER ------- SYS1.VVDS.V338001
   DATA ------- SYS1.VVDS.V338001
ALIAS --------- USER.ALIAS
CLUSTER ------- USER.DUMMY
   DATA ------- USER.DUMMY.CLDATA
   INDEX ------ USER.DUMMY.CLINDEX
GDG BASE ------ USER.GDGBASE
   NONVSAM ---- USER.GDGBASE.G0003V00
   NONVSAM ---- USER.GDGBASE.G0004V00
   NONVSAM ---- USER.GDGBASE.G0005V00
   NONVSAM ---- USER.GDGBASE.G0006V00
NONVSAM ------- USER.GDGBASE.G0001V00
NONVSAM ------- USER.GDGBASE.G0002V00
AIX ----------- USER.KSDS1.AIX1CLUS
   DATA ------- USER.KSDS1.AIX1DATA
   INDEX ------ USER.KSDS1.AIX1INDX
   PATH ------- USER.KSDS1.PATHAIX1
CLUSTER ------- USER.KSDS1.CLUSTER
   DATA ------- USER.KSDS1.CLDATA
   INDEX ------ USER.KSDS1.CLINDEX
   PATH ------- USER.KSDS1.PATHCL
CLUSTER ------- USER.LINEAR
   DATA ------- USER.LINEAR.DATA
NONVSAM ------- USER.MODEL
NONVSAM ------- USER.NONVSAM.DATA.SET
NONVSAM ------- USER.PDSE
CLUSTER ------- USER.SPANNED.CLUSTER
   DATA ------- USER.SPANNED.DATA
   INDEX ------ USER.SPANNED.INDEX

       THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------1
                   ALIAS -----------------1
                   CLUSTER --------------11
                   DATA ----------------12
                   GDG -------------------1
                   INDEX ----------------5
                   NONVSAM ---------------9
                   PAGESPACE -------------0
                   PATH ------------------2
                   SPACE ----------------0
                   USERCATALOG -----------0
                   TOTAL ----------------42
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

*Figure 9. Example of LISTCAT Output When No Parameters Are Specified*

## LISTCAT NAMES Output Listing

When you specify LISTCAT with the NAMES parameter, only the entryname and type of each entry are listed (see Figure 10). In this example, LEVEL(USER) limits the data sets that are listed by name to those with a high-level qualifier of USER.

```
  LISTCAT -
          LEVEL(USER)                /* LIST ALL 'USER' ENTRIES   */ -
          NAME                       /* NAME INFORMATION ONLY     */ -
          CATALOG(ICFUCAT1)          /* IN CATALOG ICFUCAT1       */

                            LISTING FROM CATALOG -- ICFUCAT1
ALIAS --------- USER.ALIAS
CLUSTER ------- USER.DUMMY
DATA ---------- USER.DUMMY.CLDATA
INDEX --------- USER.DUMMY.CLINDEX
GDG BASE ------ USER.GDGBASE
NONVSAM ------- USER.GDGBASE.G0003V00
NONVSAM ------- USER.GDGBASE.G0004V00
NONVSAM ------- USER.GDGBASE.G0005V00
NONVSAM ------- USER.GDGBASE.G0006V00
NONVSAM ------- USER.GDGBASE.G0001V00
NONVSAM ------- USER.GDGBASE.G0002V00
AIX ----------- USER.KSDS1.AIX1CLUS
DATA ---------- USER.KSDS1.AIX1DATA
INDEX --------- USER.KSDS1.AIX1INDX
DATA ---------- USER.KSDS1.CLDATA
INDEX --------- USER.KSDS1.CLINDEX
CLUSTER ------- USER.KSDS1.CLUSTER
PATH ---------- USER.KSDS1.PATHAIX1
PATH ---------- USER.KSDS1.PATHCL
CLUSTER ------- USER.LINEAR
DATA ---------- USER.LINEAR.DATA
NONVSAM ------- USER.MODEL
NONVSAM ------- USER.NONVSAM.DATA.SET
NONVSAM ------- USER.PDSE
CLUSTER ------- USER.SPANNED.CLUSTER
DATA ---------- USER.SPANNED.DATA
INDEX --------- USER.SPANNED.INDEX
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------1
                   ALIAS -----------------1
                   CLUSTER ---------------4
                   DATA ------------------5
                   GDG -------------------1
                   INDEX -----------------4
                   NONVSAM ---------------9
                   PAGESPACE -------------0
                   PATH ------------------2
                   SPACE -----------------0
                   USERCATALOG -----------0
                   TOTAL ----------------27
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

*Figure 10. Example of LISTCAT NAME Output*

## LISTCAT VOLUME Output Listing

When the LISTCAT command is specified with the VOLUME parameter, the volume serial number and device type of each volume that contains part or all of the cataloged object are listed (see Figure 11 on page 352).

```
  LISTCAT -
          LEVEL(USER)                 /* LIST ALL 'USER' ENTRIES   */ -
          VOLUME                      /* VOLUME INFORMATION        */ -
          CLUSTER                     /* INCLUDE CLUSTERS          */ -
          DATA                        /*   AND DATA COMPONENTS     */ -
          INDEX                       /*   AND INDEX COMPONENTS    */ -
          ALTERNATEINDEX              /*   AND ALTERNATEINDEXES    */ -
          PATH                        /*   AND PATHS               */ -
          GENERATIONDATAGROUP         /*   AND GDG BASES           */ -
          NONVSAM                     /*   AND NONVSAM DATA SETS   */ -
          CATALOG(ICFUCAT1)           /* IN CATALOG ICFUCAT1       */

                              LISTING FROM CATALOG -- ICFUCAT1
CLUSTER ------- USER.DUMMY
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1992.345
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
       BWO-------------  (NULL)
     RLSDATA
       LOG --------------(NULL)     RECOVERY REQUIRED --(NO)
       VSAM QUIESCED      (NO)      RLS IN USE ---------(NO)
       LOGSTREAMID ----------------------------------(NULL)
       RECOVERY TIMESTAMP LOCAL-----X'0000000000000000'
       RECOVERY TIMESTAMP GMT-------X'0000000000000000'
DATA ---------- USER.DUMMY.CLDATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1994.215
       RELEASE----------------2     EXPIRATION------0000.000
       ACCOUNT-INFO-----------------------ALTER ACCOUNT INFO
     VOLUMES
       VOLSER------------1P0201     DEVTYPE------X'3010200E'
INDEX --------- USER.DUMMY.CLINDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1994.221
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------1P0201     DEVTYPE------X'3010200E'
GDG BASE ------ USER.GDGBASE
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       ACCOUNT-INFO-----------------------------------(NULL)
NONVSAM ------- USER.GDGBASE.G0003V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       ACCOUNT-INFO-----------------------------------(NULL)
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'
```

*Figure 11. Example of LISTCAT VOLUME Output (Part 1 of 5)*

```
NONVSAM ------- USER.GDGBASE.G0004V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       ACCOUNT-INFO-----------------------------------(NULL)
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'
NONVSAM ------- USER.GDGBASE.G0005V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       ACCOUNT-INFO-----------------------------------(NULL)
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'
NONVSAM ------- USER.GDGBASE.G0006V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       ACCOUNT-INFO-----------------------------------(NULL)
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'
NONVSAM ------- USER.GDGBASE.G0001V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       ACCOUNT-INFO-----------------------------------(NULL)
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'
NONVSAM ------- USER.GDGBASE.G0002V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       ACCOUNT-INFO-----------------------------------(NULL)
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'
AIX ----------- USER.KSDS1.AIX1CLUS
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
       SMS MANAGED--------(YES)
```

*Figure 11. Example of LISTCAT VOLUME Output (Part 2 of 5)*

```
DATA ---------- USER.KSDS1.AIX1DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     VOLUMES
       VOLSER------------1P0201     DEVTYPE------X'3010200E'
       VOLSER------------1P0202     DEVTYPE------X'3010200E'
INDEX --------- USER.KSDS1.AIX1INDX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     VOLUMES
       VOLSER------------1P0201     DEVTYPE------X'3010200E'
       VOLSER------------1P0202     DEVTYPE------X'3010200E'
DATA ---------- USER.KSDS1.CLDATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     VOLUMES
       VOLSER------------1P0201     DEVTYPE------X'3010200E'
       VOLSER------------1P0202     DEVTYPE------X'3010200E'
INDEX --------- USER.KSDS1.CLINDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     VOLUMES
       VOLSER------------1P0201     DEVTYPE------X'3010200E'
       VOLSER------------1P0202     DEVTYPE------X'3010200E'
CLUSTER ------- USER.KSDS1.CLUSTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
       BWO---------------(NULL)
     RLSDATA
       LOG --------------(NULL)     RECOVERY REQUIRED --(NO)
       VSAM QUIESCED      (NO)      RLS IN USE ---------(NO)
       LOGSTREAMID ----------------------------------(NULL)
       RECOVERY TIMESTAMP LOCAL-----X'0000000000000000'
       RECOVERY TIMESTAMP GMT-------X'0000000000000000'
PATH ---------- USER.KSDS1.PATHAIX1
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
PATH ---------- USER.KSDS1.PATHCL
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
```

*Figure 11. Example of LISTCAT VOLUME Output (Part 3 of 5)*

```
CLUSTER ------- USER.LINEAR
       HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
       BWO---------------(NULL)
     RLSDATA
       LOG --------------(NULL)     RECOVERY REQUIRED --(NO)
       VSAM QUIESCED      (NO)      RLS IN USE ---------(NO)
       LOGSTREAMID ------------------------------------(NULL)
       RECOVERY TIMESTAMP LOCAL-----X'0000000000000000'
       RECOVERY TIMESTAMP GMT-------X'0000000000000000'
DATA ---------- USER.LINEAR.DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------9999.999
     VOLUMES
       VOLSER------------338001     DEVTYPE------X'3010200E'
NONVSAM ------- USER.MODEL
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------338001     DEVTYPE------X'3010200E'
NONVSAM ------- USER.NONVSAM.DATA.SET
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---S1P01S02     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0101     DEVTYPE------X'3030200E'
NONVSAM ------- USER.PDSE
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.244
     SMSDATA
       STORAGECLASS ---S1P03S01     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0302     DEVTYPE------X'3030200E'
CLUSTER ------- USER.SPANNED.CLUSTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1992.345
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
       BWO-------------  (NULL)
     RLSDATA
       LOG --------------(NULL)     RECOVERY REQUIRED --(NO)
       VSAM QUIESCED      (NO)      RLS IN USE ---------(NO)
       LOGSTREAMID ------------------------------------(NULL)
       RECOVERY TIMESTAMP LOCAL-----X'0000000000000000'
       RECOVERY TIMESTAMP GMT-------X'0000000000000000'
```

*Figure 11. Example of LISTCAT VOLUME Output (Part 4 of 5)*

```
DATA ---------- USER.SPANNED.DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
     VOLUMES
       VOLSER------------1P0301     DEVTYPE------X'3010200E'
       VOLSER------------1P0302     DEVTYPE------X'3010200E'
INDEX --------- USER.SPANNED.INDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
     VOLUMES
       VOLSER------------1P0301     DEVTYPE------X'3010200E'
       VOLSER-----------------*     DEVTYPE------X'3010200E'
       VOLSER------------1P0301     DEVTYPE------X'3010200E'
       VOLSER------------1P0302     DEVTYPE------X'3010200E'
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------1
                   ALIAS -----------------0
                   CLUSTER ---------------4
                   DATA ------------------5
                   GDG -------------------1
                   INDEX -----------------4
                   NONVSAM ---------------9
                   PAGESPACE -------------0
                   PATH ------------------2
                   SPACE -----------------0
                   USERCATALOG -----------0
                   TOTAL ----------------26
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

*Figure 11. Example of LISTCAT VOLUME Output (Part 5 of 5)*

## LISTCAT ALL Output Listing

When you specify the LISTCAT command and include the ALL parameter, all the information for each catalog entry is listed (see Figure 12 on page 357). This example illustrates the LISTCAT output for each type of catalog entry. You can use this type of listing to obtain all cataloged information (except password and security information) about each entry that is listed. When you want to list an entry's passwords, you must provide the catalog's master password (which results in listing the passwords of each password-protected entry) or each entry's master password.

**Note:** When ENTRIES is specified, you specify only those entrynames that identify catalog entries which are not volume entries. If a volume serial number is specified with the ENTRIES parameter, then entrynames of other entry types cannot also be specified. However, if the ENTRIES parameter is not specified and if entry types are not specified (that is, CLUSTER, SPACE, DATA, etc.), all entries in the catalog, including volume entries, are listed.

```
  LISTCAT -
          LEVEL(USER)                 /* LIST ALL 'USER' ENTRIES   */ -
          ALL                         /* SHOW ALL INFORMATION      */ -
          CATALOG(ICFUCAT1)           /* IN CATALOG ICFUCAT1       */

                              LISTING FROM CATALOG -- ICFUCAT1
ALIAS --------- USER.ALIAS
     HISTORY
       RELEASE----------------2
     ASSOCIATIONS
       NONVSAM--USER.NONVSAM.DATA.SET
CLUSTER ------- USER.DUMMY
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
       BWO STATUS------00000000     BWO TIMESTAMP---00000 00:00:00:0
       BWO               (NULL)
       BWO-------------TYPECICS
     RLSDATA
       LOG --------------(NULL)     RECOVERY REQUIRED --(NO)
       VSAM QUIESCED      (NO)      RLS IN USE ---------(NO)
       LOGSTREAMID ----------------------------------(NULL)
       RECOVERY TIMESTAMP LOCAL-----X'00000000000000000000'
       RECOVERY TIMESTAMP GMT-------X'00000000000000000000'
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       DATA-----USER.DUMMY.CLDATA
       INDEX----USER.DUMMY.CLINDEX
DATA ---------- USER.DUMMY.CLDATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
       ACCOUNT-INFO-----------------------------------(NULL)
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER--USER.DUMMY
     ATTRIBUTES
       KEYLEN-----------------4     AVGLRECL------------2000     BUFSPACE------------6656     CISIZE--------------2048
       RKP--------------------0     MAXLRECL------------2000     EXCPEXIT----------(NULL)     CI/CA----------------270
       AXRKP------------------0
       STRIPE-COUNT------(NULL)
       ACT-DIC-TOKEN----X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
       SHROPTNS(1,3)   RECOVERY     UNIQUE           NOERASE     INDEXED       NOWRITECHK     NOIMBED       NOREPLICAT
       UNORDERED        NOREUSE     NONSPANNED    NONUNIQKEY     TEXT
       CCSID----------------37     CECP EBCDIC
     STATISTICS
       REC-TOTAL--------------0     SPLITS-CI--------------0     EXCPS------------------0
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------0          X'0000000000000000'
       REC-RETRIEVED----------0     FREESPC----------9400320
```

*Figure 12. Example of LISTCAT ALL Output (Part 1 of 11)*

```
     ALLOCATION
       SPACE-TYPE------CYLINDER     HI-A-RBA---------9400320
       SPACE-PRI-------------17     HI-U-RBA---------------0
       SPACE-SEC--------------0
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE---------2048     HI-A-RBA---------400320      EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA--------------0      EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA-------------15
       EXTENTS:
       LOW-CCHH-----X'00020000'     LOW-RBA----------------0     TRACKS---------------255
       HIGH-CCHH----X'0012000E'     HIGH-RBA---------9400319
INDEX --------- USER.DUMMY.CLINDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER--USER.DUMMY
     ATTRIBUTES
       KEYLEN-----------------4     AVGLRECL---------------0     BUFSPACE---------------0     CISIZE--------------2560
       RKP--------------------0     MAXLRECL------------2553     EXCPEXIT----------(NULL)     CI/CA-----------------15
       SHROPTNS(1,3)   RECOVERY     UNIQUE           NOERASE     NOWRITECHK       NOIMBED     NOREPLICAT     UNORDERED
       NOREUSE
     STATISTICS
       REC-TOTAL--------------0     SPLITS-CI--------------0     EXCPS------------------0     INDEX:
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1     LEVELS------------------0
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:            ENTRIES/SECT----------16
       REC-UPDATED------------0     FREESPACE-%CA----------0          X'0000000000000000'     SEQ-SET-RBA------------0
       REC-RETRIEVED----------0     FREESPC------------76800                                  HI-LEVEL-RBA-----------0
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-----------76800
       SPACE-PRI--------------2     HI-U-RBA---------------0
       SPACE-SEC--------------0
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE---------2560     HI-A-RBA-----------76800     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------15     HI-U-RBA---------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA----------------0     TRACKS-----------------2
       HIGH-CCHH----X'0000000C'     HIGH-RBA-----------76799
GDG BASE ------ USER.GDGBASE
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
     ATTRIBUTES
       LIMIT------------------4     NOSCRATCH        NOEMPTY
     ASSOCIATIONS
       NONVSAM--USER.GDGBASE.G0003V00
       NONVSAM--USER.GDGBASE.G0004V00
       NONVSAM--USER.GDGBASE.G0005V00
       NONVSAM--USER.GDGBASE.G0006V00
NONVSAM ------- USER.GDGBASE.G0003V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS------------ACTIVE
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'     FSEQN------------------0
     ASSOCIATIONS
       GDG------USER.GDGBASE
```

*Figure 12. Example of LISTCAT ALL Output (Part 2 of 11)*

```
NONVSAM ------- USER.GDGBASE.G0004V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS------------ACTIVE
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'     FSEQN------------------0
     ASSOCIATIONS
       GDG------USER.GDGBASE
NONVSAM ------- USER.GDGBASE.G0005V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS------------ACTIVE
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'     FSEQN------------------0
     ASSOCIATIONS
       GDG------USER.GDGBASE
NONVSAM ------- USER.GDGBASE.G0006V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS------------ACTIVE
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'     FSEQN------------------0
     ASSOCIATIONS
       GDG------USER.GDGBASE
NONVSAM ------- USER.GDGBASE.G0001V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS--------ROLLED-OFF
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'     FSEQN------------------0
     ASSOCIATIONS--------(NULL)
NONVSAM ------- USER.GDGBASE.G0002V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS----------DEFERRED
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0202     DEVTYPE------X'3030200E'     FSEQN------------------0
     ASSOCIATIONS--------(NULL)
```

*Figure 12. Example of LISTCAT ALL Output (Part 3 of 11)*

## LISTCAT Output

```
AIX ----------- USER.KSDS1.AIX1CLUS
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
       SMS MANAGED--------(YES)
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       DATA-----USER.KSDS1.AIX1DATA
       INDEX----USER.KSDS1.AIX1INDX
       CLUSTER--USER.KSDS1.CLUSTER
       PATH-----USER.KSDS1.PATHAIX1
     ATTRIBUTES
       UPGRADE
DATA ---------- USER.KSDS1.AIX1DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       AIX------USER.KSDS1.AIX1CLUS
     ATTRIBUTES
       KEYLEN-----------------4     AVGLRECL------------4086     BUFSPACE-----------29184     CISIZE-------------14336
       RKP--------------------5     MAXLRECL-----------32600     EXCPEXIT----------(NULL)     CI/CA------------------3
       STRIPE-COUNT-----------1
       SHROPTNS(1,3)   RECOVERY     UNIQUE           NOERASE     INDEXED       NOWRITECHK     NOIMBED       NOREPLICAT
       UNORDERED        NOREUSE     NONSPANNED      EXTENDED
       CCSID--------------65535
     STATISTICS
       REC-TOTAL------------180     SPLITS-CI--------------0     EXCPS----------------373
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
       REC-INSERTED----------59     FREESPACE-%CI---------20     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA---------20          X'A0D8F717D7463101'
       REC-RETRIEVED--------479     FREESPC------------28672
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-----------86016
       SPACE-PRI--------------1     HI-U-RBA-----------43008
       SPACE-SEC--------------1
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE--------14336     HI-A-RBA-----------43008     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK------------3     HI-U-RBA----------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0000000E'     LOW-RBA----------------0     TRACKS-----------------1
       HIGH-CCHH----X'0000000E'     HIGH-RBA-----------43007
     VOLUME
       VOLSER------------1P0202     PHYREC-SIZE--------14336     HI-A-RBA-----------86016     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK------------3     HI-U-RBA-----------43008     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00020007'     LOW-RBA------------43008     TRACKS-----------------1
       HIGH-CCHH----X'00020007'     HIGH-RBA-----------86015
```

*Figure 12. Example of LISTCAT ALL Output (Part 4 of 11)*

```
INDEX --------- USER.KSDS1.AIX1INDX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       AIX------USER.KSDS1.AIX1CLUS
     ATTRIBUTES
       KEYLEN-----------------4     AVGLRECL---------------0     BUFSPACE---------------0     CISIZE---------------512
       RKP--------------------5     MAXLRECL-------------505     EXCPEXIT----------(NULL)     CI/CA-----------------46
       SHROPTNS(2,3)   RECOVERY     UNIQUE           NOERASE     INDEXED       NOWRITECHK     NOREPLICAT     UNORDERED
       UNORDERED        NOREUSE     NONSPANNED
     STATISTICS
       REC-TOTAL--------------1     SPLITS-CI--------------0     EXCPS-----------------15     INDEX:
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1     LEVELS------------------1
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:            ENTRIES/SECT-----------1
       REC-UPDATED------------0     FREESPACE-%CA----------0          X'A0D8F717D7463101'     SEQ-SET-RBA--------23552
       REC-RETRIEVED----------0     FREESPC------------23040                                  HI-LEVEL-RBA-------23552
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-----------47104
       SPACE-PRI--------------1     HI-U-RBA-----------24064
       SPACE-SEC--------------1
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE----------512     HI-A-RBA-----------23552     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------46     HI-U-RBA---------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00010000'     LOW-RBA----------------0     TRACKS-----------------1
       HIGH-CCHH----X'00010000'     HIGH-RBA-----------23551
     VOLUME
       VOLSER------------1P0202     PHYREC-SIZE----------512     HI-A-RBA-----------47104     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------46     HI-U-RBA-----------24064     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00020008'     LOW-RBA------------23552     TRACKS-----------------1
       HIGH-CCHH----X'00020008'     HIGH-RBA-----------47103
DATA ---------- USER.KSDS1.CLDATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER--USER.KSDS1.CLUSTER
     ATTRIBUTES
       KEYLEN-----------------4     AVGLRECL------------2000     BUFSPACE------------6144     CISIZE--------------2048
       RKP--------------------0     MAXLRECL------------2000     EXCPEXIT----------(NULL)     CI/CA-----------------90
       SHROPTNS(1,3)   RECOVERY     UNIQUE           NOERASE     INDEXED       NOWRITECHK     NOIMBED       NOREPLICAT
       UNORDERED        NOREUSE     NONSPANNED
```

*Figure 12. Example of LISTCAT ALL Output (Part 5 of 11)*

## LISTCAT Output

```
     STATISTICS
       REC-TOTAL------------180     SPLITS-CI--------------0     EXCPS---------------1524
       REC-DELETED------------0     SPLITS-CA--------------1     EXTENTS----------------2
       REC-INSERTED----------59     FREESPACE-%CI---------20     SYSTEM-TIMESTAMP:
       REC-UPDATED----------179     FREESPACE-%CA---------20          X'A0D8F6E3D70D9401'
       REC-RETRIEVED-------1051     FREESPC----------------0
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA----------552960
       SPACE-PRI--------------5     HI-U-RBA----------552960
       SPACE-SEC--------------5
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE---------2048     HI-A-RBA----------184320     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------5
       EXTENTS:
       LOW-CCHH-----X'0013000A'     LOW-RBA----------------0     TRACKS-----------------5
       HIGH-CCHH----X'0013000E'     HIGH-RBA----------184319
     VOLUME
       VOLSER------------1P0202     PHYREC-SIZE---------2048     HI-A-RBA----------552960     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------552960     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------5
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA-----------184320     TRACKS-----------------5
       HIGH-CCHH----X'00010000'     HIGH-RBA----------368639
       LOW-CCHH-----X'0002000B'     LOW-RBA-----------368640     TRACKS-----------------5
       HIGH-CCHH----X'00030000'     HIGH-RBA----------552959
INDEX --------- USER.KSDS1.CLINDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER--USER.KSDS1.CLUSTER
     ATTRIBUTES
       KEYLEN-----------------4     AVGLRECL---------------0     BUFSPACE---------------0     CISIZE--------------2048
       RKP--------------------0     MAXLRECL------------2041     EXCPEXIT----------(NULL)     CI/CA-----------------18
       SHROPTNS(1,3)   RECOVERY     UNIQUE           NOERASE     NOWRITECHK       NOIMBED     NOREPLICAT     UNORDERED
       NOREUSE
     STATISTICS
       REC-TOTAL--------------4     SPLITS-CI--------------1     EXCPS---------------1377     INDEX:
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1     LEVELS-----------------2
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:            ENTRIES/SECT-----------9
       REC-UPDATED----------183     FREESPACE-%CA----------0          X'A0D8F6E3D70D9401'     SEQ-SET-RBA--------36864
       REC-RETRIEVED----------0     FREESPC------------28672                                  HI-LEVEL-RBA-------40960
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-----------73728
       SPACE-PRI--------------1     HI-U-RBA-----------45056
       SPACE-SEC--------------1
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE---------2048     HI-A-RBA-----------36864     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------------0     EXTENT-TYPE--------X'40'
```

*Figure 12. Example of LISTCAT ALL Output (Part 6 of 11)*

```
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0000000D'     LOW-RBA----------------0     TRACKS-----------------1
       HIGH-CCHH----X'0000000D'     HIGH-RBA-----------36863
     VOLUME
       VOLSER------------1P0202     PHYREC-SIZE---------2048     HI-A-RBA-----------73728     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA-----------45056     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00010001'     LOW-RBA------------36864     TRACKS-----------------1
       HIGH-CCHH----X'00010001'     HIGH-RBA-----------73727
CLUSTER ------- USER.KSDS1.CLUSTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1992.345
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
       BWO-------------  (NULL)
     RLSDATA
       LOG --------------(NULL)     RECOVERY REQUIRED --(NO)
       VSAM QUIESCED      (NO)      RLS IN USE ---------(NO)
       LOGSTREAMID ----------------------------------(NULL)
       RECOVERY TIMESTAMP LOCAL-----X'0000000000000000'
       RECOVERY TIMESTAMP GMT-------X'0000000000000000'
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       DATA-----USER.KSDS1.CLDATA
       INDEX----USER.KSDS1.CLINDEX
       AIX------USER.KSDS1.AIX1CLUS
       PATH-----USER.KSDS1.PATHCL
PATH ---------- USER.KSDS1.PATHAIX1
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       AIX------USER.KSDS1.AIX1CLUS
       DATA-----USER.KSDS1.AIX1DATA
       INDEX----USER.KSDS1.AIX1INDX
       DATA-----USER.KSDS1.CLDATA
       INDEX----USER.KSDS1.CLINDEX
     ATTRIBUTES
       UPDATE
PATH ---------- USER.KSDS1.PATHCL
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER--USER.KSDS1.CLUSTER
       DATA-----USER.KSDS1.CLDATA
       INDEX----USER.KSDS1.CLINDEX
     ATTRIBUTES
       UPDATE
```

*Figure 12. Example of LISTCAT ALL Output (Part 7 of 11)*

## LISTCAT Output

```
CLUSTER ------- USER.LINEAR
     HISTORY
       DATASET-OWNER---DEPTUSER     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------9999.999
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       DATA----USER.LINEAR.DATA
DATA ---------- USER.LINEAR.DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------9999.999
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER------USER.LINEAR
     ATTRIBUTES
       KEYLEN-----------------0     AVGLRECL---------------0     BUFSPACE------------8192     CISIZE--------------4096
       RKP--------------------0     MAXLRECL---------------0     EXCPEXIT----------(NULL)     CI/CA-----------------20
       SHROPTNS(1,3)   RECOVERY     UNIQUE           NOERASE     LINEAR        NOWRITECHK     NOIMBED       NOREPLICAT
       UNORDERED        NOREUSE     NONSPANNED
     STATISTICS
       REC-TOTAL--------------0     SPLITS-CI--------------0     EXCPS------------------0
       REC-DELETED------------0     SPLITS-CA--------------0     EXTENTS----------------1
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------0          X'0000000000000000'
       REC-RETRIEVED----------0     FREESPC-----------163840
     ALLOCATION
       SPACE-TYPE----------TRACK     HI-A-RBA----------163840
       SPACE-PRI--------------4     HI-U-RBA----------------0
       SPACE-SEC--------------2
     VOLUME
       VOLSER------------338001     PHYREC-SIZE---------4096     HI-A-RBA----------163840     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------10     HI-U-RBA----------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------2
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA----------------0     TRACKS-----------------4
       HIGH-CCHH----X'0000000E'     HIGH-RBA----------163839
NONVSAM ------- USER.MODEL
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1989.213
       RELEASE----------------2     EXPIRATION------0000.000
     VOLUMES
       VOLSER------------338001     DEVTYPE------X'3010200E'     FSEQN------------------0
     ASSOCIATIONS--------(NULL)
NONVSAM ------- USER.NONVSAM.DATA.SET
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1993.294
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---S1P01S02     MANAGEMENTCLASS---(NULL)
       DATACLASS ------SRX00001     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0302     DEVTYPE------X'3010200E'     FSEQN------------------0
     ASSOCIATIONS--------(NULL)
       ALIAS---------USER.ALIAS
     ATTRIBUTES
       STRIPE-COUNT-----------1
       ACT-DICT-TOKEN---X'4000000C01C401E301F001F101F201F301F401F501F601F701F801F907030D0108FE0DFE'
       COMP-FORMT
       CCSID--------------65535
     STATISTICS
       USER-DATA-SIZE--------------------------------3389920     COMP-USER-DATA-SIZE----------------------------910988
       SIZES-VALID--------(YES)
NONVSAM ------- USER.PDSE
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.244
       RELEASE----------------2     EXPIRATION------1992.244
       DSNTYPE----------LIBRARY
     SMSDATA
       STORAGECLASS ---S1P03S01     MANAGEMENTCLASS---(NULL)
       DATACLASS ------SRX00001     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0302     DEVTYPE------X'3030200E'     FSEQN------------------0
     ASSOCIATIONS--------(NULL)
```

*Figure 12. Example of LISTCAT ALL Output (Part 8 of 11)*

```
CLUSTER ------- USER.SPANNED.CLUSTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1992.345
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
       BWO-------------  (NULL)
     RLSDATA
       LOG --------------(NULL)     RECOVERY REQUIRED --(NO)
       VSAM QUIESCED      (NO)      RLS IN USE ---------(NO)
       LOGSTREAMID ------------------------------------(NULL)
       RECOVERY TIMESTAMP LOCAL-----X'0000000000000000'
       RECOVERY TIMESTAMP GMT-------X'0000000000000000'
     PROTECTION
       MASTERPW--------CLUSMPW1     UPDATEPW----------(NULL)     CODE--------------(NULL)     RACF----------------(NO)
       CONTROLPW---------(NULL)     READPW------------(NULL)     ATTEMPTS----------------2     USVR--------------(NULL)
       USAR--------------(NONE)
     ASSOCIATIONS
       DATA-----USER.SPANNED.DATA
       INDEX----USER.SPANNED.INDEX
DATA ---------- USER.SPANNED.DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER--USER.SPANNED.CLUSTER
     ATTRIBUTES
       KEYLEN-----------------4     AVGLRECL------------6000     BUFSPACE------------6144     CISIZE--------------2048
       RKP------------------10     MAXLRECL------------6000     EXCPEXIT----------(NULL)     CI/CA-----------------90
       SHROPTNS(1,3)   RECOVERY     UNIQUE           NOERASE     INDEXED       NOWRITECHK     IMBED         NOREPLICAT
       UNORDERED        NOREUSE     SPANNED
     STATISTICS
       REC-TOTAL------------100     SPLITS-CI--------------0     EXCPS----------------726
       REC-DELETED------------0     SPLITS-CA--------------2     EXTENTS----------------4
       REC-INSERTED----------40     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:
       REC-UPDATED------------0     FREESPACE-%CA----------0          X'A0D8F75A5E7E1700'
       REC-RETRIEVED--------100     FREESPC-----------491520
```

*Figure 12. Example of LISTCAT ALL Output (Part 9 of 11)*

```
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA----------737280
       SPACE-PRI----------------6   HI-U-RBA----------737280
       SPACE-SEC----------------6
     VOLUME
       VOLSER------------1P0301     PHYREC-SIZE---------2048     HI-A-RBA----------552960     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------552960     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------6
       LOW-KEY---------00000010
       HIGH-KEY--------00000320
       HI-KEY-RBA--------491520
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA----------------0     TRACKS-----------------6
       HIGH-CCHH----X'00010001'     HIGH-RBA----------184319
       LOW-CCHH-----X'00020008'     LOW-RBA-----------368640     TRACKS-----------------6
       HIGH-CCHH----X'0002000D'     HIGH-RBA----------552959
     VOLUME
       VOLSER------------1P0302     PHYREC-SIZE---------2048     HI-A-RBA----------737280     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------737280     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------6
       LOW-KEY---------00000330
       HIGH-KEY--------00000640
       HI-KEY-RBA--------675840
       EXTENTS:
       LOW-CCHH-----X'00020007'     LOW-RBA-----------184320     TRACKS-----------------6
       HIGH-CCHH----X'0002000C'     HIGH-RBA----------368639
       LOW-CCHH-----X'0002000D'     LOW-RBA-----------552960     TRACKS-----------------6
       HIGH-CCHH----X'00030003'     HIGH-RBA----------737279
INDEX --------- USER.SPANNED.INDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
     PROTECTION-PSWD-----(NULL)     RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER--USER.SPANNED.CLUSTER
     ATTRIBUTES
       KEYLEN-----------------4     AVGLRECL----------------0   BUFSPACE----------------0   CISIZE--------------2048
       RKP------------------10      MAXLRECL------------2041    EXCPEXIT----------(NULL)    CI/CA-----------------18
       SHROPTNS(1,3)   RECOVERY     UNIQUE           NOERASE     NOWRITECHK         IMBED    REPLICATE      UNORDERED
       NOREUSE
     STATISTICS
       REC-TOTAL----------------5   SPLITS-CI----------------2   EXCPS----------------401   INDEX:
       REC-DELETED------------0     SPLITS-CA----------------0   EXTENTS----------------5   LEVELS-----------------2
       REC-INSERTED-----------0     FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:          ENTRIES/SECT-----------9
       REC-UPDATED----------183     FREESPACE-%CA----------0         X'A0D8F75A5E7E1700'    SEQ-SET-RBA---------2048
       REC-RETRIEVED----------0     FREESPC----------------0                                HI-LEVEL-RBA-----------0
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-----------10240
       SPACE-PRI----------------1   HI-U-RBA-----------10240
       SPACE-SEC----------------1
     VOLUME
       VOLSER------------1P0301     PHYREC-SIZE---------2048     HI-A-RBA------------2048    EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA------------2048    EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA----------------1
       EXTENTS:
       LOW-CCHH-----X'0001000C'     LOW-RBA----------------0     TRACKS-----------------1
       HIGH-CCHH----X'0001000C'     HIGH-RBA------------2047
```

*Figure 12. Example of LISTCAT ALL Output (Part 10 of 11)*

```
     VOLUME
       VOLSER-----------------*     PHYREC-SIZE------------0     HI-A-RBA---------------0     EXTENT-NUMBER----------0
       DEVTYPE------X'3010200E'     PHYRECS/TRK------------0     HI-U-RBA---------------0     EXTENT-TYPE--------X'FF'
       VOLFLAG--------CANDIDATE     TRACKS/CA--------------0
     VOLUME
       VOLSER------------1P0301     PHYREC-SIZE---------2048     HI-A-RBA------------8192     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA------------8192     EXTENT-TYPE--------X'80'
       VOLFLAG------------PRIME     TRACKS/CA--------------6
       LOW-KEY---------00000010
       HIGH-KEY--------00000320
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA-------------2048     TRACKS-----------------6
       HIGH-CCHH----X'00010001'     HIGH-RBA------------4095
       LOW-CCHH-----X'00020008'     LOW-RBA-------------6144     TRACKS-----------------6
       HIGH-CCHH----X'0002000D'     HIGH-RBA------------8191
     VOLUME
       VOLSER------------1P0302     PHYREC-SIZE---------2048     HI-A-RBA-----------10240     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA-----------10240     EXTENT-TYPE--------X'80'
       VOLFLAG------------PRIME     TRACKS/CA--------------6
       LOW-KEY---------00000330
       HIGH-KEY--------00000640
       EXTENTS:
       LOW-CCHH-----X'00020007'     LOW-RBA-------------4096     TRACKS-----------------6
       HIGH-CCHH----X'0002000C'     HIGH-RBA------------6143
       LOW-CCHH-----X'0002000D'     LOW-RBA-------------8192     TRACKS-----------------6
       HIGH-CCHH----X'00030003'     HIGH-RBA-----------10239
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------1
                   ALIAS -----------------1
                   CLUSTER ---------------4
                   DATA ------------------5
                   GDG -------------------1
                   INDEX -----------------4
                   NONVSAM ---------------9
                   PAGESPACE -------------0
                   PATH ------------------2
                   SPACE -----------------0
                   USERCATALOG -----------0
                   TOTAL ----------------27
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

*Figure 12. Example of LISTCAT ALL Output (Part 11 of 11)*

## LISTCAT ALL Output Listing for a Non-VSAM Tailored Compressed Data Set

The following example illustrates the output produced for a non-VSAM tailored compressed data set. Tailored compression (used only with non-VSAM compressed data sets) creates dictionaries tailored specifically to the initial data written to the data set. Once derived, the dictionary is stored within the data set. This technique improves compression ratios over generic DBB compression.

**Note:** The following information is not an intended programming interface. It is provided for diagnostic purposes only.

The first byte of the dictionary token indicates the type of compression used for the data set.

**X'100. ....'** indicates compression has been rejected for the data set. No data is compressed.

**X'010. ....'** indicates generic DBB compression is used.

**X'011. ....'** indicates tailored compression is used.

### LISTCAT Output

```
    LISTCAT LEVEL(ZTAMS002) ALL
NONVSAM ------- ZTAMS002.COMPRS.DATA01
     IN-CAT --- SYS1.MVSRES.MASTCAT
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1996.297
       RELEASE----------------2     EXPIRATION------0000.000
       ACCOUNT-INFO-----------------------------------(NULL)
     SMSDATA
       STORAGECLASS ---S1P03S12     MANAGEMENTCLASS---(NULL)
       DATACLASS ------SRX00001     LBACKUP ---0000.000.0000
     VOLUMES
       VOLSER------------1P0301     DEVTYPE------X'3010200F'
FSEQN------------------0
       VOLSER------------1P0302     DEVTYPE------X'3010200F'
FSEQN------------------0
     ASSOCIATIONS--------(NULL)
     ATTRIBUTES
       STRIPE-COUNT-----------2

ACT-DIC-TOKEN----X'6200000B000000000000000000000000000000000000000000000000000
0000000000000'
       COMP-FORMT      EXTENDED
     STATISTICS
       USER-DATA-SIZE--------------------------------4509920
COMP-USER-DATA-SIZE----------------------------351227
       SIZES-VALID--------(YES)
```

*Figure 13. Example of LISTCAT ALL Output for a Non-VSAM Tailored Compressed Dataset*

## LISTCAT ALLOCATION Output Listing

When you specify the LISTCAT command and include the ALLOCATION parameter, each cataloged object with space allocated to it is listed (see Figure 14 on page 369). All information about the object's space is listed, but none of the object's other cataloged information is listed. The entry types that can be specified when the ALLOCATION parameter is specified are limited to DATA and INDEX.

```
  LISTCAT -
          LEVEL(USER)                 /* LIST ALL 'USER' ENTRIES   */ -
          ALLOCATION                  /* ALLOCATION INFORMATION    */ -
          CLUSTER                     /* INCLUDE CLUSTERS          */ -
          DATA                        /* AND DATA COMPONENTS       */ -
          INDEX                       /* AND INDEX COMPONENTS      */ -
          ALTERNATEINDEX              /* AND ALTERNATEINDEXES      */ -
          PATH                        /* AND PATHS                 */ -
          CATALOG(ICFUCAT1)           /* IN CATALOG ICFUCAT1       */

                              LISTING FROM CATALOG -- ICFUCAT1
CLUSTER ------- USER.DUMMY
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
DATA ---------- USER.DUMMY.CLDATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
     ALLOCATION
       SPACE-TYPE------CYLINDER     HI-A-RBA---------9400320
       SPACE-PRI-------------17     HI-U-RBA----------------0
       SPACE-SEC--------------0
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE---------2048     HI-A-RBA---------9400320     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA-------------15
       EXTENTS:
       LOW-CCHH-----X'00020000'     LOW-RBA----------------0     TRACKS----------------255
       HIGH-CCHH----X'0012000E'     HIGH-RBA---------9400319
INDEX --------- USER.DUMMY.CLINDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-----------76800
       SPACE-PRI--------------2     HI-U-RBA----------------0
       SPACE-SEC--------------0
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE---------2560     HI-A-RBA-----------76800     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------15     HI-U-RBA----------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA-------------1
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA----------------0     TRACKS-----------------2
       HIGH-CCHH----X'0000000C'     HIGH-RBA-----------76799
AIX ----------- USER.KSDS1.AIX1CLUS
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
       SMS MANAGED--------(YES)
```

*Figure 14. Example of LISTCAT ALLOCATION Output (Part 1 of 5)*

## LISTCAT Output

```
DATA ---------- USER.KSDS1.AIX1DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-----------86016
       SPACE-PRI--------------1     HI-U-RBA-----------43008
       SPACE-SEC--------------1
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE--------14336     HI-A-RBA-----------43008     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK------------3     HI-U-RBA---------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0000000E'     LOW-RBA----------------0     TRACKS-----------------1
       HIGH-CCHH----X'0000000E'     HIGH-RBA-----------43007
     VOLUME
       VOLSER------------1P0202     PHYREC-SIZE--------14336     HI-A-RBA-----------86016     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK------------3     HI-U-RBA-----------43008     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00020007'     LOW-RBA------------43008     TRACKS-----------------1
       HIGH-CCHH----X'00020007'     HIGH-RBA-----------86015
INDEX --------- USER.KSDS1.AIX1INDX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-------47104
       SPACE-PRI--------------1     HI-U-RBA--------24064
       SPACE-SEC--------------1
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE----------512     HI-A-RBA-----------23552     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------46     HI-U-RBA---------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00010000'     LOW-RBA----------------0     TRACKS-----------------1
       HIGH-CCHH----X'00010000'     HIGH-RBA-----------23551
     VOLUME
       VOLSER------------1P0202     PHYREC-SIZE----------512     HI-A-RBA-----------47104     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------46     HI-U-RBA-----------24064     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00020008'     LOW-RBA------------23552     TRACKS-----------------1
       HIGH-CCHH----X'00020008'     HIGH-RBA-----------47103
DATA ---------- USER.KSDS1.CLDATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA------552960
       SPACE-PRI--------------5     HI-U-RBA-------552960
       SPACE-SEC--------------5
```

*Figure 14. Example of LISTCAT ALLOCATION Output (Part 2 of 5)*

```
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE---------2048     HI-A-RBA------184320     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------5
       EXTENTS:
       LOW-CCHH-----X'0013000A'     LOW-RBA----------------0     TRACKS-----------------5
       HIGH-CCHH----X'0013000E'     HIGH-RBA----------184319
     VOLUME
       VOLSER------------1P0202     PHYREC-SIZE---------2048     HI-A-RBA----------552960     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------552960     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------5
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA-----------184320     TRACKS-----------------5
       HIGH-CCHH----X'00010000'     HIGH-RBA----------368639
       LOW-CCHH-----X'0002000B'     LOW-RBA-----------368640     TRACKS-----------------5
       HIGH-CCHH----X'00030000'     HIGH-RBA----------552959
INDEX --------- USER.KSDS1.CLINDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-----------73728
       SPACE-PRI--------------1     HI-U-RBA-----------45056
       SPACE-SEC--------------1
     VOLUME
       VOLSER------------1P0201     PHYREC-SIZE---------2048     HI-A-RBA-----------36864     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0000000D'     LOW-RBA----------------0     TRACKS-----------------1
       HIGH-CCHH----X'0000000D'     HIGH-RBA-----------36863
     VOLUME
       VOLSER------------1P0202     PHYREC-SIZE---------2048     HI-A-RBA-------73728     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA--------45056     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00010001'     LOW-RBA------------36864     TRACKS-----------------1
       HIGH-CCHH----X'00010001'     HIGH-RBA-----------73727
CLUSTER ------- USER.KSDS1.CLUSTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
PATH ---------- USER.KSDS1.PATHAIX1
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
```

*Figure 14. Example of LISTCAT ALLOCATION Output (Part 3 of 5)*

## LISTCAT Output

```
PATH ---------- USER.KSDS1.PATHCL
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
CLUSTER ------- USER.LINEAR
     HISTORY
       DATASET-OWNER---DEPTUSER     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------9999.999
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
DATA ---------- USER.LINEAR.DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------9999.999
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA----------163840
       SPACE-PRI--------------4     HI-U-RBA----------------0
       SPACE-SEC--------------2
     VOLUME
       VOLSER------------338001     PHYREC-SIZE---------4096     HI-A-RBA----------163840     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------10     HI-U-RBA----------------0     EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME     TRACKS/CA--------------2
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA----------------0     TRACKS-----------------4
       HIGH-CCHH----X'0000000E'     HIGH-RBA----------163839
CLUSTER ------- USER.SPANNED.CLUSTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
     SMSDATA
       STORAGECLASS ---S1P03S01     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
DATA ---------- USER.SPANNED.DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA----------737280
       SPACE-PRI--------------6     HI-U-RBA----------737280
       SPACE-SEC--------------6
     VOLUME
       VOLSER------------1P0301     PHYREC-SIZE---------2048     HI-A-RBA----------552960     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------552960     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------6
       LOW-KEY---------00000010
       HIGH-KEY--------00000320
       HI-KEY-RBA--------491520
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA----------------0     TRACKS-----------------6
       HIGH-CCHH----X'00010001'     HIGH-RBA----------184319
       LOW-CCHH-----X'00020008'     LOW-RBA-----------368640     TRACKS-----------------6
       HIGH-CCHH----X'0002000D'     HIGH-RBA----------552959
```

*Figure 14. Example of LISTCAT ALLOCATION Output (Part 4 of 5)*

```
     VOLUME
       VOLSER------------1P0302     PHYREC-SIZE---------2048     HI-A-RBA----------737280     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA----------737280     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------6
       LOW-KEY---------00000330
       HIGH-KEY--------00000640
       HI-KEY-RBA--------675840
       EXTENTS:
       LOW-CCHH-----X'00020007'     LOW-RBA-----------184320     TRACKS-----------------6
       HIGH-CCHH----X'0002000C'     HIGH-RBA----------368639
       LOW-CCHH-----X'0002000D'     LOW-RBA-----------552960     TRACKS-----------------6
       HIGH-CCHH----X'00030003'     HIGH-RBA----------737279
INDEX --------- USER.SPANNED.INDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
     ALLOCATION
       SPACE-TYPE---------TRACK     HI-A-RBA-----------10240
       SPACE-PRI--------------1     HI-U-RBA-----------10240
       SPACE-SEC--------------1
     VOLUME
       VOLSER------------1P0301     PHYREC-SIZE---------2048     HI-A-RBA------------2048     EXTENT-NUMBER----------1
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA------------2048     EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME     TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0001000C'     LOW-RBA----------------0     TRACKS-----------------1
       HIGH-CCHH----X'0001000C'     HIGH-RBA------------2047
     VOLUME
       VOLSER-----------------*     PHYREC-SIZE------------0     HI-A-RBA---------------0     EXTENT-NUMBER----------0
       DEVTYPE------X'3010200E'     PHYRECS/TRK------------0     HI-U-RBA---------------0     EXTENT-TYPE--------X'FF'
       VOLFLAG--------CANDIDATE     TRACKS/CA--------------0
     VOLUME
       VOLSER------------1P0301     PHYREC-SIZE---------2048     HI-A-RBA------------8192     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA------------8192     EXTENT-TYPE--------X'80'
       VOLFLAG------------PRIME     TRACKS/CA--------------6
       LOW-KEY---------00000010
       HIGH-KEY--------00000320
       EXTENTS:
       LOW-CCHH-----X'0000000B'     LOW-RBA-------------2048     TRACKS-----------------6
       HIGH-CCHH----X'00010001'     HIGH-RBA------------4095
       LOW-CCHH-----X'00020008'     LOW-RBA-------------6144     TRACKS-----------------6
       HIGH-CCHH----X'0002000D'     HIGH-RBA------------8191
     VOLUME
       VOLSER------------1P0302     PHYREC-SIZE---------2048     HI-A-RBA-----------10240     EXTENT-NUMBER----------2
       DEVTYPE------X'3010200E'     PHYRECS/TRK-----------18     HI-U-RBA-----------10240     EXTENT-TYPE--------X'80'
       VOLFLAG------------PRIME     TRACKS/CA--------------6
       LOW-KEY---------00000330
       HIGH-KEY--------00000640
       EXTENTS:
       LOW-CCHH-----X'00020007'     LOW-RBA-------------4096     TRACKS-----------------6
       HIGH-CCHH----X'0002000C'     HIGH-RBA------------6143
       LOW-CCHH-----X'0002000D'     LOW-RBA-------------8192     TRACKS-----------------6
       HIGH-CCHH----X'00030003'     HIGH-RBA-----------10239
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------1
                   ALIAS -----------------0
                   CLUSTER ---------------4
                   DATA ------------------5
                   GDG -------------------0
                   INDEX -----------------4
                   NONVSAM ---------------0
                   PAGESPACE -------------0
                   PATH ------------------2
                   SPACE -----------------0
                   USERCATALOG -----------0
                   TOTAL ----------------16
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

*Figure 14. Example of LISTCAT ALLOCATION Output (Part 5 of 5)*

## LISTCAT HISTORY Output Listing

When you specify the LISTCAT command and include the HISTORY or ALL parameter, only the name, ownerid, creation date, account information, and expiration date are listed for each entry that is selected (see Figure 15 on page 375). Only these types of entries have HISTORY information: ALTERNATEINDEX, CLUSTER, DATA, GDG, INDEX, NONVSAM, PAGESPACE and PATH.

```
  LISTCAT -
          LEVEL(USER)                /* LIST ALL 'USER' ENTRIES   */ -
          HISTORY                    /* SHOW HISTORY INFORMATION  */ -
          CATALOG(ICFUCAT1)          /* IN CATALOG ICFUCAT1       */

                            LISTING FROM CATALOG -- ICFUCAT1
ALIAS --------- USER.ALIAS
     HISTORY
       RELEASE----------------2
CLUSTER ------- USER.DUMMY
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
DATA ---------- USER.DUMMY.CLDATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
INDEX --------- USER.DUMMY.CLINDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
GDG BASE ------ USER.GDGBASE
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     LAST ALTER DATE-1993.001
NONVSAM ------- USER.GDGBASE.G0003V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS------------ACTIVE
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
NONVSAM ------- USER.GDGBASE.G0004V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS------------ACTIVE
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
NONVSAM ------- USER.GDGBASE.G0005V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS------------ACTIVE
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
NONVSAM ------- USER.GDGBASE.G0006V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS------------ACTIVE
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
```

*Figure 15. Example of LISTCAT HISTORY Output (Part 1 of 3)*

```
NONVSAM ------- USER.GDGBASE.G0001V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS--------ROLLED-OFF
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
NONVSAM ------- USER.GDGBASE.G0002V00
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.001
       STATUS----------DEFERRED
     SMSDATA
       STORAGECLASS ---S1P02S01     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
AIX ----------- USER.KSDS1.AIX1CLUS
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
       SMS MANAGED--------(YES)
DATA ---------- USER.KSDS1.AIX1DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
INDEX --------- USER.KSDS1.AIX1INDX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
DATA ---------- USER.KSDS1.CLDATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
INDEX --------- USER.KSDS1.CLINDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
CLUSTER ------- USER.KSDS1.CLUSTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
     SMSDATA
       STORAGECLASS ---S1P02S02     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
PATH ---------- USER.KSDS1.PATHAIX1
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
PATH ---------- USER.KSDS1.PATHCL
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.254
CLUSTER ------- USER.LINEAR
     HISTORY
       DATASET-OWNER---DEPTUSER     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------9999.999
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
DATA ---------- USER.LINEAR.DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------9999.999
```

*Figure 15. Example of LISTCAT HISTORY Output (Part 2 of 3)*

```
NONVSAM ------- USER.MODEL
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
NONVSAM ------- USER.NONVSAM.DATA.SET
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------0000.000
     SMSDATA
       STORAGECLASS ---S1P01S02     MANAGEMENTCLASS-S1P01M02
       DATACLASS ------PS000000     LBACKUP ---0000.000.0000
NONVSAM ------- USER.PDSE
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------1993.244
       DSNTYPE----------LIBRARY
     SMSDATA
       STORAGECLASS ---S1P03S01     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
CLUSTER ------- USER.SPANNED.CLUSTER
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
       BWO STATUS------11100000     BWO TIMESTAMP---0000 00:00:00.0
     SMSDATA
       STORAGECLASS ---S1P03S01     MANAGEMENTCLASS---(NULL)
       DATACLASS --------(NULL)     LBACKUP ---0000.000.0000
DATA ---------- USER.SPANNED.DATA
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
INDEX --------- USER.SPANNED.INDEX
     HISTORY
       DATASET-OWNER-----(NULL)     CREATION--------1991.323
       RELEASE----------------2     EXPIRATION------2007.365
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------1
                   ALIAS -----------------1
                   CLUSTER ---------------4
                   DATA ------------------5
                   GDG -------------------1
                   INDEX -----------------4
                   NONVSAM ---------------9
                   PAGESPACE -------------0
                   PATH ------------------2
                   SPACE -----------------0
                   USERCATALOG -----------0
                   TOTAL ----------------27
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

*Figure 15. Example of LISTCAT HISTORY Output (Part 3 of 3)*

## LISTCAT LEVEL Output Listing

LISTCAT LEVEL(USER) is specified in various LISTCAT examples throughout this appendix. Refer to the LISTCAT ALL or LISTCAT ALLOCATION examples for an example of a LISTCAT LEVEL output listing.

## LISTCAT ENTRIES Output Listing

When you specify the LISTCAT command and include the ENTRIES parameter, entries specified by the entryname are listed.

```
  LISTCAT -
          ENTRIES(USER.GDGBASE.*)     /* LIST ALL 'USER.GDGBASE'   */ -
          NAME                        /* NAMES ONLY                */ -
          CATALOG(ICFUCAT1)           /* IN CATALOG ICFUCAT1       */

                              LISTING FROM CATALOG -- ICFUCAT1
NONVSAM ------- USER.GDGBASE.G0001V00
NONVSAM ------- USER.GDGBASE.G0002V00
NONVSAM ------- USER.GDGBASE.G0003V00
NONVSAM ------- USER.GDGBASE.G0004V00
NONVSAM ------- USER.GDGBASE.G0005V00
NONVSAM ------- USER.GDGBASE.G0006V00
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------0
                   ALIAS -----------------0
                   CLUSTER ---------------0
                   DATA ------------------0
                   GDG -------------------0
                   INDEX -----------------0
                   NONVSAM ---------------6
                   PAGESPACE -------------0
                   PATH ------------------0
                   SPACE -----------------0
                   USERCATALOG -----------0
                   TOTAL -----------------6
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

*Figure 16. Example of LISTCAT ENTRIES Output*

## LISTCAT CREATION/EXPIRATION Output Listing

When you specify the LISTCAT command and include the CREATION or EXPIRATION parameter (or both), entries that have a creation or expiration date are selected according to the number of days you specify in the subparameter.

For example, in Figure 17 on page 379 only the entry USER.ALIAS is listed as a result of the LISTCAT CREATION(5) job because ALIAS entries have no creation date field and all the objects were created on the same day as the LISTCAT. When that job is run on an older catalog, each entry that was created the specified number of days ago or earlier is listed (that is, the CREATION number of days specifies that all objects in the catalog at least 5 days old are to be listed). The creation date of the data and index objects of a cluster or alternate index is always the same as the creation date of its associated cluster or alternate index object.

When you list all entries of a catalog, and you specify the CREATION parameter, each user catalog connector entry and each alias entry are also listed regardless of their creation date.

The LISTCAT CREATION keywords on the LISTCAT command are ignored if the LIBRARYENTRIES or VOLUMEENTRIES keywords are also supplied. Date filtering cannot be done for these types of entries being listed.

When the LISTCAT EXPIRATION(365) job is run, each entry whose expiration date occurs within 365 days of today's date is listed, as in Figure 18 on page 380.

When you list all entries of a catalog and you specify the EXPIRATION parameter, each volume entry is listed because volume entries have no expiration date.

The following can have a creation or expiration date: ALTERNATEINDEX, CLUSTER, DATA, GDG, INDEX, NONVSAM, PAGESPACE and PATH.

The LISTCAT EXPIRATION keywords on the LISTCAT command are ignored if the LIBRARYENTRIES or VOLUMEENTRIES keywords are also supplied. Date filtering cannot be done for these types of entries being listed.

```
 /*************************************************************************/
 /*   LIST EACH CATALOG ENTRY WHOSE CREATION DATE IS 5 DAYS AGO OR EARLIER  */
 /*   (THAT IS, THE OBJECT IS AT LEAST 5 DAYS OLD)                          */
 /*************************************************************************/
                                                        LISTING FROM CATALOG -- ICFUCAT1
 LISTCAT -
    CREATION(5) -
    CATALOG(ICFUCAT1/   )
ALIAS --------- USER.ALIAS
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------0
                   ALIAS -----------------1
                   CLUSTER ---------------0
                   DATA ------------------0
                   GDG -------------------0
                   INDEX -----------------0
                   NONVSAM ---------------0
                   PAGESPACE -------------0
                   PATH ------------------0
                   SPACE -----------------0
                   USERCATALOG -----------0
                   TOTAL -----------------1
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 17. Example of LISTCAT CREATION(5) Output*

## LISTCAT Output

```
/*****************************************************************/
/* LISTCAT 'USER' ENTRIES THAT HAVE EXPIRATION DATES WITHIN THE   */
/* NEXT 365 DAYS                                                  */
/*****************************************************************/
 LISTCAT -
         EXPIRATION(365)            /* LIST EXPIRATION OF 365    */ -
         NAME                       /* NAMES ONLY                */ -
         CATALOG(ICFUCAT1)          /* IN CATALOG ICFUCAT1       */

                            LISTING FROM CATALOG -- ICFUCAT1
CLUSTER ------- 00000000000000000000000000000000000000000000000
   DATA ------- ICFUCAT1
   INDEX ------ ICFUCAT1.CATINDEX
CLUSTER ------- SYS1.VVDS.V1P0101
   DATA ------- SYS1.VVDS.V1P0101
CLUSTER ------- SYS1.VVDS.V1P0201
   DATA ------- SYS1.VVDS.V1P0201
CLUSTER ------- SYS1.VVDS.V1P0202
   DATA ------- SYS1.VVDS.V1P0202
CLUSTER ------- SYS1.VVDS.V1P0301
   DATA ------- SYS1.VVDS.V1P0301
CLUSTER ------- SYS1.VVDS.V1P0302
   DATA ------- SYS1.VVDS.V1P0302
CLUSTER ------- SYS1.VVDS.V338001
   DATA ------- SYS1.VVDS.V338001
ALIAS --------- USER.ALIAS
CLUSTER ------- USER.DUMMY
   DATA ------- USER.DUMMY.CLDATA
   INDEX ------ USER.DUMMY.CLINDEX
GDG BASE ------ USER.GDGBASE
   NONVSAM ---- USER.GDGBASE.G0003V00
   NONVSAM ---- USER.GDGBASE.G0004V00
   NONVSAM ---- USER.GDGBASE.G0005V00
   NONVSAM ---- USER.GDGBASE.G0006V00
NONVSAM ------- USER.GDGBASE.G0001V00
NONVSAM ------- USER.GDGBASE.G0002V00
AIX ----------- USER.KSDS1.AIX1CLUS
   DATA ------- USER.KSDS1.AIX1DATA
   INDEX ------ USER.KSDS1.AIX1INDX
   PATH ------- USER.KSDS1.PATHAIX1
CLUSTER ------- USER.KSDS1.CLUSTER
   DATA ------- USER.KSDS1.CLDATA
   INDEX ------ USER.KSDS1.CLINDEX
   PATH ------- USER.KSDS1.PATHCL
NONVSAM ------- USER.MODEL
NONVSAM ------- USER.NONVSAM.DATA.SET
NONVSAM ------- USER.PDSE
         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------1
                   ALIAS -----------------1
                   CLUSTER ---------------9
                   DATA ----------------10
                   GDG -------------------1
                   INDEX ----------------4
                   NONVSAM ---------------9
                   PAGESPACE -------------0
                   PATH ------------------2
                   SPACE ----------------0
                   USERCATALOG -----------0
                   TOTAL ----------------37
         THE NUMBER OF PROTECTED ENTRIES SUPPRESSED WAS 0
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

*Figure 18. Example of LISTCAT EXPIRATION(365) Output*

## Examples of LISTCAT in a TSO Environment

The following examples illustrate the output produced at a TSO terminal for a LISTCAT NAMES (default) and LISTCAT VOLUME. A TSO logon ID of IBMUSER is assumed.

For LISTCAT NAMES, the catalog name is printed followed by the names of all entries that have a high-level qualifier equal to the USER logon ID.

For LISTCAT VOLUME, all entrynames that have a high-level qualifier equal to the USER logon ID are printed, followed by the volume serial numbers for those entries that contain volume information.

**Note:** Because volume serial numbers for a cluster or an alternate index are contained in the data and index components, the data and index must have been named on the initial DEFINE in order to list the volume serial numbers.

```
LOGON  IBMUSER

READY
LISTCAT

IN  CATALOG:  ICFMAST1
IBMUSER.AIX
IBMUSER.AIXDATA
IBMUSER.AIXIDX
IBMUSER.GDG
IBMUSER.GDG.G0001V00
IBMUSER.GDG.G0002V00
IBMUSER.GDG.G0003V00
IBMUSER.KSDS
IBMUSER.KSDSDATA
IBMUSER.KSDSIDX
IBMUSER.NVSAM1
IBMUSER.NVSAM2
IBMUSER.NVSAM3
IBMUSER.NVSAM4
IBMUSER.NVSAM5
READY

LISTCAT  VOLUME

IBMUSER.AIX
IBMUSER.AIXDATA
   --VOLUMES--
      333001
IBMUSER.AIXIDX
   --VOLUMES--
      333001
IBMUSER.GDG
IBMUSER.GDG.G0001V00
   --VOLUMES--
      333001
      333002
```

```
      333003
IBMUSER.GDG.G0002V00
   --VOLUMES--
      333004
      333005
      333006
      333007
      333008
IBMUSER.GDG.G0003V00
   --VOLUMES--
      333009
      333010
IBMUSER.KSDS
IBMUSER.KSDSDATA
   --VOLUMES--
      333001
IBMUSER.KSDSIDX
   --VOLUMES--
      333001
IBMUSER.NVSAM1
   --VOLUMES--
      333001
      333002
IBMUSER.NVSAM2
   --VOLUMES--
      333003
      333004
      333005
IBMUSER.NVSAM3
   --VOLUMES--
      333006
IBMUSER.NVSAM4
   --VOLUMES--
      333007
IBMUSER.NVSAM5
   --VOLUMES--
      333008
      333009
      333010
      333011
      333012
  READY
```

# Appendix C. Interpreting SHCDS Output Listings

## LISTDS

The following is listed for each data set:

- Cache structure name
- If the subsystem sharing the data set owns:
  - Retained locks
  - Lost locks
- If locks are not bound to the data set
- If the data set is recoverable
- If non-RLS updates are permitted (PERMITNONRLSUPDATE was used)
- Status of RLS usage since permitting non-RLS update
- If forward recovery is required

The report also gives a list of subsystems sharing the data set. For each subsystem, LISTDS returns:

- Subsystem name
- If sharing protocol is used by the subsystem (online) and if it is currently active
- Retained lock status
- Lost lock status
- If the subsystem requires recovery for data sets in a non-RLS update permitted state

## LISTDS with Data Set in Retained Lock State

The first part of Figure 19 on page 384 gives the status of the data set. The data set:

- Has retained locks
- Is a recoverable data set
- Is not in NON-RLS UPDATE PERMITTED state

The second part of Figure 19 on page 384 shows a subsystem sharing the data set and its status relative to the data set. In this example, the only subsystem sharing the data set is RETLK05A. The RETLK05A subsystem:

- Is a commit protocol application (ONLINE)
- Is currently active (ACTIVE)
- Owns retained locks for this data set

If there are no subsystems sharing the data set, the following is displayed:

```
IDC3189I SUBSYSTEM NOT LISTED RC=8, RS=4.
```

The retained lock state example follows:

© Copyright IBM Corp. 1973, 2002

```
  SHCDS LISTDS(SYSPLEX.KSDS.RETAINED.CLUS1)
----- LISTING FROM SHCDS ----- IDCSH02
-----------------------------------------------------------------
DATA SET NAME----SYSPLEX.KSDS.RETAINED.CLUS1
  CACHE STRUCTURE----CACHE01
  RETAINED LOCKS---------YES   NON-RLS UPDATE PERMITTED---------NO
  LOST LOCKS--------------NO   PERMIT FIRST TIME----------------NO
  LOCKS NOT BOUND---------NO   FORWARD RECOVERY REQUIRED--------NO
  RECOVERABLE------------YES

                          SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  RETLK05A     ONLINE--ACTIVE    YES         NO       NO
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 19. LISTDS with Data Set in Retained Lock State*

## LISTDS for Data Set Shared by Multiple Subsystems

The first part of Figure 20 summarizes the status for the data set. In this example, the data set:

- Has retained locks
- Is a recoverable data set
- Non-RLS update is not permitted.

The second part of Figure 20 lists the subsystems sharing the data set and their status relative to the data set. All of the subsystems are commit protocol applications. Subsystem RETLK05A is active, while the others are not currently active. All of the applications own retained locks for this data set.

```
  SHCDS LISTDS(SYSPLEX.KSDS.SHARED.CLUS4)
----- LISTING FROM SHCDS ----- IDCSH02
-----------------------------------------------------------------
DATA SET NAME----SYSPLEX.KSDS.SHARED.CLUS4
  CACHE STRUCTURE----CACHE01
  RETAINED LOCKS---------YES   NON-RLS UPDATE PERMITTED---------NO
  LOST LOCKS--------------NO   PERMIT FIRST TIME----------------NO
  LOCKS NOT BOUND---------NO   FORWARD RECOVERY REQUIRED--------NO
  RECOVERABLE------------YES

                          SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  KMKLK05D     ONLINE--FAILED    YES         NO       NO
  KMKLK05F     ONLINE--FAILED    YES         NO       NO
  RETLK05A     ONLINE--ACTIVE    YES         NO       NO
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 20. LISTDS for Data Set Being Shared by Multiple Subsystems*

## LISTDS for Data Set in Non-RLS Permitted State

The first part of Figure 21 on page 385 summarizes the status for the data set. In this example, the data set:

- Has retained locks
- Is a recoverable data set

- Is not in NON-RLS UPDATE PERMITTED state
- Has been processed by an RLS application that reset the PERMIT FIRST TIME status

The second part of Figure 21 lists the subsystems sharing the data set and their status relative to the data set. The subsystems are commit protocol applications (ONLINE), and are not currently active (FAILED). Both of the applications own retained locks for this data set.

Subsystem RETLK05A has not completed recovery for the NON-RLS PERMITTED state of the data set.

Subsystem KMKLK05D has either cleared the NON-RLS PERMITTED state or began sharing the data set after setting the NON-RLS PERMITTED state.

```
  SHCDS LISTDS(SYSPLEX.KSDS.PERMIT.CLUS2)
----- LISTING FROM SHCDS ----- IDCSH02
-----------------------------------------------------------------
DATA SET NAME----SYSPLEX.KSDS.PERMIT.CLUS2
  CACHE STRUCTURE----CACHE01
  RETAINED LOCKS---------YES   NON-RLS UPDATE PERMITTED--------YES
  LOST LOCKS--------------NO   PERMIT FIRST TIME----------------NO
  LOCKS NOT BOUND---------NO   FORWARD RECOVERY REQUIRED--------NO
  RECOVERABLE------------YES

                           SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  KMKLK05D     ONLINE--FAILED    YES         NO       NO
  RETLK05A     ONLINE--FAILED    YES         NO       YES
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 21. LISTDS for Data Set in NON-RLS PERMITTED State*

## LISTDS with Data Set in Non-RLS Update and Permit First Time States

The first part of Figure 22 on page 386 gives the status for the data set. In this example, the data set:

- Has retained locks
- Is a recoverable data set
- Is in NON-RLS UPDATE PERMITTED state

The PERMIT FIRST TIME--YES status indicates that the data set has not been processed by an RLS application since the data set was put in the NON-RLS UPDATE PERMITTED state.

The second part of Figure 22 on page 386 shows a subsystem sharing the data set and its status relative to the data set. In this example, the only subsystem sharing the data set is RETLK05A. The RETLK05A subsystem:

- Is a commit protocol application (ONLINE)
- Is currently active (ACTIVE)
- Owns retained locks for this data set

In this case, the RETLK05A subsystem is required to recover the NON-RLS UPDATE PERMITTED data set. Until this recovery is done, the subsystem is notified that the data set is in the NON-RLS UPDATE PERMITTED state.

```
  SHCDS LISTDS(SYSPLEX.KSDS.PERMIT.CLUS2)
----- LISTING FROM SHCDS ----- IDCSH02
-------------------------------------------------------------------
DATA SET NAME----SYSPLEX.KSDS.PERMIT.CLUS2
  CACHE STRUCTURE----CACHE01
  RETAINED LOCKS---------YES   NON-RLS UPDATE PERMITTED--------YES
  LOST LOCKS--------------NO   PERMIT FIRST TIME---------------YES
  LOCKS NOT BOUND---------NO   FORWARD RECOVERY REQUIRED--------NO
  RECOVERABLE------------YES

                          SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  RETLK05A     ONLINE--ACTIVE    YES         NO       YES
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 22. LISTDS for Data Set in Both NON-RLS UPDATE and PERMIT FIRST TIME States*

## LISTDS for Data Set in Lost Lock State

The first part of Figure 23 summarizes the status for the data set. In this example, the data set:

- Has lost locks
- Is a recoverable data set
- Is not in NON-RLS UPDATE PERMITTED state

The second part of Figure 23 shows a subsystem sharing the data set and gives status relative to the data set. The subsystem is a commit protocol application (ONLINE), and is not currently active (FAILED). The application owns locks that have been lost.

```
  SHCDS LISTDS(SYSPLEX.KSDS.LOSTLOCK.CLUS5)
----- LISTING FROM SHCDS ----- IDCSH02
-------------------------------------------------------------------
DATA SET NAME----SYSPLEX.KSDS.LOSTLOCK.CLUS5
  CACHE STRUCTURE----- NOT ASSIGNED -
  RETAINED LOCKS----------NO   NON-RLS UPDATE PERMITTED---------NO
  LOST LOCKS-------------YES   PERMIT FIRST TIME----------------NO
  LOCKS NOT BOUND---------NO   FORWARD RECOVERY REQUIRED--------NO
  RECOVERABLE------------YES

                          SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  ONLINE01     ONLINE--FAILED    NO          YES      NO
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 23. LISTDS for Data Set in Lost Lock State*

# LISTSUBSYS

The following is listed for each subsystem:

- Sharing protocol and current status
- If recovery is required for the subsystem
- Subsystem owned retained locks or lost locks
- Number of:
  - Locks held

  - Locks waiting
  - Locks retained
  - Data sets shared by the application in lost locks state
  - Data sets in NON-RLS UPDATE PERMITTED state
  - Current transactions

## LISTSUBSYS for All Subsystems Sharing Data Sets in the Sysplex

Figure 24 shows a SHCDS LISTSUBSYS for all subsystems registered with the SMSVSAM server.

- The SMSVSAM subsystem:
  - Is a non commit protocol application (BATCH)
  - Is currently active (ACTIVE)
- The KMKLK05D subsystem:
  - Is a commit protocol application (ONLINE)
  - Is not currently active (FAILED)
  - Has one retained lock
  - Has one active transaction
- The KMKLK05F subsystem:
  - Is a commit protocol application (ONLINE)
  - Is not currently active (FAILED)
  - Has one retained lock
  - Has one active transaction
- The RETLK05A subsystem:
  - Is a commit protocol application (ONLINE)
  - Has fifteen retained locks
  - Has one data set in a lost lock state
  - Has one active transaction

```
  SHCDS LISTSUBSYS(ALL)
----- LISTING FROM SHCDS ----- IDCSH03
-----------------------------------------------------------------------------------------------
                                  RECOVERY            LOCKS         LOCKS          LOCKS
SUBSYSTEM NAME   STATUS           NEEDED              HELD         WAITING        RETAINED
--------------   --------------   --------------   -----------------------------------------
SMSVSAM          BATCH --ACTIVE   NO                       0             0              0
    DATA SETS IN LOST LOCKS------------         0
    DATA SETS IN NON-RLS UPDATE STATE--         0
    TRANSACTION COUNT------------------         0
KMKLK05D         ONLINE--FAILED   YES                      0             0              1
    DATA SETS IN LOST LOCKS------------         0
    DATA SETS IN NON-RLS UPDATE STATE--         0
    TRANSACTION COUNT------------------         1
KMKLK05F         ONLINE--FAILED   YES                      0             0              1
    DATA SETS IN LOST LOCKS------------         0
    DATA SETS IN NON-RLS UPDATE STATE--         0
    TRANSACTION COUNT------------------         1
RETLK05A         ONLINE--ACTIVE   YES                      0             0             15
    DATA SETS IN LOST LOCKS------------         1
    DATA SETS IN NON-RLS UPDATE STATE--         0
    TRANSACTION COUNT------------------         1
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 24. LISTSUBSYS for all Subsystems Sharing Data Sets in the Sysplex*

## LISTSUBSYSDS

The following is listed for each subsystem:

- Sharing protocol and current status
- Retained locks owned
- Lost locks owned
- Locks not bound to the data set
- If forward recovery has been set in the catalog entry for shared data sets.
- Whether non-RLS update is permitted relative to the subsystem
- Subsystem access to the data set at the time the data set was placed in NON-RLS UPDATE PERMITTED state

## LISTSUBSYSDS for Subsystem Sharing Multiple Data Sets

Figure 25 shows a SHCDS LISTSUBSYSDS for a single subsystem registered with SMSVSAM address space.

- Subsystem RETLK05A is a commit protocol application (ONLINE) and is currently active (ACTIVE).
- The subsystem is sharing three data sets and owns retained locks on all three data sets.
- None of the data sets:
  - Have locks that are not bound to the data set
  - Have forward recovery set in their catalog entries
  - Are in NON-RLS UPDATE PERMITTED state

Because the data sets are not in NON-RLS UPDATE PERMITTED state, none of the data sets were accessed at the time the NON-RLS PERMITTED state was set.

```
  SHCDS LISTSUBSYSDS(RETLK05A)
----- LISTING FROM SHCDS ----- IDCSH04
--------------------------------------------------------------------------------
SUBSYSTEM NAME---- RETLK05A      SUBSYSTEM STATUS----ONLINE--ACTIVE

                                        LOCKS                NON-RLS     PERMIT
DATA SET NAME /     RETAINED   LOST    NOT       RECOVERY   UPDATE      FIRST TIME
CACHE  STRUCTURE    LOCKS      LOCKS   BOUND     REQUIRED   PERMITTED   SWITCH
----------------    --------   -----   -------   --------   ---------   -----
SYSPLEX.KSDS.PERMIT.CLUS2
CACHE01             YES        NO      NO        NO         NO          NO
SYSPLEX.KSDS.RETAINED.CLUS1
CACHE01             YES        NO      NO        NO         NO          NO
SYSPLEX.KSDS.SHARED.CLUS4
CACHE01             YES        NO      NO        NO         NO          NO
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 25. LISTSUBSYSDS for Subsystem Sharing Multiple Data Sets*

## LISTSUBSYSDS for All Subsystems in the Sysplex and the Shared Data Sets

Figure 26 on page 390 shows a SHCDS LISTSUBSYSDS for all subsystems registered with SMSVSAM address space.

Subsystem SMSVSAM is:

- A non-commit protocol application (BATCH)
- Is currently active (ACTIVE)

- Not currently sharing any data sets

If a subsystem is not sharing any data sets, the following is displayed:

```
IDC31890I DATASET NOT LISTED RC = 8, RS =   2.
```

Subsystem KMKLK05D:
- Is a commit protocol application (ONLINE)
- Is not currently active (FAILED)
- Is sharing one data set
- Owns retained locks for that data set

Subsystem KMKLK05F:
- Is a commit protocol application (ONLINE)
- Is not currently active (FAILED)
- Is sharing one data set
- Owns retained locks for the data set

Subsystem RETLK05A:
- Is a commit protocol application (ONLINE)
- Is not currently active (FAILED)
- Is sharing three data sets
- Owns retained locks on all of the data sets

RETLK05A has to perform recovery for the NON-RLS UPDATE PERMITTED state of data set SYSPLEX.KSDS.PERMIT.CLUS2.

RETLK05A accessed SYSPLEX.KSDS.PERMIT.CLUS2 when the NON-RLS UPDATE PERMITTED state was set.

### SHCDS Output

```
  SHCDS LISTSUBSYSDS(ALL)
----- LISTING FROM SHCDS ----- IDCSH04
--------------------------------------------------------------------------------
SUBSYSTEM NAME---- SMSVSAM       SUBSYSTEM STATUS----BATCH --ACTIVE
IDC31890I DATASET NOT LISTED RC = 8, RS =   2.
SUBSYSTEM NAME---- KMKLK05D      SUBSYSTEM STATUS----ONLINE--FAILED

                                        LOCKS                NON-RLS     PERMIT
DATA SET NAME /     RETAINED   LOST    NOT       RECOVERY   UPDATE      FIRST TIME
CACHE  STRUCTURE    LOCKS      LOCKS   BOUND     REQUIRED   PERMITTED   SWITCH
----------------    --------   -----   -------   --------   ---------   -----
SYSPLEX.KSDS.SHARED.CLUS4
CACHE01             YES        NO      NO        NO         NO          NO
SUBSYSTEM NAME---- KMKLK05F      SUBSYSTEM STATUS----ONLINE--FAILED

                                        LOCKS                NON-RLS     PERMIT
DATA SET NAME /     RETAINED   LOST    NOT       RECOVERY   UPDATE      FIRST TIME
CACHE  STRUCTURE    LOCKS      LOCKS   BOUND     REQUIRED   PERMITTED   SWITCH
----------------    --------   -----   -------   --------   ---------   -----
SYSPLEX.KSDS.SHARED.CLUS4
CACHE01             YES        NO      NO        NO         NO          NO
SUBSYSTEM NAME---- RETLK05A      SUBSYSTEM STATUS----ONLINE--ACTIVE

                                        LOCKS                NON-RLS     PERMIT
DATA SET NAME /     RETAINED   LOST    NOT       RECOVERY   UPDATE      FIRST TIME
CACHE  STRUCTURE    LOCKS      LOCKS   BOUND     REQUIRED   PERMITTED   SWITCH
----------------    --------   -----   -------   --------   ---------   -----
SYSPLEX.KSDS.PERMIT.CLUS2
CACHE01             YES        NO      NO        NO         YES         YES
SYSPLEX.KSDS.RETAINED.CLUS1
CACHE01             YES        NO      NO        NO         NO          NO
SYSPLEX.KSDS.SHARED.CLUS4
CACHE01             YES        NO      NO        NO         NO          NO
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 26. LISTSUBSYSDS for all Subsystems in the Sysplex and the Shared Data Sets*

## LISTRECOVERY

The following is listed for each data set:

- If the subsystem sharing the data set owns:
  - Retained locks
  - Lost locks
- If locks are not bound to the data set
- If the forward recovery is set in the catalog entry
- NON-RLS UPDATE PERMITTED status
- Status of RLS usage for a data set since setting the NON-RLS PERMITTED state.

The report also gives a list of subsystems sharing the data set. You will get this information:

- Subsystem name
- If sharing protocol is used by the subsystem and if it is currently active
- Retained locks status
- Lost lock status

- If the subsystem requires recovery for data sets in a NON-RLS UPDATE PERMITTED state

## LISTRECOVERY for Data Set Requiring Recovery

Figure 27 shows a SHCDS LISTRECOVERY for a single data set. The SHCDS LISTRECOVERY command displays data set information if the data sets have a form of recovery to be performed.

The first part of the report shows the data set:
- Has lost locks
- Does not have locks bound to the data set
- Does not have forward recovery set in the catalog entry
- Has not been set to the NON-RLS UPDATE PERMITTED state

The second part of the report shows the data set is:
- Shared by a single commit protocol application (ONLINE)
- Is not currently active (FAILED), owns lost locks
- Has no NON-RLS UPDATE PERMITTED recovery to perform

```
  SHCDS LISTRECOVERY(SYSPLEX.LOSTLOCK.CLUS1)
----- LISTING FROM SHCDS ----- IDCSH05
--------------------------------------------------------------------------------
                                                    LOCKS    NON-RLS               PERMIT
                                  RETAINED  LOST    NOT      RECOVERY   UPDATE     FIRST TIME
DATA SET NAME                     LOCKS     LOCKS   BOUND    REQUIRED   PERMITTED  SWITCH
----------------------------      --------  -----   -----    --------   -------    ----------
SYSPLEX.LOSTLOCK.CLUS1            NO        YES     NO       NO         NO         NO

                           SHARING SUBSYSTEM STATUS
  SUBSYSTEM    SUBSYSTEM         RETAINED    LOST     NON-RLS UPDATE
  NAME         STATUS            LOCKS       LOCKS    PERMITTED
  ---------    --------------    --------    -----    --------------
  RETLK05A     ONLINE--FAILED    NO          YES      NO
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

*Figure 27. LISTRECOVERY for Data Set Requiring Recovery*

# Appendix D. Invoking Access Method Services from Your Program

This appendix is intended to help you to invoke access method services from your program.

Access method services is invoked by your program through the ATTACH, LINK, or LOAD and CALL macro instructions.

The dynamic invocation of access method services enables respecification of selected processor defaults as well as the ability to manage input/output operations for selected data sets.

A processing program invokes access method services with the ATTACH, LINK, LOAD, and CALL macros. Before issuing the invoking macro, however, the program must initialize the appropriate register and operand list contents.

The register contents follow standard linkage conventions:

- register 1 contains the address of the argument list
- register 13 contains the address of a save area
- register 14 contains the address of the return point
- register 15 contains the address of the entry point IDCAMS in access method services.

IDCAMS only supports 24-bit virtual addresses that are passed in argument lists, control blocks, buffers, and user exit routines, and USER EXIT can only be called in AMODE24. However, IDCAMS can be *called* in 31-bit mode but will switch to 24-bit mode and then return to the calling program in the caller's mode.

The contents of the operand list are described in Figure 28 on page 397. Refer to "Authorized Program Facility" in *z/OS DFSMS: Managing Catalogs* for information on manipulating sensitive data in a secured environment.

## Authorized Program Facility (APF)

Any program calling IDCAMS to issue a DCOLLECT or SHCDS command must be APF authorized or these commands will terminate. When IDCAMS is called from a program to issue the ALLOCATE command to allocate a SMS-managed data set, that program must be in an APF authorized program. For information on APF authorization, see *z/OS DFSMS: Managing Catalogs*. For information on using APF, see *z/OS MVS JCL User's Guide*.

## Invoking Macro Instructions

The following descriptions of the invoking macro instructions are related to Figure 28, which describes the argument lists referenced by the invoking macros.

### LINK or ATTACH Macro Instruction

Access method services is invoked through either the LINK or the ATTACH macro instruction.

You cannot use the IDCAMS ALLOCATE command after using ATTACH to call IDCAMS. If you do, ALLOCATE fails with an ATTACH return code.

© Copyright IBM Corp. 1973, 2002

### Invoking from Program

The syntax of the LINK or ATTACH macro instruction is:

| | | |
|---|---|---|
| [*label*] | **LINK\|ATTACH** | **EP=IDCAMS,**<br>**PARAM=(***optionaddr*<br>**[,***dnameaddr***]**<br>**[,***pgnoaddr***]**<br>**[,***iolistaddr***]**<br>**[,***auxlistaddr***]),**<br>**VL=1** |

**EP=IDCAMS**
specifies that the program to be invoked is IDCAMS.

**PARAM=**
specifies the addresses of the parameters to be passed to IDCAMS. These values can be coded:

*optionaddr*
specifies the address of an option list, which can be specified in the PARM parameter of the EXEC statement and is a valid set of parameters for the access method services PARM command. If you do not want to specify any options, this address must point to a halfword of binary zeros. Figure 28 on page 397 shows the format of the options list.

*dnameaddr*
specifies the address of a list of alternate ddnames for standard data sets used during IDCAMS processing. If standard ddnames are used and this is not the last parameter in the list, it should point to a halfword of binary zeros. If it is the last parameter, it can be omitted. Figure 28 on page 397 shows the format of the alternate ddname list.

*pgnoaddr*
specifies the address of a 3- to 6-byte area that contains an EBCDIC starting page number for the system output file. If the page number is not specified, but this is not the last parameter in the list, the parameter must point to a halfword of binary zeros. If it is the last parameter, it can be omitted. If omitted, the default page number is 1. Figure 28 shows the format of the page number area.

*iolistaddr*
specifies the address of a list of externally controlled data sets and the addresses of corresponding I/O routines. If no external I/O routines are supplied, this parameter can be omitted. Figure 28 shows the format of the I/O list.

*auxlistaddr*
specifies the address of the auxiliary list. Figure 28 on page 397 shows the format of the auxiliary list.

**VL=1**
causes the high-order bit of the last address parameter of the PARAM list to be set to 1.

## LOAD and CALL Macro Instructions

Access method services is also invoked with a LOAD of the module IDCAMS, followed by a CALL to that module. The syntax of the LOAD macro instruction is:

| | | |
|---|---|---|
| [*label*] | **LOAD** | **{EP=IDCAMS\|EPLOC=***address of name***}** |

| | | |
|---|---|---|
| | | |

where:

**EP=IDCAMS**
is the entry point name of the IDCAMS program to be loaded into virtual storage.

**EPLOC=***address of name*
is the address of an 8-byte character string IDCAMSbb.

After loading IDCAMS, register 15 must be loaded with the address returned from the LOAD macro. Use CALL to pass control to IDCAMS. The syntax of the CALL macro instruction is:

| | | |
|---|---|---|
| **[***label***]** | **LR**<br>**CALL** | **15,0**<br>**(15),**<br>**(***optionaddr*<br>**[,***dnameaddr***]**<br>**[,***pgnoaddr***]**<br>**[,***iolistaddr***]**<br>**[,***auxlistaddr***]),**<br>**VL** |

where:

**15** is the register containing the address of the entry point to be given control.

*optionaddr*
specifies the address of an options list that can be specified in the PARM parameter of the EXEC statement and is a valid set of parameters for the access method services PARM command. If you do not want to specify any options, this address must point to a halfword of binary zeros. Figure 28 on page 397 shows the format of the options list.

*dnameaddr*
specifies the address of a list of alternate ddnames for standard data sets used during IDCAMS processing. If standard ddnames are used and this is not the last parameter in the list, it should point to a halfword of binary zeros. If it is the last parameter, it can be omitted. Figure 28 on page 397 shows the format of the alternate ddname list.

*pgnoaddr*
specifies the address of a 6-byte area that contains an EBCDIC starting page number for the system output file. If the page number is not specified, but this is not the last parameter in the list, the parameter must point to a halfword of binary zeros. If it is the last parameter, it can be omitted. If omitted, the default page number is 1. Figure 28 on page 397 shows the format of the page number area.

*iolistaddr*
specifies the address of a list of externally controlled data sets and the addresses of corresponding I/O routines. If no external I/O routines are supplied, this parameter can be omitted. Figure 28 on page 397 shows the format of the I/O list.

*auxlistaddr*
specifies the address of the auxiliary list. Figure 28 on page 397 shows the format of the auxiliary list.

**VL**
causes the high-order bit of the last address parameter in the macro expansion to be set to 1.

## Invocation from a PL/I Program

Access method services can also be invoked from a PL/I program using the facilities of the IBM PL/I Optimizing Compiler Licensed Program. IDCAMS must be declared to the compiler as an external entry point with the ASSEMBLER and INTER options. The access method services processor is loaded by issuing a FETCH IDCAMS statement, is reached with a CALL statement, and deleted by a RELEASE IDCAMS statement. The syntax of the CALL statement is:

| | | |
|---|---|---|
| **CALL** | **IDCAMS** | **(***options[,dnames][,pageno][,iolist] [,auxlist]***);** |

where:

*options*
specifies a valid set of parameters for the access method services PARM command. If no parameters are to be specified, options should be a halfword of binary zeros. Figure 28 shows the format of the options area.

*dnames*
specifies a list of alternate ddnames for standard data sets used during IDCAMS processing. If standard ddnames are used and this is not the last parameter in the list, dnames should be a halfword of binary zeros. If it is the last parameter, it can be omitted. Figure 28 shows the format of the alternate ddnames list.

*pageno*
specifies a 6-byte field that contains an EBCDIC starting page number for the system output file. If the page number is not specified, but this is not the last parameter in the list, the parameter must be a halfword of binary zeros. If it is the last parameter, it can be omitted. If not specified, the default page number is 1. Figure 28 shows the format of the page number area.

*iolist*
specifies a list of externally controlled data sets and the addresses of corresponding I/O routines. If no external I/O routines are supplied, this parameter can be omitted. Figure 28 shows the format of the I/O list.

*auxlist*
specifies the auxiliary list. Figure 28 shows the format of the auxiliary list.

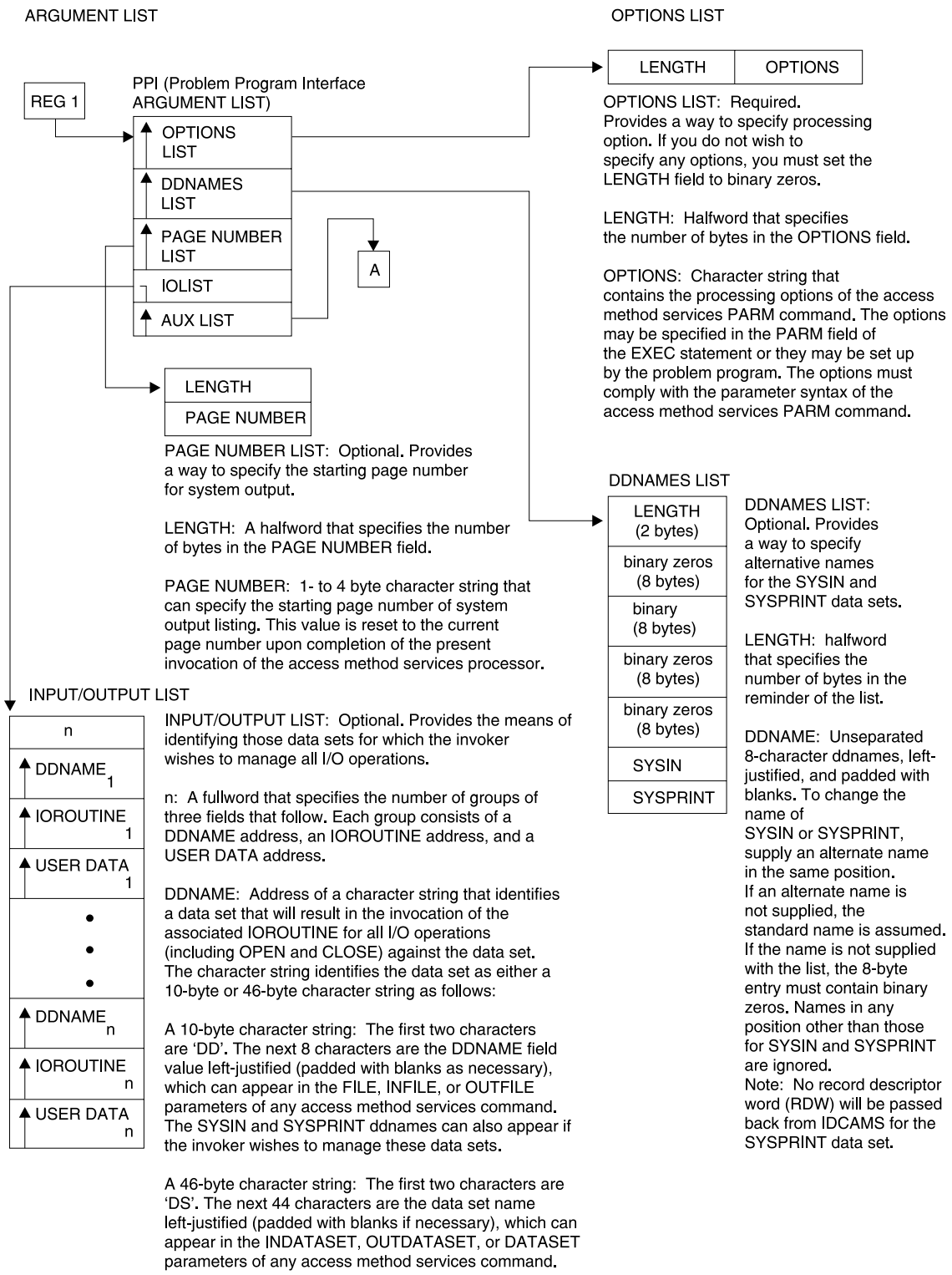
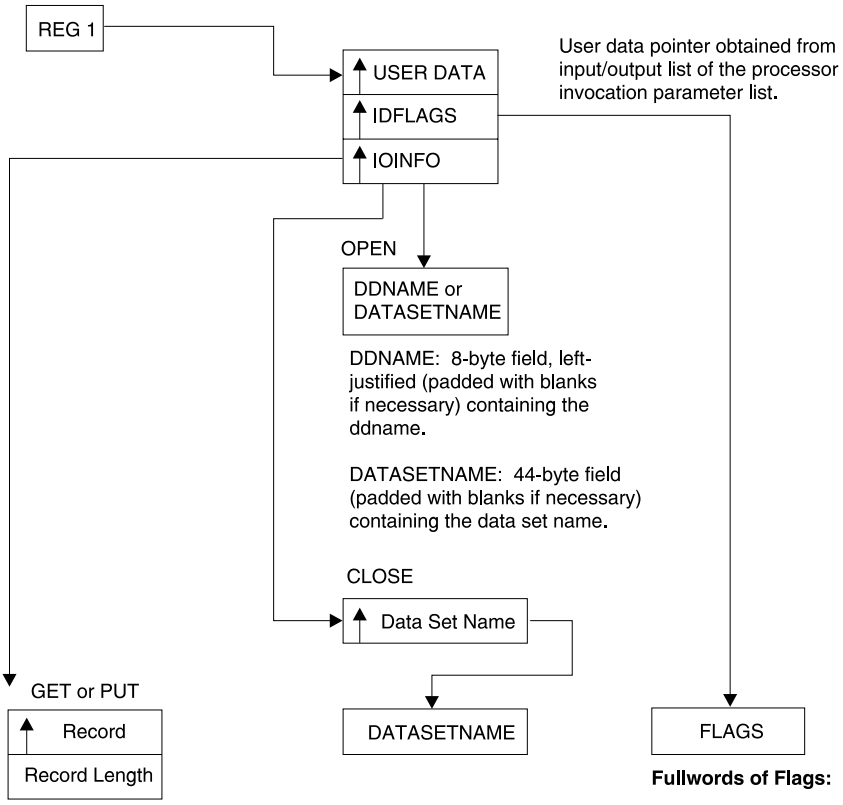



*Figure 28. Processor Invocation Argument List from Your Program (Part 1 of 2)*

### Invoking from Program




For a GET, this information is returned to the processor by the user's I/O routine in the 8-byte area passes to the routine.

For a PUT, the processor gives this information to the user's I/O routine.

Record (GET): Address of retrieved record.

Record Length (GET): Full-word length or retrieved record.

Record (PUT): Address of record to the written.

Record length (PUT): full-word length or record to be written.

DATASETNAME: 44-BYTE fields (padded with blanks if necessary) containing the data set name.

| Fullwords of Flags: | Value or Bit Pattern | Meaning |
|---|---|---|
| Byte 1 (Operation) | X'00' | OPEN |
| | X'04' | CLOSE |
| | X'08' | GET |
| | X'0C' | PUT |
| Byte 2 | 1... .... | OPEN for input |
| | .1.. .... | OPEN for output |
| | ..1. .... | indicates IOINFO contains the address of a ddname on OPEN. |
| | ...1 .... | Indicates IOINFO contains the address of a data set name on OPEN. |
| Bytes 3, 4 (Record type for PUT only) | 0 | Normal data record is to be written. |
| | n | Message serial number converted to binary if IDC message is to be written. (See "User I/O Routine". For a description of this value.) |

*Figure 28. Processor Invocation Argument List from Your Program (Part 2 of 2)*

## Processor Invocation

Figure 28 on page 397 shows the processor invocation argument list as it exists in the user's area. The 24-bit virtual addresses are passed in argument lists, control blocks, buffers, and user exit routines.

Entry and exit to the access method services processor occur through a module of the system adapter. Standard linkage is used; that is, register 1 points to the argument list, register 13 points to a save area, register 14 contains the return address, and register 15 contains the entry point address. On exit from the access method services processor, register 15 contains MAXCC. (See “Processor Condition Codes”.)

The argument list, as shown in Figure 28, can be a maximum of five fullword addresses pointing to strings of data. The last address in the list contains a 1 in the sign field. The first three possible strings of data begin with a 2-byte length field. A null element in the list can be indicated by either an address of zeros or a length of zero.

## Processor Condition Codes

The processor’s condition code is LASTCC, which can be interrogated in the command stream following each functional command. The possible values, their meanings, and examples of causes are:

| Code | Meaning |
|---|---|
| **0(0)** | The function was successful. Informational messages might have been issued. |
| **4(4)** | Some minor problems in executing the complete function were encountered, but it was possible to continue. The results might not be exactly what the user wants, but no permanent harm appears to have been done by continuing. A warning message was issued. |
| **8(8)** | A function could not perform all that was asked of it. The function was completed, but specific details were bypassed. |
| **12(C)** | The entire function could not be done. |
| **16(10)** | Severe error or problem encountered. Remainder of command stream is erased and processor returns condition code 16 to the operating system. |

LASTCC is set by the processor at the completion of each functional command. MAXCC, which can also be interrogated in the command stream, is the highest value of LASTCC thus far encountered.

## User I/O Routines

User I/O routines enable a user to perform all I/O operations for a data set that would normally be handled by the access method services processor. This makes it possible, for instance, to control the command input stream by providing an I/O routine for SYSIN. Standard linkage must be used and standard register convention must be followed. See “Processor Invocation” on page 398 for an explanation of standard linkage.

A user I/O routine is invoked by access method services for all operations against the selected data sets. The identification of the data sets and their associated I/O routines is through the input/output list of the processor invocation parameter list (see Figure 28 on page 397).

When writing a user I/O routine, the user must be aware of three things:

1. The processor handles the user data set as if it were a non-VSAM data set that contains variable-length unblocked records (maximum record length is 32760

bytes) with a physical sequential organization. The processor does not test for the existence of the data set, except for the REPRO command with OUTDATASET.

2. The user must know the data format so that the routine can be coded for the correct type of input and format the correct type of output.
3. Each user routine must handle errors encountered for data sets it is managing and provide a return code to the processor in register 15. The processor uses the return code to determine what it is to do next.

The permissible return codes are:

| Code | Meaning |
|---|---|
| **0(0)** | Operation successful. |
| **4(4)** | End of data for a GET operations. |
| **8(8)** | Error encountered during a GET/PUT operation, but continue processing. |
| **12(C)** | Error encountered during GET/PUT operation; do not allow any further calls (except CLOSE) to this routine. |

Figure 29 on page 401 shows the argument list used in communication between the user I/O routine and the access method services processor. The user I/O routine is invoked by the processor for OPEN, CLOSE, GET, and PUT routines.

The type of operation to be done is indicated with IOFLAGS. The IOINFO field indicates, for OPEN and CLOSE operations, the data set name or ddname of the data set; for GET and PUT operations, the IOINFO field communicates the record length and address.

A user I/O routine for SYSPRINT receives control each time the processor issues a PUT against the SYSPRINT data set. If the PUT has been issued to print an IDC message, the unique message number is passed to the routine with IOFLAGS (see Figure 29 on page 401). Each IDC message is in the form IDCsnnnI or IDCsnnnnI, where:

- s is a code indicating the severity of the problem.
- nnn or nnnn is the message number that is unique across all IDC messages.

The 2-byte message number passed with IOFLAGS is the nnn or nnnn portion of the message converted to binary. If the message is to be suppressed in TSO, the twos complement of the message number are passed.

## VSAM Record-Level Sharing Considerations

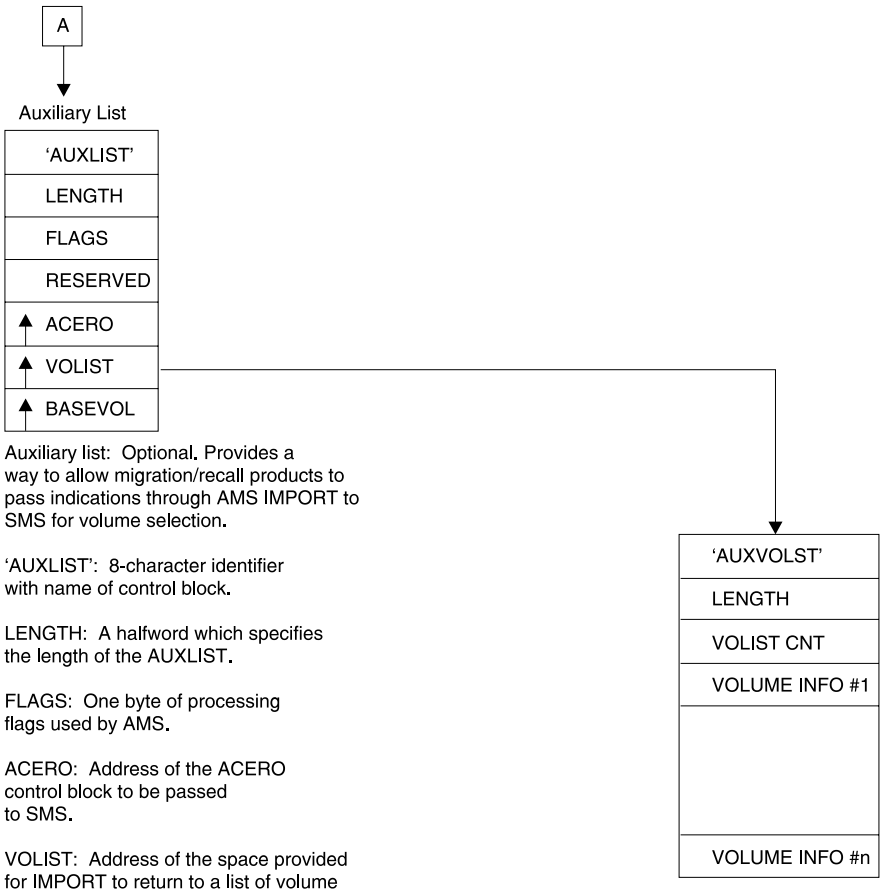
Do not open data sets for record-level sharing (RLS) in the user exit. Access method services expects the exit to use only non-RLS access. Set the high-order bit of byte X’19’ in the ACB to prevent a potential JCL DD override that specifies RLS processing. For example:

```
OI    ACB1+X'19',X'80'              SET ACBNOJCL FLAG
```

If the data set that is being opened is currently open for RLS, the non-RLS open fails. If the data set has previously been opened for RLS, but requires recovery, a non-RLS open for input is allowed. However, open for output fails.

IOROUTINE: Address of the program that is to be invoked to process I/O operation upon the data set Associated with DDNAME. This routine, instead of the processor, is invoked for all operations against The data set. See “USED I/O ROUTINES” in this appendix for linkage and interface conventions Between the IOROUTINE and access method services.

USER DATA: Address of data area user can for any purpose.




Auxiliary list: Optional. Provides a way to allow migration/recall products to pass indications through AMS IMPORT to SMS for volume selection.

‘AUXLIST’: 8-character identifier with name of control block.

LENGTH: A halfword which specifies the length of the AUXLIST.

FLAGS: One byte of processing flags used by AMS.

ACERO: Address of the ACERO control block to be passed to SMS.

VOLIST: Address of the space provided for IMPORT to return to a list of volume name and associations used to satisfy the request.

BASEVOL: Address pointing to the volume serial number of the base cluster for an AIX.

WORKAREA VOLUME LIST: Provides a list large enough to return the information necessary for the object being imported.

‘AUXVOLST’: 8-character identifier with name of control block.

LENGTH: One fullword that specifies the number of bytes in the volume list.

VOLIST CNT: One fullword that specifies the number of VOLSERS returned. For multi-volume import, the volume list can be as large as 118 volume (59 for data component, 59 for index component).

VOLUME INFO: Multiple volume information entries. Each contains two bytes of a six character volume name, return status for the volume, and a character association (D or I).

*Figure 29. Arguments Passed to and from a User I/O Routine*

# Appendix E. IBM Storage Control Units—Caching Models

A cache device is an IBM storage control device that contains high speed storage called subsystem storage. A portion of this storage is called cache and is used to store data between devices and the processor.

The following access method services commands are used to establish and maintain cache devices.

- You can use LISTDATA to:
  - Obtain information about activity within all caching models of the IBM 3880 Storage Control (including the Model 23 with the AJ4/AK4 attachment, feature 3010), and the IBM 3990 Storage Control with cache, as well as the IBM RAMAC® Array Subsystem.
  - You can also obtain information on the 2105 ESS (the Enterprise Storage Server)
  - For both 3880 and 3990 storage controls, you can get two types of reports:
    - Subsystem Counters Report—a record of the counters within the subsystem at the time the report is requested.
    - Subsystem Status Report—a record of the status within the subsystem at the time the report is requested.
  - For 3990 storage controls, you can also get:
    - Remote Access Code—authorization code used for remote maintenance support.
  - For 3990 storage controls with extended function, you can get:
    - Device Status Report—a report of device status with both the channel connection address (CCA) and the director to device connection (DDC) address for each device. This report is useful in determining the state of devices that are, or have been, used in a dual copy pair.
    - Pinned Track to Data Set Cross-Reference Report—a report of pinned tracks in cache and NVS cross-referenced to the data sets involved.
  - For IBM RAMAC Subsystem, you can get:
    - Subsystem Counters Report—a record of the counters within the subsystem at the time the report is requested.
- For the IBM Enterprise Storage Server, you can get:
  - RAID Rank Counters report and the Subsystem Counters report.
- You can use SETCACHE to:
  - Allow or prohibit access to the cache of IBM 3880 Storage Control Models 13, 21, and 23 and IBM 3990 Storage Control with cache.

In a sysplex environment, the LISTDATA and SETCACHE commands support Four-Digit Device Numbers.

- When specifying UNITNUMBER as a subparameter of these commands, you can omit leading zeros and can enter an address up to X'FFFF'.
- When LISTDATA and SETCACHE messages and reports use the Device ID, a four character field with leading zeros is used.
- For specific interface data when using the User Access to Subsystem Status or Counts, refer to *IBM 3990 Storage Control Operations and Recovery Guide, GA32-0253*.

© Copyright IBM Corp. 1973, 2002

- For specific interface data for the IBM RAMAC Array Subsystem when using the User Access to Subsystem Counts, refer to *Using IBM RAMAC Array Subsystem in MVS, VM, or VSE Environment, GC26-7005*.
- The LISTDATA DSTATUS report will list device IDs as four-character fields.

For a complete description of command syntax and parameters as well as for identification of IBM caching devices, see the following books:

- For 3880 storage controls with cache, refer to *Cache Device Administration.*
- For 3990 storage controls, refer to *3990 Storage Control Operations and Recovery Guide.*
- For further information, also see *3990 Planning, Installation, and Storage Administration Guide.*

# Appendix F. DCOLLECT User Exit

This appendix is intended to help you understand the DCOLLECT user exit.

## User Exit Description

DCOLLECT enables you to intercept records after they are created, but before they are written to the output data set. This ability is provided by either IDCDCX1, the default DCOLLECT user exit, or any load module named with the EXITNAME parameter. In this chapter, the term **DCOLLECT user exit** is used to denote either the default exit or a named exit.

The DCOLLECT user exit allows the programmer to enhance, modify, or delete records created by DCOLLECT. If IDCDCX1 is modified, it must be link-edited into the IDCDC01 load module, or applied to the system by the System Modification Program Extended (SMP/E). A separate user exit must be loaded into an APF authorized load library if the EXITNAME parameter is used. All records produced by DCOLLECT, including records created for DFSMShsm are passed to the DCOLLECT user exit before they are written to the output data set.

Use the default exit, IDCDCX1, to provide some standard customization to DCOLLECT. You can use and the EXITNAME parameter for special situations, or testing a new exit for DCOLLECT.

The DCOLLECT user exit should use standard save area conventions, and it should be reentrant. The user exit must return to the caller in the caller's addressing mode.

Each record is passed to the DCOLLECT user exit by placing the length of the record in register 0, and its address in register 1. If the record is modified, the contents of register 0 and register 1 must be updated to reflect the new length and address of the record. The record can be modified in any way by the exit, except that it cannot exceed 32760 bytes. If the user exit extends the record, the supplied record buffer must not be used.

When a record is passed to the DCOLLECT user exit, the user exit has the option of leaving the record unmodified, changing one or more existing fields, adding new fields to the end of the record, or specifying that the record not be written to the output data set.

- To leave the record unmodified, the user exit should set register 15 to 0, and return control to the caller.
- To change any existing fields but not change the length of the record itself, the user exit can overwrite the appropriate field in the record passed to it. Register 15 should be set to 4 to indicate that the record has been modified.
- To add new fields to the end of the record, the user exit should get sufficient storage for a new record buffer that is large enough to hold the original record plus the fields that are to be added. This buffer must reside in storage below 16MB.

  The new fields can be written into the new buffer. Register 0 should be loaded with the length of the new record. Register 1 should contain the address of the new buffer. Register 15 should be set to 4 to indicate that the record has been modified.

© Copyright IBM Corp. 1973, 2002

- To specify that a record not be written to the output data set, the user exit should set register 15 to 12. DCOLLECT will bypass any further processing for this record.

The following is a summary of the register usage at the interface level of the IDCDCX1 user exit:

**Register 0** Contains the length of the current record being processed. This value must be updated if the length of the record changes during exit processing.

**Register 1** Contains the address of the current record being processed. This address must be updated if the address of the record changes during exit processing.

**Register 2** Contains the address of a 100-byte work area. At the first call to the user exit, DCOLLECT sets the work area to zeros. DCOLLECT does not further modify the work area.

The address of this work area is passed to the user exit each time the user exit is called. The user exit uses the work area to store values that are needed for the life of the DCOLLECT job. For example, the work area can contain counters, totals, or the address of an exit-acquired record buffer.

After all processing is complete, DCOLLECT calls the user exit, but does not pass a record. DCOLLECT sets Register 0 to X'0' and register 1 to X'FFFF FFFF'. These settings indicate to the user exit that this is the final call. The user exit then proceeds to clean-up exit-acquired buffers.

**Register 13** Contains the address of a 72-byte register save area. This save area is sufficient to store the program state. We recommend that you use IBM's standard register save area convention.

**Register 14** Contains the return address that should be branched to upon return from the user exit.

**Note:** The caller's registers should be restored before returning to the caller.

**Register 15** Contains the exit return code:

| Code | Description |
|---|---|
| **0** | Write record as is. |
| **4** | Record has been modified or replaced. Write the record pointed to by register 0. |
| **12** | Skip this record. Do not write it to the output data set. |

## User Exit Example

The following is the source code for a sample user exit. This exit will change the storage group to non-SMS if the data set is non-SMS-managed. In addition, this exit will test if the 'A' record contains a high used RBA and a high allocated RBA value of zero. If a value of zero is found in this test, then the record is not written out.

```
IDCDCX1  TITLE 'USER EXIT FOR DCOLLECT - EXAMPLE'
IDCDCX1  CSECT
IDCDCX1  AMODE 24
IDCDCX1  RMODE 24
*********************************************************************
* DESCRIPTIVE NAME: USER EXIT FOR DCOLLECT - EXAMPLE                *
*                                                                   *
* FUNCTION: THIS MODULE TESTS IF A STORAGE GROUP NAME EXISTS IN THE *
*           'D ' RECORD, AND IF NOT, SETS THE STORAGE GROUP NAME TO *
*           A VALUE OF "NON-SMS ". IT ALSO TESTS IF THE 'A ' RECORD *
*           CONTAINS A HURBA AND HARBA VALUE OF ZERO, AND IF SO,    *
*           INDICATES THAT THIS RECORD SHOULD NOT BE WRITTEN TO THE *
*           OUTPUT DATA SET.                                        *
*                                                                   *
* REGISTER CONVENTIONS:                                             *
*     ON ENTRY:  R0  = LENGTH OF RECORD                             *
*                R1  = ADDRESS OF RECORD                            *
*                R2  = 100 BYTE WORK AREA ADDRESS                   *
*                R13 = CALLER'S SAVE AREA ADDRESS                   *
*                R14 = RETURN ADDRESS                               *
*     ON EXIT :  R0  = NEW RECORD LENGTH (IF MODIFIED)              *
*                R1  = NEW RECORD ADDRESS (IF MODIFIED)             *
*                R15 = RETURN CODE                                  *
*                                                                   *
* RETURN CODE VALUES                                                *
*   0 = NO CHANGES MADE. WRITE RECORD TO OUTPUT DATA SET            *
*   4 = CHANGES MADE TO RECORD. WRITE RECORD TO OUTPUT DATA SET     *
*  12 = DO NOT WRITE RECORD TO OUTPUT DATA SET                      *
*                                                                   *
*                                                                   *
* ENTRY POINT: IDCDCX1                                              *
*                                                                   *
* CONTROL BLOCKS REFERENCED:                                        *
*   IDCDOUT - AMS DCOLLECT FUNCTION OUTPUT RECORD FORMATS           *
*                                                                   *
*********************************************************************
*
         DS    0H
         USING *,R15
         B     START
         DC    C'IDCDCX1 '
         DC    C'EXAMPLE 1 '
         DROP  R15
```

*Figure 30. DCOLLECT User Exit Example (Part 1 of 3)*

```
*********************************************************************
* SAVE REGISTERS FROM CALLER                                        *
*********************************************************************
START    STM   R14,R12,12(R13)
         LR    R12,R15
         USING IDCDCX1,R12
         USING DCUOUTH,R1
*
* INITIALIZE THE RETURN REGISTER (R15)
         SLR   R15,R15
*
*********************************************************************
* TEST REG1 FOR A VALUE OF 'FFFFFFFF'X, INDICATING THE FINAL CALL    *
* TO THE USER EXIT. IF FINAL CALL TO EXIT, JUST RETURN TO DCOLLECT.  *
* IF ANY AREAS WERE GETMAINED, THEY WOULD BE FREED AT THIS TIME,     *
* AND ANY OTHER NECESSARY CLEANUP PERFORMED.                         *
*********************************************************************
         SLR   R14,R14
         BCTR  R14,0
         CLR   R1,R14
         BE    EXIT
*
*********************************************************************
* IF THIS IS A 'D ' TYPE RECORD, TEST THE STORAGE GROUP LENGTH FIELD *
* FOR A VALUE OF ZERO. IF ZERO, PUT THE VALUE 'NON_SMS ' IN THE      *
* STORAGE GROUP FIELD. THE TYPES OF RECORDS USED BY DCURCTYP CAN BE  *
* OBTAINED FROM THE MAPPING MACRO ICDOUT FOR USE BY THE CUSTOMER-    *
* DESIGNED EXIT.                                                     *
*********************************************************************
         CLI   DCURCTYP,='D '
         BNE   TEST_A
*
* TEST FOR A STORAGE GROUP FOR THIS DATA SET
         LH    R14,DCDSGLNG
         LTR   R14,R14
         BNZ   EXIT
*
* SET DCDSGLNG TO 8
         LA    R3,8
         STH   R3,DCDSGLNG
* SET DCDSTGRP TO 'NON_SMS '
         MVI   DCDSTGRP+8,C' '
         MVC   DCDSTGRP+9(21),DCDSTGRP+8
         MVC   DCDSTGRP(8),NON_SMS
* INDICATE THAT THE RECORD HAS BEEN MODIFIED
         LA    R15,4
         B     EXIT
*
```

*Figure 30. DCOLLECT User Exit Example (Part 2 of 3)*

```
*********************************************************************
* IF THIS IS AN 'A ' TYPE RECORD, TEST DACHURBA AND DCAHARBA FOR A   *
* VALUE OF ZERO. IF BOTH FIELDS ARE ZERO, THEN SET REGISTER 15 TO    *
* 12, INDICATING THAT THIS RECORD SHOULD NOT BE WRITTEN OUT.         *
* THE TYPES OF RECORDS USED BY DCURCTYP CAN BE OBTAINED FROM THE     *
* MAPPING MACRO ICDOUT FOR USE BY THE CUSTOMER-DESIGNED EXIT.        *
*********************************************************************
TEST_A   DS    0H
         CLI   DCURCTYP,DCUASSOC
         BNE   EXIT
*
* TEST IF DCAHURBA = 0 & DCAHARBA = 0
         L     R3,DCAHURBA
         LTR   R3,R3
         BNZ   EXIT
         L     R14,DCAHARBA
         LTR   R14,R14
         BNZ   EXIT
* DON'T WRITE THIS RECORD OUT TO THE OUTPUT DATA SET
         LA    R15,12
*
EXIT     DS    0H
* RETURN TO DCOLLECT WITH THE RETURN CODE IN REGISTER 15
         L     R14,12(,R13)
         LM    R0,R12,20(R13)
         BR    R14
*
         LTORG
         DS    0D
NON_SMS  DC    CL8'NON_SMS '
R0       EQU   0
R1       EQU   1
R3       EQU   3
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         IDCDOUT
*
         END
```

*Figure 30. DCOLLECT User Exit Example (Part 3 of 3)*

# Appendix G. Interpreting DCOLLECT Output

This appendix contains General-use Programming Interface and Associated Guidance Information.

This appendix is intended to help you to interpret DCOLLECT output.

DCOLLECT provides you with data set information, volume usage information, and information about data sets and storage controlled by DFSMShsm. Running DCOLLECT produces a snapshot of the requested information as it exists at that time. DCOLLECT does not monitor the information continuously. This information can then be used for accounting, planning, statistical, and other purposes.

This appendix has two parts. The first part, “DCOLLECT Output Record Structure” on page 413, shows the structure of the different output records produced by DCOLLECT. The second part, “DCOLLECT Output Record Field Descriptions” on page 444, provides field descriptions for the different output records.

The following output record types are included in this appendix:

| Type | Name |
|---|---|
| **D** | Active Data Set Record |
| **A** | VSAM Association Information |
| **V** | Volume Information |
| **M** | Migrated Data Set Information |
| **B** | Backup Data Set Information |
| **C** | DASD Capacity Planning Information |
| **T** | Tape Capacity Planning Information |
| **DC** | Data Class construct information |
| **SC** | Storage Class construct information |
| **MC** | Management Class construct Information |
| **BC** | Base Configuration Information |
| **SG** | Storage Group construct Information |
| **VL** | Storage Group volume Information |
| **AG** | Aggregate Group Information |
| **DR** | OAM Drive Record Information |
| **LB** | OAM Library Record Information |
| **CN** | Cache Names from the Base Configuration Information |
| **AI** | Accounting Information from the ACS |

The output data set used by DCOLLECT must be created prior to calling the function. It must have a physical sequential organization (PS) and a record format of variable (V) or variable blocked (VB). Using the following guidelines, the primary space for the data set can be estimated:

**Volume list**
: Size of record (336 + 4) * average number of data sets on volume * number of volumes scanned.

**Storage Group list**
: Size of record (260 + 4) * average number of data sets on volume * number of volumes in the each storage group * number of storage groups.

**Migration data**
: Size of record (248 + 4) * number of data sets migrated.

© Copyright IBM Corp. 1973, 2002

**Backup data**
Size of record (228 + 4) * number of data set backup versions.

**Data Class Construct**
Size of record (316 + 4) * number of data class constructs.

**Storage Class Construct**
Size of record (280 + 4) * number of storage class constructs.

**Management Class Construct**
Size of record (308 + 4) * number of management class constructs.

**Storage Group Construct**
Size of record (848 + 4) * number of storage group constructs.

**SMS Managed volumes**
Size of record (440 + 4) * number of SMS managed volumes.

**Base Configuration**
Size of record (928 + 4).

**Aggregate Group Constructs**
Size of record (640 + 4) * number of aggregate group constructs.

**Optical Drives**
Size of record (424 + 4) * number of optical drives.

**Optical Libraries**
Size of record (448 + 4) * number of optical libraries.

**Cache Names**
Size of record (176 + 4) * number of cache names.

**Accounting Information**
Size of record (352 + 4) * number of records.

**Note:** The fields described here are available in a macro form that can be included in an application program. Record formats for the D, A, and V records are mapped by IDCDOUT available in SYS1.MACLIB. Record formats for the M, B, C, and T records are available in ARCUTILP, also available in SYS1.MACLIB.

# DCOLLECT Output Record Structure

*Table 14. DCOLLECT Output Record Structure*

| Offset | Type | Lngth | Name | Description |
|---|---|---|---|---|
| **HEADER PORTION OF DCOLLECT OUTPUT RECORD. EACH DATA SECTION IS PRECEDED BY THIS HEADER.** | | | | |
| 0(X'0') | STRUCTURE | 24 | DCUOUTH | DATA COLLECTION OUTPUT RECORD |
| 0(X'0') | SIGNED | 4 | DCURDW | RECORD DESCRIPTOR WORD |
| 0(X'0') | SIGNED | 2 | DCULENG | LENGTH OF THIS RECORD |
| 2(X'2') | CHARACTER | 2 | * | RESERVED |
| 4(X'4') | CHARACTER | 2 | DCURCTYP | RECORD TYPE FOR THIS RECORD (seeTable 26 on page 438) |
| 6(X'6') | SIGNED | 2 | DCUVERS | VERSION |
| 8(X'8') | CHARACTER | 4 | DCUSYSID | SYSTEM ID FOR THIS OPERATION |
| 12(X'C') | CHARACTER | 8 | DCUTMSTP | TIMESTAMP FIELD |
| 12(X'C') | UNSIGNED | 4 | DCUTIME | TIME IN SMF HEADER FORMAT |
| 16(X'10') | CHARACTER | 4 | DCUDATE | DATE IN SMF FORMAT (CCYYDDDF) |
| 20(X'14') | CHARACTER | 4 | * | RESERVED |
| 24(X'18') | CHARACTER | | DCUDATA | END OF HEADER SECTION |
| **ACTIVE DATA SET INFORMATION (RECORD TYPE “D”)** | | | | |
| 24(X'18') | STRUCTURE | 312 | DCDADSI | ACTIVE DATA SET INFORMATION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 44 | DCDDSNAM | DATA SET NAME |
| 68(X'44') | BITSTRING | 1 | DCDERROR | ERROR INFORMATION FLAG |
| | 1... .... | | DCDEMNGD | SMS-MANAGED INCONSISTENCY |
| | .1.. .... | | DCDEDVVR | DUPLICATE VVR FOUND |
| | ..1. .... | | DCDNOSPC | NO SPACE INFORMATION PROVIDED |
| | ...1 .... | | DCDVSAMI | VSAM INDICATORS INCONSISTENT |
| | .... 1... | | DCDNOFM1 | NO FMT 1 DSCB FOR THIS DATA SET |
| | .... .111 | | * | RESERVED |
| 69(X'45') | BITSTRING | 1 | DCDFLAG1 | INFORMATION FLAG #1 |
| | 1... .... | | DCDRACFD | DATA SET IS RACF-DEFINED |
| | .1.. .... | | DCDSMSM | SMS-MANAGED DATA SET |
| | ..1. .... | | DCDTEMP | TEMPORARY DATA SET |
| | ...1 .... | | DCDPDSE | PARTITIONED DATA SET (EXTENDED) |
| | .... 1... | | DCDGDS | GENERATION DATA GROUP DATA SET |
| | .... .1.. | | DCDREBLK | DATA SET CAN BE REBLOCKED |
| | .... ..1. | | DCDCHIND | CHANGE INDICATOR |
| | .... ...1 | | DCDCKDSI | CHECKPOINT DATA SET INDICATOR |
| 70(X'46') | BITSTRING | 1 | DCDFLAG2 | INFORMATION FLAG #2 |
| | 1... .... | | DCDNOVVR | NO VVR FOR THIS DATA SET |
| | .1.. .... | | DCDINTCG | DATA SET IS AN INTEGRATED CATALOG FACILITY CATALOG |

*Table 14. DCOLLECT Output Record Structure (continued)*

| Offset | Type | Lngth | Name | Description |
|---|---|---|---|---|
| | ..1. .... | | DCDINICF | DATA SET IS CATALOGED IN INTEGRATED CATALOG FACILITY CATALOG |
| | ...1 .... | * | RESERVED | |
| | .... 1... | | DCDALLFG | ALLOCATED SPACE RETURNED |
| | .... .1.. | | DCDUSEFG | USED SPACE INFORMATION RETURNED |
| | .... ..1. | | DCDSECFG | SECONDARY SPACE INFORMATION RETURNED |
| | .... ...1 | | DCDNMBFG | UNUSABLE SPACE RETURNED |
| 71(X'47') | BITSTRING | 1 | DCDFLAG3 | INFORMATION FLAG #3 |
| | 1... .... | | DCDPDSEX | POSIX FILE SYSTEM FILE |
| | .1.. .... | | DCDSTRP | DATA SET IS IN EXTENDED FORMAT |
| | ..1. .... | | DCDDDMEX | DDM INFO EXISTS FOR THIS DATA SET |
| | ...1 1111 | | * | RESERVED |
| 72(X'48') | CHARACTER | 2 | * | RESERVED |
| 74(X'4A') | BITSTRING | 2 | DCDDSORG | DATA SET ORGANIZATION |
| 74(X'4A') | BITSTRING | 2 | DCDDSOR0 | DATA SET ORGANIZATION BYTE 0 |
| | 1... .... .... .... | | DCDDSGIS | IS INDEXED SEQUENTIAL ORG |
| | .1.. .... .... .... | | DCDDSGPS | PS PHYSICAL SEQUENTIAL ORG |
| | ..1. .... .... .... | | DCDDSGDA | DA DIRECT ORGANIZATION |
| | ...1 11.. .... .... | | * | RESERVED |
| | .... ..1. .... .... | | DCDDSGPO | PO PARTITIONED ORGANIZATION |
| | .... ...1 .... .... | | DCDDSGU | U UNMOVABLE DATA SET |
| 75(X'4B') | BITSTRING | 1 | DCDDSOR1 | DATA SET ORGANIZATION BYTE 1 |
| | 1... .... | | DCDDSGGS | GS GRAPHICS ORGANIZATION |
| | .111 .... | | * | RESERVED |
| | .... 1... | | DCDDSGVS | VS VSAM DATA SET |
| | .... .111 | | * | RESERVED |
| 76(X'4C') | BITSTRING | 1 | DCDRECRD | RECORD FORMAT BYTE |
| | 11.. .... | | DCDRECFM | RECORD FORMAT BITS (see Table 26 on page 438) |
| | ..1. .... | | DCDRECFT | TRACK OVERFLOW |
| | ...1 .... | | DCDRECFB | BLOCKED RECORDS |
| | .... 1... | | DCDRECFS | STANDARD BLOCKS(F) OR SPANNED(V) |
| | .... .1.. | | DCDRECFA | ANSI CONTROL CHARACTER |
| | .... ..1. | | DCDRECFC | MACHINE CONTROL CHARACTER |
| | .... ...1 | | * | RESERVED |
| 77(X'4D') | UNSIGNED | 1 | DCDNMEXT | NUMBER OF EXTENTS OBTAINED |
| 78(X'4E') | CHARACTER | 6 | DCDVOLSR | VOLUME SERIAL NUMBER |
| 84(X'54') | SIGNED | 2 | DCDBKLNG | BLOCK LENGTH |
| 86(X'56') | SIGNED | 2 | DCDLRECL | RECORD LENGTH |
| 88(X'58') | SIGNED | 4 | DCDALLSP | SPACE ALLOCATED TO DATA SET |

*Table 14. DCOLLECT Output Record Structure (continued)*

| Offset | Type | Lngth | Name | Description |
|---|---|---|---|---|
| 92(X'5C') | SIGNED | 4 | DCDUSESP | SPACE USED BY DATA SET |
| 96(X'60') | SIGNED | 4 | DCDSCALL | SECONDARY ALLOCATION |
| 100(X'64') | SIGNED | 4 | DCDNMBLK | NUMBER OF BYTES UNUSABLE IN BLOCKS |
| 104(X'68') | CHARACTER | 4 | DCDCREDT | CREATION DATE (*yyyyddd* F) |
| 108(X'6C') | CHARACTER | 4 | DCDEXPDT | EXPIRATION DATE (*yyyyddd* F) |
| 112(X'70') | CHARACTER | 4 | DCDLSTRF | DATE LAST REFERENCED (*yyyyddd* F) |
| 116(X'74') | CHARACTER | 6 | DCDDSSER | DATA SET SERIAL NUMBER |
| 122(X'7A') | CHARACTER | 2 | DCDVOLSQ | VOLUME SEQUENCE NUMBER |
| 124(X'7C') | CHARACTER | 8 | DCDLBKDT | LAST BACKUP TIME AND DATE |
| 132(X'84') | CHARACTER | 32 | DCDDCLAS | |
| 132(X'84') | SIGNED | 2 | DCDDCLNG | DATA CLASS NAME LENGTH |
| 134(X'86') | CHARACTER | 30 | DCDDATCL | DATA CLASS NAME |
| 164(X'A4') | CHARACTER | 32 | DCDSCLAS | |
| 164(X'A4') | SIGNED | 2 | DCDSCLNG | STORAGE CLASS NAME LENGTH |
| 166(X'A6') | CHARACTER | 30 | DCDSTGCL | STORAGE CLASS NAME |
| 196(X'C4') | CHARACTER | 32 | DCDMCLAS | |
| 196(X'C4') | SIGNED | 2 | DCDMCLNG | MANAGEMENT CLASS NAME LENGTH |
| 198(X'C6') | CHARACTER | 30 | DCDMGTCL | MANAGEMENT CLASS NAME |
| 228(X'E4') | CHARACTER | 32 | DCDSTOGP | |
| 228(X'E4') | SIGNED | 2 | DCDSGLNG | STORAGE GROUP NAME LENGTH |
| 230(X'E6') | CHARACTER | 30 | DCDSTGRP | STORAGE GROUP NAME |
| 260(X'104') | CHARACTER | 2 | DCDCCSID | CODED CHARACTER SET IDENTIFIER |
| 262(X'106') | CHARACTER | 2 | * | RESERVED |
| 264(X'108') | CHARACTER | 8 | DCDUDSIZ | USER DATA SIZE (64 bit unsigned binary number) |
| 272(X'110') | CHARACTER | 8 | DCDCUDSZ | COMPRESSED DATA SET SIZE (64 bit unsigned binary number) |
| 280(X'118') | BITSTRING | 2 | DCDEXFLG | COMPRESSION FLAGS |
| | 1... .... | | DCDBDSZ | DATA SIZES THAT ARE NOT VALID |
| 282(X'11A') | UNSIGNED | 2 | DCDSCNT | STRIPE COUNT |
| 284(X'11C') | SIGNED | 4 | DCDOVERA | OVER-ALLOCATED SPACE |
| 288(X'120') | CHARACTER | 32 | DCDACCT | ACCOUNT INFORMATION |
| 320(X'140') | CHARACTER | 16 | * | RESERVED |
| 336(X'142') | CHARACTER | | DCDADSIE | END OF DCUDSET<br>**Note:** DCDDCLAS, DCDSCLAS, DCDMCLAS, DCDSTOGP and DCDACCT are not returned for AIX. |
| **VSAM BASE CLUSTER ASSOCIATION INFORMATION (RECORD TYPE "A")** | | | | |
| 24(X'18') | STRUCTURE | 180 | DCASSOC | VSAM BASE CLUSTER ASSOCIATIONS (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 44 | DCADSNAM | DATA SET NAME |
| 68(X'44') | CHARACTER | 44 | DCAASSOC | BASE CLUSTER NAME |

*Table 14. DCOLLECT Output Record Structure (continued)*

| Offset | Type | Lngth | Name | Description |
|---|---|---|---|---|
| 112(X'70') | BITSTRING | 1 | DCAFLAG1 | VSAM INFORMATION FLAG #1 |
| | 1... .... | | DCAKSDS | KEY-SEQUENCED DATA SET |
| | .1.. .... | | DCAESDS | ENTRY-SEQUENCED DATA SET |
| | ..1. .... | | DCARRDS | RELATIVE RECORD DATA SET |
| | ...1 .... | | DCALDS | LINEAR DATA SET |
| | .... 1... | | DCAKRDS | KEY RANGE DATA SET |
| | .... .1.. | | DCAAIX | ALTERNATE INDEX DATA SET |
| | .... ..1. | | DCADATA | VSAM DATA COMPONENT |
| | .... ...1 | | DCAINDEX | VSAM INDEX COMPONENT |
| 113(X'71') | BITSTRING | 1 | DCAFLAG2 | VSAM INFORMATION FLAG #2 |
| | 1... .... | | DCAKR1ST | 1ST SEGMENT OF KR DATA SET |
| | .1.. .... | | DCAIXUPG | ALTERNATE INDEX W/ UPGRADE |
| | ..1. .... | | DCAVRRDS | VARIABLE LENGTH RELATIVE RECORD DATA SET |
| | ...1 .... | | DCANSTAT | NO VSAM STATISTICS FOR THIS RECORD |
| | .... 1... | | DCASRCI | RBA IS CI NUMBER |
| | .... .1.. | | DCAG4G | EXTENDED ADDRESSABILITY |
| | .... ..11 | | * | RESERVED |
| 114(X'72') | CHARACTER | 2 | * | RESERVED |
| 116(X'74') | UNSIGNED | 4 | DCAHURBA | HIGH USED RELATIVE BYTE ADDRESS |
| 120(X'78') | UNSIGNED | 4 | DCAHARBA | HIGH ALLOCATED RELATIVE BYTE ADDRESS |
| 124(X'7C') | SIGNED | 4 | DCANLR | NUMBER OF LOGICAL RECORDS |
| 128(X'80') | SIGNED | 4 | DCADLR | NUMBER OF DELETED RECORDS |
| 132(X'84') | SIGNED | 4 | DCAINR | NUMBER OF INSERTED RECORDS |
| 136(X'88') | SIGNED | 4 | DCAUPR | NUMBER OF UPDATED RECORDS |
| 140(X'8C') | SIGNED | 4 | DCARTR | NUMBER OF RETRIEVED RECORDS |
| 144(X'90') | SIGNED | 4 | DCAASP | BYTES OF FREESPACE IN DATA SET |
| 148(X'94') | SIGNED | 4 | DCACIS | NUMBER OF CONTROL INTERVAL (CI) SPLITS |
| 152(X'98') | SIGNED | 4 | DCACAS | NUMBER OF CONTROL AREA SPLITS |
| 156(X'9C') | SIGNED | 4 | DCAEXC | NUMBER OF EXCPs |
| 160(X'A0') | SIGNED | 2 | DCARKP | RELATIVE KEY POSITION |
| 162(X'A2') | SIGNED | 2 | DCAKLN | KEY LENGTH |
| 164(X'A4') | CHARACTER | 8 | DCAHARBC | HIGH USED RBA CALCULATED FROM CI |
| 172(X'AC') | CHARACTER | 8 | DCAHURBC | HIGH ALLOCATED RBA CALCULATED FROM CI |
| 180(X'B4') | SIGNED | 4 | DCACISZ | NUMBER OF BYTES IN A CI |
| 184(X'B8') | SIGNED | 4 | DCACACI | NUMBER OF CIs IN A CA |
| 188(X'BC') | CHARACTER | 16 | * | RESERVED |
| 204(X'CC') | CHARACTER | | DCASSOCE | END OF DCASSOC |
| **VOLUME INFORMATION (RECORD TYPE "V")** | | | | |

*Table 14. DCOLLECT Output Record Structure (continued)*

| Offset | Type | Lngth | Name | Description |
|---|---|---|---|---|
| 24(X'18') | STRUCTURE | 112 | DCVVOLI | VOLUME INFORMATION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 6 | DCVVOLSR | VOLUME SERIAL NUMBER |
| 30(X'1E') | BITSTRING | 1 | DCVFLAG1 | INFORMATION FLAG #1 |
| | 11.. .... | | DCVINXST | INDEX STATUS |
| | 1... .... | | DCVINXEX | INDEXED VTOC EXISTS |
| | .1.. .... | | DCVINXEN | INDEXED VTOC IS ENABLED |
| | ..11 1... | | DCVUSATR | USE ATTRIBUTE |
| | ..1. .... | | DCVUSPVT | PRIVATE |
| | ...1 .... | | DCVUSPUB | PUBLIC |
| | .... 1... | | DCVUSSTO | STORAGE |
| | .... .1.. | | DCVSHRDS | DEVICE IS SHAREABLE |
| | .... ..11 | | DCVPHYST | PHYSICAL STATUS (see Table 26 on page 438) |
| 31(X'1F') | BITSTRING | 1 | DCVERROR | ERROR INFORMATION FLAG |
| | 1... .... | | DCVEVLCP | ERROR CALCULATING VOL CAPACITY |
| | .1.. .... | | DCVEBYTK | ERROR CALCULATING BYTES/TRK |
| | ..1. .... | | DCVELSPC | ERROR DURING LSPACE PROCESSING |
| | ...1 1111 | | * | RESERVED |
| 32(X'20') | CHARACTER | 3 | * | RESERVED |
| 35(X'23') | UNSIGNED | 1 | DCVPERCT | PERCENT FREE SPACE ON VOLUME |
| 36(X'24') | UNSIGNED | 4 | DCVFRESP | FREE SPACE ON VOLUME (IN KB) |
| 40(X'28') | UNSIGNED | 4 | DCVALLOC | ALLOCATED SPACE ON VOL (IN KB) |
| 44(X'2C') | UNSIGNED | 4 | DCVVLCAP | TOTAL CAPACITY OF VOL (IN KB) |
| 48(X'30') | SIGNED | 4 | DCVFRAGI | FRAGMENTATION INDEX |
| 52(X'34') | UNSIGNED | 4 | DCVLGEXT | LARGEST EXTENT ON VOLUME |
| 56(X'38') | SIGNED | 4 | DCVFREXT | NUMBER OF FREE EXTENTS |
| 60(X'3C') | SIGNED | 4 | DCVFDSCB | FREE DSCBS IN VTOC |
| 64(X'40') | SIGNED | 4 | DCVFVIRS | FREE VIRS |
| 68(X'44') | CHARACTER | 8 | DCVDVTYP | DEVICE TYPE |
| 76(X'4C') | UNSIGNED | 2 | DCVDVNUM | DEVICE NUMBER |
| 78(X'4E') | CHARACTER | 2 | * | RESERVED |
| 80(X'50') | CHARACTER | 32 | DCVSTGGP | |
| 80(X'50') | SIGNED | 2 | DCVSGLNG | STORAGE GROUP NAME LENGTH |
| 82(X'52') | CHARACTER | 30 | DCVSGTCL | STORAGE GROUP NAME |
| 112(X’70’) | CHARACTER | 8 | DCVDPTYP | PHYSICAL DEVICE TYPE |
| 120(X’78’) | CHARACTER | 16 | * | RESERVED |
| 136(X'88') | CHARACTER | | DCVVOLIE | END IF DCVVOLI |
| **MIGRATED DATA SET INFORMATION (RECORD TYPE “M”)** | | | | |
| 24 (18) | CHARACTER | 224 | UMMDSI | MIGRATED DATA SET INFORMATION (DEFINED ON DCUDATA) |

*Table 14. DCOLLECT Output Record Structure (continued)*

| Offset | Type | Lngth | Name | Description |
|---|---|---|---|---|
| 24 (18) | CHARACTER | 44 | UMDSNAM | USER DATA SET NAME |
| 68 (44) | BITSTRING<br>11.. ....<br>..1. ....<br>...1 ....<br><br>.... XXXX | 1 | UMFLAG1<br>UMLEVEL<br>UMCHIND<br>UMSDSP<br><br>* | INFORMATION FLAG 1<br>MIGRATED LEVEL (see Table 26 on page 438)<br>CHANGED-SINCE-LAST-BACKUP INDICATOR<br>SMALL DATA SET PACKING (SDSP)<br>MIGRATED DATA SET<br>RESERVED |
| 69 (45) | CHARACTER | 1 | UMDEVCL | DEVICE CLASS OF THE MIGRATION VOLUME (see Table 26 on page 438) |
| 70 (46) | CHARACTER | 2 | UMDSORG | DATA SET ORGANIZATION AT TIME OF MIGRATION |
| 72 (48) | SIGNED | 4 | UMDSIZE | MIGRATION COPY DATA SET SIZE IN KILOBYTES |
| 76 (4C) | CHARACTER | 8 | UMMDATE | TIMESTAMP FIELD |
| 76 (4C) | CHARACTER | 4 | UMTIME | MIGRATED TIME (*hhmmssth* FORMAT) |
| 80 (50) | CHARACTER | 4 | UMDATE | MIGRATED DATE (*yyyyddd* F FORMAT) |
| 84 (54) | CHARACTER | 96 | UMCLASS | |
| 84 (54) | CHARACTER | 32 | UMDCLAS | |
| 84 (54) | SIGNED | 2 | UMDCLNG | LENGTH OF DATA CLASS NAME |
| 86 (56) | CHARACTER | 30 | UMDATCL | DATA CLASS NAME |
| 116 (74) | CHARACTER | 32 | UMSCLAS | |
| 116 (74) | SIGNED | 2 | UMSCLNG | LENGTH OF STORAGE CLASS NAME |
| 118 (76) | CHARACTER | 30 | UMSTGCL | STORAGE CLASS NAME |
| 148 (94) | CHARACTER | 32 | UMMCLAS | |
| 148 (94) | SIGNED | 2 | UMMCLNG | LENGTH OF MANAGEMENT CLASS NAME |
| 150 (96) | CHARACTER | 30 | UMMGTCL | MANAGEMENT CLASS NAME |
| 180 (B4) | BITSTRING | 1 | UMRECRD | RECORD FORMAT OF THIS BYTE |
| 181 (B5) | BITSTRING<br>1... ....<br>.1.. ....<br>..1. ....<br>...1 ....<br>.... XXXX | 1 | UMRECOR<br>UMESDS<br>UMKSDS<br>UMLDS<br>UMRRDS<br>* | VSAM ORGANIZATION OF THIS DATA SET<br>ENTRY-SEQUENCED DATA SET<br>KEY-SEQUENCED DATA SET<br>LINEAR DATA SET<br>RELATIVE-RECORD DATA SET<br>RESERVED |
| 182 (B6) | CHARACTER | 2 | UMBKLNG | BLOCK LENGTH OF THIS DATA SET |
| 184 (B8) | BITSTRING<br>1... ....<br>.1.. ....<br>..1. ....<br><br>...1 ....<br><br>.... 1...<br>.... .1..<br>.... ..11 | 1 | UMFLAG2<br>UMRACFD<br>UMGDS<br>UMREBLK<br><br>UMPDSE<br><br>UMSMSM<br>UMCOMPR<br>* | INFORMATION FLAG 2<br>RACF-INDICATED DATA SET<br>IF SET TO 1, GENERATION GROUP DATA SET[1]<br>IF SET TO 1, SYSTEM-REBLOCKABLE DATA SET[1]<br>IF SET TO 1, PARTITIONED DATA SET EXTENDED[1]<br>IF SET TO 1, SMS-MANAGED DATA SET.<br>IF SET TO 1, COMPRESSED DATA SET.<br>RESERVED<br>**Note:** [1]Only valid when the dataset is SMS-managed |

*Table 14. DCOLLECT Output Record Structure (continued)*

| Offset | Type | Lngth | Name | Description |
|---|---|---|---|---|
| 185 (B9) | CHARACTER | 1 | * | RESERVED |
| 186 (BA) | SIGNED | 2 | UMNMIG | NUMBER OF MIGRATIONS FOR THIS DATA SET |
| 188 (BC) | SIGNED | 4 | UMALLSP | SPACE ALLOCATED IN KILOBYTES |
| 192 (C0) | SIGNED | 4 | UMUSESP | SPACE USED IN KILOBYTES |
| 196 (C4) | SIGNED | 4 | UMRECSP | RECALL SPACE ESTIMATE IN KILOBYTES |
| 200 (C8) | CHARACTER | 4 | UMCREDT | CREATION DATE (*yyyyddd* F FORMAT) |
| 204 (CC) | CHARACTER | 4 | UMEXPDT | EXPIRATION DATE (*yyyyddd* F FORMAT) |
| 208 (D0) | CHARACTER | 8 | UMLBKDT | DATE OF LAST BACKUP (STCK FORMAT CONSISTENT WITH DCDLBKDT)[1]<br>[1]Only valid when the dataset is SMS-managed |
| 216 (D8) | CHARACTER | 4 | UMLRFDT | DATE LAST REFERENCED (*yyyyddd* F FORMAT) |
| 220 (DC) | SIGNED | 4 | UM_USER_ DATASIZE | DATA-SET SIZE, IN KB, IF NOT COMPRESSED |
| 224 (E0) | SIGNED | 4 | UM_COMP_ DATASIZE | COMPRESSED DATA-SET SIZE, IN KB. VALID WHEN UMCOMPR SET. |
| 228 (E4) | CHARACTER | 6 | UMFRVOL | THE FIRST SOURCE VOLUME SERIAL OF THE MIGRATED DATA |
| 234 (EA) | CHARACTER | 14 | * | RESERVED SPACE |
| 248 (F8) | CHARACTER | | UMMDSIE | END OF DCUMCDS |
| **BACKUP DATA SET INFORMATION (RECORD TYPE "B")** | | | | |
| 24 (0) | CHARACTER | 184 | UBBDSI | BACKUP DATA SET INFORMATION (DEFINED ON DCUDATA) |
| 24 (0) | CHARACTER | 44 | UBDSNAM | USER DATA SET NAME |
| 68 (44) | BITSTRING<br>1... ....<br><br>.1.. ....<br>..1. ....<br>...1 ....<br>.... 1....<br><br>.... .111 | 1 | UBFLAG1<br>UBINCAT<br><br>UBNOENQ<br>UBBWO<br>UBNQN1<br>UBNQN2<br><br>* | INFORMATION FLAG 1<br>BACKUP VERSION OF A CATALOGED DATA SET<br>NO DFSMShsm ENQUEUE<br>BACKUP-WHILE-OPEN CANDIDATE<br>ENQ ATTEMPTED, BUT FAILED<br>ENQ ATTEMPTED BUT FAILED, BACKUP RETRIED, AND ENQ FAILED AGAIN<br>RESERVED |
| 69 (45) | CHARACTER | 1 | UBDEVCL | DEVICE CLASS OF BACKUP VOLUME (see Table 26 on page 438) |
| 70 (46) | CHARACTER | 2 | UBDSORG | DATA SET ORGANIZATION |
| 72 (48) | SIGNED | 4 | UBDSIZE | BACKUP VERSION SIZE IN KILOBYTES |
| 76 (4C) | CHARACTER | 8 | UBBDATE | BACKUP DATE/TIME |
| 76 (4C) | CHARACTER | 4 | UBTIME | BACKUP TIME (*hhmmssth* FORMAT) |
| 80 (50) | CHARACTER | 4 | UBDATE | BACKUP DATE (*yyyyddd* F FORMAT) |
| 84 (54) | CHARACTER | 96 | UBCLASS | SMS CLASS INFORMATION |
| 84 (54) | CHARACTER | 32 | UBDCLAS | DATA CLASS WHEN BACKUP MADE |
| 84 (54) | SIGNED | 2 | UBDCLNG | LENGTH OF DATA CLASS NAME |
| 86 (56) | CHARACTER | 30 | UBDATCL | DATA CLASS NAME |

*Table 14. DCOLLECT Output Record Structure (continued)*

| Offset | Type | Lngth | Name | Description |
|---|---|---|---|---|
| 116 (74) | CHARACTER | 32 | UBSCLAS | STORAGE CLASS WHEN BACKUP MADE |
| 116 (74) | SIGNED | 2 | UBSCLNG | LENGTH OF STORAGE CLASS NAME |
| 118 (76) | CHARACTER | 30 | UBSTGCL | STORAGE CLASS NAME |
| 148 (94) | CHARACTER | 32 | UBMCLAS | MANAGEMENT CLASS WHEN BACKUP MADE |
| 148 (94) | SIGNED | 2 | UBMCLNG | LENGTH OF MANAGEMENT CLASS NAME |
| 150 (96) | CHARACTER | 30 | UBMGTCL | MANAGEMENT CLASS NAME |
| 180 (B4) | BITSTRING | 1 | UBRECRD | RECORD FORMAT BYTE OF THIS DATA SET |
| 181 (B5) | BITSTRING<br>1... ....<br>.1.. ....<br>..1. ....<br>...1 ....<br>.... 1111 | 1 | UBRECOR<br>UBESDS<br>UBKSDS<br>UBLDS<br>UBRRDS<br>* | VSAM ORGANIZATION OF THIS DATA SET<br>ENTRY-SEQUENCED DATA SET<br>KEY-SEQUENCED DATA SET<br>LINEAR DATA SET<br>RELATIVE-RECORD DATA SET<br>RESERVED |
| 182 (B6) | CHARACTER | 2 | UBBKLNG | BLOCK LENGTH OF THIS DATA SET |
| 184 (B8) | BITSTRING<br>1... ....<br>.1.. ....<br>..1. ....<br>...1 ....<br>.... 1...<br>.... .1..<br>.... ..11 | 1 | UBFLAG2<br>UBRACFD<br>UBGDS<br>UBREBLK<br>UBPDSE<br>UBSMSM<br>UBCOMPR<br>* | INFORMATION FLAG 2<br>RACF-INDICATED DATA SET<br>IF SET TO 1, GENERATION GROUP DATASET[1]<br>IF SET TO 1, SYSTEM-REBLOCKABLE DATA SET[1]<br>IF SET TO 1, PARTITIONED DATA SET EXTENDED (PDSE)[1]<br>IF SET TO 1, SMS-MANAGED DATA SET AT TIME OF BACKUP<br>IF SET TO 1, COMPRESSED DATA SET<br>RESERVED<br>**Note:** [1]Only valid when the dataset is SMS-managed |
| 185 (B9) | CHARACTER | 3 | * | RESERVED |
| 188 (BC) | SIGNED | 4 | UBALLSP | SPACE ALLOCATED IN KILOBYTES |
| 192 (C0) | SIGNED | 4 | UBUSESP | SPACE USED IN KILOBYTES |
| 196 (C4) | SIGNED | 4 | UBRECSP | RECOVERY SPACE ESTIMATE IN KILOBYTES |
| 200 (C8) | SIGNED | 4 | UB_USER_<br>DATASIZE | VALID WHEN UBCOMPR SET, VALUE IS DATA-SET SIZE, IN KB, IF NOT COMPRESSED |
| 204 (CC) | SIGNED | 4 | UB_COMP_<br>DATASIZE | VALID WHEN UBCOMPR SET, THIS VALUE IS ACTUALCOMPRESSED DATA-SET SIZE, IN KB |
| 208 (D0) | CHARACTER | 6 | UBFRVOL | THE FIRST SOURCE VOLUME SERIAL OF THE BACKUP DATA |
| 214 (D6) | CHARACTER | 14 | * | RESERVED |
| 228 (E4) | CHARACTER | | UBBDSIE | END OF DCUBCDS |
| **DASD CAPACITY PLANNING INFORMATION (RECORD TYPE "C")** | | | | |
| 24 (18) | CHARACTER | 23 | UCCAPD | DASD CAPACITY PLANNING RECORD (DEFINED ON DCUDATA) |
| 24 (18) | CHARACTER | 6 | UCVOLSR | VOLUME SERIAL NUMBER |
| 30 (1E) | CHARACTER | 4 | UCCOLDT | DATE THE STATISTICAL DATA WAS COLLECTED BY DFSMShsm FOR THE VOLUME (*yyyyddd* F FORMAT) |

*Table 14. DCOLLECT Output Record Structure (continued)*

| Offset | Type | Lngth | Name | Description |
|---|---|---|---|---|
| 34 (22) | BITSTRING<br>11.. ....<br><br>..11 1111 | 1 | UCFLAG1<br><br>UCLEVEL<br>* | INFORMATION FLAG 1<br>LEVEL OF VOLUME (L0, L1; see Table 26 on page 438)<br>RESERVED |
| 35 (23) | CHARACTER | 1 | * | RESERVED |
| 36 (24) | SIGNED | 4 | UCTOTAL | TOTAL CAPACITY OF VOLUME IN KILOBYTES |
| 40 (28) | CHARACTER | 7 | UCOCCUP | |
| 40 (28) | UNSIGNED | 1 | UCTGOCC | SPECIFIED TARGET OCCUPANCY OF VOLUME |
| 41 (29) | UNSIGNED | 1 | UCTROCC | SPECIFIED TRIGGER OCCUPANCY OF VOLUME |
| 42 (2A) | UNSIGNED | 1 | UCBFOCC | OCCUPANCY OF VOLUME BEFORE PROCESSING |
| 43 (2B) | UNSIGNED | 1 | UCAFOCC | OCCUPANCY OF VOLUME AFTER PROCESSING (0 IF NOT PROCESSED) |
| 44 (2C) | UNSIGNED | 1 | UCNOMIG | PERCENTAGE OF VOLUME DATA NOT MIGRATED BUT ELIGIBLE TO MIGRATE (EXCESS ELIGIBLE) |
| 45 (2D) | UNSIGNED | 1 | UCNINTV | NUMBER OF TIMES INTERVAL MIGRATION WAS RUN AGAINST THE VOLUME |
| 46 (2E) | UNSIGNED | 1 | UCINTVM | NUMBER OF TIMES TARGET OCCUPANCY WAS MET FOR THE VOLUME DURING INTERVAL MIGRATION |
| 47 (2F) | CHARACTER | | UCCAPDE | END OF DCCCAPD |
| **TAPE CAPACITY PLANNING INFORMATION (RECORD TYPE “T”)** | | | | |
| 24 (18) | STRUCTURE | 16 | UTCAPT | TAPE CAPACITY PLANNING RECORD (DEFINED ON DCUDATA) |
| 24 (18) | CHARACTER | 1 | UTSTYPE | TYPE OF TAPE CAPACITY PLANNING RECORD (see Table 26 on page 438) |
| 25 (19) | CHARACTER | 3 | * | RESERVED |
| 28 (1C) | SIGNED | 4 | UTFULL | NUMBER OF FULL TAPE VOLUMES |
| 32 (20) | SIGNED | 4 | UTPART | NUMBER OF PARTIALLY FILLED TAPE VOLUMES |
| 36 (24) | SIGNED | 4 | UTEMPTY | NUMBER OF EMPTY TAPE VOLUMES |
| 40 (28) | CHARACTER | | UTCAPTE | END PF DCTCAPT |

The following records are generated when SMSDATA is specified:

| Type | Description |
|---|---|
| **DC** | Data Class construct information |
| **SC** | Storage Class construct information |
| **MC** | Management Class construct Information |
| **BC** | Base Configuration Information |
| **SG** | Storage Group construct Information |
| **VL** | Storage Group volume Information |
| **AG** | Aggregate Group Information |
| **DR** | OAM Drive Record Information |

**LB** OAM Library Record Information
**CN** Cache Names from the Base Configuration Information
**AI** Accounting Information from the ACS

*Table 15. DCOLLECT Data Class Definition (Record Type 'DC')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **DATA CLASS CONSTRUCT INFORMATION (RECORD TYPE 'DC')** | | | | |
| 24(X'18') | STRUCTURE | 292 | DDCDATA | DATA CLASS DEFINITION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 32 | DDCNMFLD | SPACE FOR NAME AND LENGTH |
| 24(X'18') | SIGNED | 2 | DDCNMLEN | LENGTH OF NAME |
| 26(X'1A') | CHARACTER | 30 | DDCNAME | NAME OF DATA CLASS |
| 56(X'38') | CHARACTER | 8 | DDCUSER | USERID OF LAST UPDATER |
| 64(X'40') | CHARACTER | 10 | DDCDATE | DATE OF LAST UPDATE |
| 74(X'4A') | CHARACTER | 2 | * | RESERVED |
| 76(X'4C') | CHARACTER | 8 | DDCTIME | TIME OF LAST UPDATE |
| 84(X'54') | CHARACTER | 120 | DDCDESC | DESCRIPTION |
| **DATA CLASS PARAMETERS SPECIFICATION BITS** | | | | |
| 204(X'CC') | CHARACTER | 4 | DDCSPEC | |
| 204(X'CC') | BITSTRING | 1 | DDCSPEC1 | |
| | 1... .... | | DDCFRORG | RECORG SPECIFIED FLAG |
| | .1.. .... | | DDCFLREC | LRECL SPECIFIED FLAG |
| | ..1. .... | | DDCFRFM | RECFM SPECIFIED FLAG |
| | ...1 .... | | DDCFKLEN | KEYLEN SPECIFIED FLAG |
| | .... 1... | | DDCFKOFF | KEYOFF SPECIFIED FLAG |
| | .... .1.. | | DDCFEXP | EXPIRATION ATTRIB SPEC'D FLAG |
| | .... ..1. | | DDCFRET | RETENTION ATTRIB SPEC'D FLAG |
| | .... ...1 | | DDCFPSP | PRIMARY SPACE SPECIFIED FLAG |
| 205(X'CD') | BITSTRING | 1 | DDCSPEC2 | |
| | 1... .... | | DDCFSSP | SECONDARY SPACE SPEC'D FLAG |
| | .1.. .... | | DDCFDIR | DIRECTORY BLOCKS SPEC'D FLAG |
| | ..1. .... | | DDCFAUN | ALLOCATION UNIT SPEC'D FLAG |
| | ...1 .... | | DDCFAVR | AVGREC SPECIFIED FLAG |
| | .... 1... | | DDCFVOL | VOLUME CNT SPECIFIED FLAG |
| | .... .1.. | | DDCFCIS | DATA CI SIZE SPECIFIED FLAG |
| | .... ..1. | | DDCFCIF | FREE CI % SPECIFIED FLAG |
| | .... ...1 | | DDCFCAF | FREE CA % SPECIFIED FLAG |
| 206(X'CE') | BITSTRING | 1 | DDCSPEC3 | |
| | 1... .... | | DDCFXREG | SHAREOPT XREGION SPEC'D FLAG |
| | .1.. .... | | DDCFXSYS | SHAREOPT XSYSTEM SPEC'D FLAG |
| | ..1. .... | | DDCFIMBD | VSAM IMBED SPECIFIED FLAG |
| | ...1 .... | | DDCFRPLC | VSAM REPLICATE SPECIFIED FLAG |

*Table 15. DCOLLECT Data Class Definition (Record Type 'DC') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| | .... 1... | | DDCFCOMP | COMPACTION SPECIFIED FLAG |
| | .... .1.. | | DDCFMEDI | MEDIA TYPE SPECIFIED FLAG |
| | .... ..1. | | DDCFRECT | RECORDING TECHNOLOGY FLAG |
| | .... ...1 | | DDCFVEA | VSAM EXTENDED ADDRESSING |
| 207(X'CF') | BITSTRING | 1 | DDCSPEC4 | |
| | 1... .... | | DDCSPRLF | SPACE CONSTRAINT RELIEF |
| | .1.. .... | | DDCREDUS | REDUCE SPACE BY % SPECIFIED |
| | ..1. .... | | DDCRABS | REC ACCESS BIAS SPECIFIED |
| | ...1 1111 | | * | RESERVED |
| **DATA SET ATTRIBUTES** | | | | |
| 208(X'D0') | UNSIGNED | 1 | DDCRCORG | DATA SET RECORG -- SEE CONSTANTS |
| 209(X'D1') | UNSIGNED | 1 | DDCRECFM | DATA SET RECFM -- SEE CONSTANTS |
| 210(X'D2') | BITSTRING | 1 | DDCDSFLG | |
| | 1... .... | | DDCBLK | 1 = BLOCKED, 0 = UNBLKED/NULL |
| | .1.. .... | | DDCSTSP | 1 = STANDARD OR SPANNED, ELSE 0 |
| | ..11 1111 | | * | RESERVED |
| 211(X'D3') | UNSIGNED | 1 | DDCCNTL | CARRIAGE CONTROL -- SEE CONSTS |
| 212(X'D4') | SIGNED | 4 | DDCRETPD | RETENTION PERIOD-TIME ACCESSIBLE TO SYS |
| 212(X'D4') | SIGNED | 2 | DDCEXPYR | EXPIRATION DATE - YEAR |
| 214(X'D6') | SIGNED | 2 | DDCEXPDY | EXPDT - ABSOLUTE DAY OF YEAR |
| 216(X'D8') | SIGNED | 2 | DDCVOLCT | MAXIMUM VOL COUNT FOR EXTEND |
| 218(X'DA') | UNSIGNED | 2 | DDCDSNTY | DSN TYPE -- SEE CONSTS |
| **DATA SET SPACE ATTRIBUTES** | | | | |
| 220(X'DC') | SIGNED | 4 | DDCSPPRI | PRIMARY SPACE AMOUNT |
| 224(X'E0') | SIGNED | 4 | DDCSPSEC | SECONDARY SPACE AMOUNT |
| 228(X'E4') | SIGNED | 4 | DDCDIBLK | DIRECTORY BLOCKS |
| 232(X'E8') | UNSIGNED | 1 | DDCAVREC | AVGREC -- M, K, U -- SEE CONSTS |
| 233(X'E9') | UNSIGNED | 1 | DDCREDUC | REDUCE PRIMARY OR SECONDARY SPACE BY 0-99%. DDCSPRLF AND DDCREDUS MUST BE ON. |
| 234(X'EA') | UNSIGNED | 1 | DDCRBIAS | VSAM RECORD ACCESS BIAS. REQUIRES DDCRABS, SEE CONSTANTS. |
| 235(X'EB') | CHARACTER | 1 | * | RESERVED |
| 236(X'EC') | SIGNED | 4 | DDCAUNIT | ALLOCATION UNIT AMOUNT |
| 240(X'F0') | SIGNED | 4 | DDCBSZLM | DATA SET BLOCKSIZE LIMIT |
| 244(X'F4') | SIGNED | 4 | DDCLRECL | RECORD LENGTH |
| **VSAM ATTRIBUTES** | | | | |
| 248(X'F8') | SIGNED | 4 | DDCCISZ | CISIZE FOR KS, ES OR RR |
| 252(X'FC') | CHARACTER | 4 | DDCFRSP | FREESPACE |

*Table 15. DCOLLECT Data Class Definition (Record Type 'DC') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 252(X'FC') | SIGNED | 2 | DDCCIPCT | CI FREESPACE % |
| 254(X'FE') | SIGNED | 2 | DDCCAPCT | CA FREESPACE % |
| 256(X'100') | SIGNED | 2 | DDCSHROP | VSAM SHARE OPTIONS |
| 256(X'100') | UNSIGNED | 1 | DDCXREG | VSAM XREGION SHARE OPTIONS |
| 257(X'101') | UNSIGNED | 1 | DDCXSYS | VSAM XSYSTEM SHARE OPTIONS |
| 258(X'102') | BITSTRING | 1 | DDCVINDX | VSAM SHARE OPTIONS |
| | 1... .... | | DDCIMBED | 1 = IMBED, 0 = NO |
| | .1.. .... | | DDCREPLC | 1 = REPLICATE, 0 = NO |
| | ..11 1111 | | * | RESERVED |
| 259(X'103') | UNSIGNED | 1 | DDCKLEN | VSAM KEY LENGTH |
| 260(X'104') | SIGNED | 2 | DDCKOFF | VSAM KEY OFFSET |
| 262(X'106') | UNSIGNED | 1 | DDCCAMT | VSAM CANDIDATE AMOUNT |
| 263(X'107') | CHARACTER | 1 | * | RESERVED |
| **MOUNTABLE DEVICE ATTRIBUTES** | | | | |
| 264(X'108') | UNSIGNED | 1 | DDCCOMP | COMPACTION TYPE - SEE CONSTANTS |
| 265(X'109') | UNSIGNED | 1 | DDCMEDIA | MEDIA TYPE - SEE CONSTANTS |
| 266(X'10A') | UNSIGNED | 1 | DDCRECTE | RECORDING TECHNOLOGY - SEE CONSTANTS |
| 267(X'10B') | CHARACTER | 1 | * | RESERVED |
| 268(X'10C') | CHARACTER | 4 | DDCRLS1 | RLS SUPPORT |
| 268(X'10C') | UNSIGNED | 1 | DDCBWOTP | RWO TYPE, REQUIRES DDCBWOS. SEE CONSTANTS. |
| 269(X'10D') | UNSIGNED | 4 | DDCLOGRC | SPHERE RECOVERABILITY, REQUIRES DDCLOGRS. SEE CONSTANTS. |
| 270(X'10E') | UNSIGNED | 1 | DDCSPAND | RECORD SPANS CI ABILITY, REQUIRES DDCSPANS. SEE CONSTANTS. |
| 271(X'10F') | UNSIGNED | 1 | * | RESERVED |
| 272(X'110') | CHARACTER | 28 | DDCLOGNM | LOG STREAM ID, REQUIRES DDCLSIDS. |
| 272(X'10C') | SIGNED | 2 | DDCLOGLN | ID LENGTH |
| 274(X'112') | CHARACTER | 26 | DDCLOGID | ID |
| 300(X'12C') | CHARACTER | 1 | DDCSPECX | |
| 300(X'12C') | BITSTRING | 1 | DDCSPECA | ADDITIONAL SPECIFICATION FLAGS |
| | 1... .... | | DDCBWOS | BWO SPECIFIED |
| | .1.. .... | | DDCLOGRS | SPHERE RECOVERABILITY SPECIFIED |
| | ..1. .... | | DDCSPANS | CI SPAN SPECIFIED |
| | ...1 .... | | DDCLSIDS | LOGSTREAMID SPECIFIED |
| | .... 1111 | | * | RESERVED |
| 301(X'12D') | CHARACTER | 3 | * | RESERVED |
| 304(X'130') | CHARACTER | 16 | * | RESERVED |
| 320(X'140') | CHARACTER | | DDCDATAE | END OF DDCDATA |

*Table 16. DCOLLECT Storage Class Definition (Record Type 'SC')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **STORAGE CLASS CONSTRUCT INFORMATION (RECORD TYPE 'SC')** | | | | |
| 24(X'18') | STRUCTURE | 256 | DSCDATA | STORAGE CLASS DEFINITION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 32 | DSCNMFLD | SPACE FOR NAME AND LENGTH |
| 24(X'18') | SIGNED | 2 | DSCNMLEN | LENGTH OF NAME |
| 26(X'1A') | CHARACTER | 30 | DSCNAME | NAME OF STORAGE CLASS |
| 56(X'38') | CHARACTER | 8 | DSCUSER | USERID OF LAST UPDATER |
| 64(X'40') | CHARACTER | 10 | DSCDATE | DATE OF LAST UPDATE |
| 74(X'4A') | CHARACTER | 2 | * | RESERVED |
| 76(X'4C') | CHARACTER | 8 | DSCTIME | TIME OF LAST UPDATE |
| 84(X'54') | CHARACTER | 120 | DSCDESC | DESCRIPTION |
| **STORAGE CLASS FLAGS** | | | | |
| 204(X'CC') | BITSTRING | 1 | DSCFLAGS | |
| | 1... .... | | DSCDFGSP | GUARANTEED SPACE 1=YES, 0=NO |
| | .1.. .... | | DSCDFAVL | AVAILABILITY, 1=SEE DSCAVAIL 0=DEFAULT=STANDARD |
| | ..1. .... | | DSCFDIRR | DIRECT RESPONSE TIME OBJECT, 0= DON'T CARE, 1= SEE DSCDIRR |
| | ...1 .... | | DSCFDIRB | DIRECT BIAS, 0= DON'T CARE, 1= SEE DSCDIRB |
| | .... 1... | | DSCFSEQR | SEQ RESPONSE TIME OBJECTIVE, 0= DON'T CARE, 1= SEE DSCSEQR |
| | .... .1.. | | DSCFSEQB | SEQ BIAS, 0= DON'T CARE, 1= SEE DSCSEQB |
| | .... ..1. | | DSCSYNCD | SYNCDEV, 1 = YES, 0 = NO |
| | .... ...1 | | DSCFIAD | 1 = INITIAL ACCESS RESPONSE |
| 205(X'CD') | BITSTRING | 1 | DSCFLAG2 | |
| | 1... .... | | DSCDFACC | ACCESSIBILITY, 1 =SEE SCDACCES, 0 (DEFAULT) =CONTINUOUS PREFERRED |
| | .1.. .... | | DSCDFSDR | STRIPING SUSTAINED DATA RATE 0 =NOT SPECD,1 =SEE SCDSTSDR |
| | ..1. .... | | DSCFDCFW | DIRECT CF WEIGHT SPECIFIED: 1 = YES, 0 = NO |
| | ...1 .... | | DSCFSCFW | SEQUENTIAL WEIGHT SPECIFIED: 1 = YES, 0 = NO |
| | .... 1111 | | * | RESERVED |
| 206(X'CE') | CHARACTER | 2 | * | RESERVED |
| **STORAGE CLASS ATTRIBUTES** | | | | |
| 208(X'D0') | UNSIGNED | 1 | DSCAVAIL | AVAILABILITY OPTIONS |
| 209(X'D1') | UNSIGNED | 1 | DSCDIRB | DIRECT BIAS - SEE CONSTS BELOW |
| 210(X'D2') | UNSIGNED | 1 | DSCSEQB | SEQ BIAS - SEE CONSTS BELOW |
| 211(X'D3') | UNSIGNED | 1 | DSCACCES | ACCESSIBILITY - SEE CONSTANTS |

*Table 16. DCOLLECT Storage Class Definition (Record Type 'SC') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 212(X'D4') | SIGNED | 4 | DSCIACDL | INITIAL ACCESS RESPONSE SEC |
| 216(X'D8') | SIGNED | 4 | DSCDIRR | MICROSECOND RESPONSE TIME OBJECTIVE -- DIRECT |
| 220(X'DC') | SIGNED | 4 | DSCSEQR | MICROSECOND RESPONSE TIME OBJECTIVE -- SEQUENTIAL |
| 224(X'E0') | SIGNED | 4 | DSCSTSDR | STRIPING SUSTAINED DATA RATE |
| 228(X'E4') | CHARACTER | 32 | DSCCCHST | CACHE SET NAME |
| 228(X'E4') | SIGNED | 2 | DSCCSLEN | CACHE SET NAME LENGTH |
| 230(X'E6') | CHARACTER | 30 | DSCCSNAM | CACHE SET NAME VALUE |
| 260(X'104') | SIGNED | 2 | DSCDIRCW | DIRECT CF WEIGHT |
| 262(X'106') | SIGNED | 2 | DSCSEQCW | SEQUENTIAL CF WEIGHT |
| 264(X'108') | CHARACTER | 16 | * | RESERVED |
| 280(X'118') | CHARACTER | | DSCDATAE | END OF DSCDATA |

*Table 17. DCOLLECT Management Class Definition (Record Type 'MC')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **MANAGEMENT CLASS CONSTRUCT INFORMATION (RECORD TYPE 'MC')** | | | | |
| 24(X'18') | STRUCTURE | 284 | DMCDATA | MANAGEMENT CLASS DEFINITION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 32 | DMCNMFLD | SPACE FOR NAME AND LENGTH |
| 24(X'18') | SIGNED | 2 | DMCNMLEN | LENGTH OF NAME |
| 26(X'1A') | CHARACTER | 30 | DMCNAME | NAME OF MANAGEMENT CLASS |
| 56(X'38') | CHARACTER | 8 | DMCUSER | USERID OF LAST UPDATER |
| 64(X'40') | CHARACTER | 10 | DMCDATE | DATE OF LAST UPDATE |
| 74(X'4A') | CHARACTER | 2 | * | RESERVED |
| 76(X'4C') | CHARACTER | 8 | DMCTIME | TIME OF LAST UPDATE |
| 84(X'54') | CHARACTER | 120 | DMCDESC | DESCRIPTION |
| **GENERAL SPECIFICATION FLAGS** | | | | |
| 204(X'CC') | BITSTRING | 1 | DMCSPEC1 | ATTRIBUTE SPECIFIED FLAGS, 1= SPECIFIED, 0= NOT SPEC'D |
| | 1... .... | | DMCFBVER | MCBKVS SPECIFIED FLAG |
| | .1.. .... | | DMCFBVRD | DMCBVRD SPECIFIED FLAG |
| | ..1. .... | | DMCFRBK | DMCBKDY SPECIFIED FLAG |
| | ...1 .... | | DMCFRNP | DMCBKNP SPECIFIED FLAG |
| | .... 1... | | DMCFEXDT | DMCEXDAT SPECIFIED FLAG |
| | .... .1.. | | DMCFEXDY | DMCEXPDY SPECIFIED FLAG |
| | .... ..1. | | DMCFPRDY | DMCPRDY SPECIFIED FLAG |
| | .... ...1 | | * | RESERVED |
| 205(X'CD') | BITSTRING | 1 | DMCSPEC2 | ATTRIBUTE SPECIFIED FLAGS, 1= SPECIFIED, 0= NOT SPEC'D |
| | 1... .... | | DMCFL1DY | DMCL1DY SPECIFIED FLAG |

*Table 17. DCOLLECT Management Class Definition (Record Type 'MC') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| | .1.. .... | | DMCFRLMG | DMCRLOMG SPECIFIED FLAG |
| | ..1. .... | | DMCFPELE | DMCPELEM SPECIFIED FLAG |
| | ...1 .... | | DMCFBKFQ | DMCBKFQ SPECIFIED FLAG |
| | .... 1111 | | * | RESERVED |
| **PARTIAL RELEASE CRITERIA** | | | | |
| 206(X'CE') | BITSTRING | 1 | DMCRLF | PARTIAL RELEASE FLAGS |
| | 1... .... | | DMCPREL | RELEASE 1 = YES, 0 = NO |
| | .1.. .... | | DMCPRCN | CONDITIONAL PARTITION RELEASE |
| | ..1. .... | | DMCPRIM | IMMEDIATE VALUE FOR RELEASE |
| | ...1 1111 | | * | RESERVED |
| 207(X'CF') | CHARACTER | 1 | * | RESERVED |
| **GENERATION DATA GROUP CRITERIA** | | | | |
| 208(X'D0') | BITSTRING | 1 | DMCGDGFL | GDG ATTRIBUTE FLAGS |
| | 1... .... | | DMCRLOMG | MIGRATE OR EXPIRE ROLLED OFF GDS, 1 = MIGRATE, 0 = EXPIRE |
| | .111 1111 | | * | RESERVED |
| 209(X'D1') | CHARACTER | 1 | * | RESERVED |
| 210(X'D2') | SIGNED | 2 | DMCPELEM | NUMBER OF GDG ELEMENTS ON PRIMARY |
| **DATA SET EXPIRATION CRITERIA** | | | | |
| 212(X'D4') | BITSTRING | 1 | DMCEXPF | FLAGS |
| | 1... .... | | DMCARNOL | DMCRDARC IS NOLIMIT = 1 |
| | .111 1111 | | * | RESERVED |
| 213(X'D5') | UNSIGNED | 1 | DMCEXPAC | EXPIRATION ACTION, SEE CONSTANTS |
| 214(X'D6') | SIGNED | 2 | DMCRDARC | RETAIN DAYS ARCHIVE COPY |
| **DATA SET RETENTION CRITERIA** | | | | |
| 216(X'D8') | BITSTRING | 1 | DMCRETF | DATA SET RETENTION FLAGS |
| | 1... .... | | DMCDYNOL | 1=EXPIRE AFTER DAYS= NOLIMIT ELSE 0 AND SEE DMCEXPDY |
| | .1.. .... | | DMCDTNOL | 1=EXPIRE AFTER DATE= NOLIMIT ELSE 0 AND SEE DMCEXDAT |
| | ..11 1111 | | * | RESERVED |
| 217(X'D9') | UNSIGNED | 1 | DMCRFMT | FORMAT USED FOR DMCEXDAT -- DATE OR DAYS SEE CONSTANTS |
| 218(X'DA') | CHARACTER | 2 | * | RESERVED |
| 220(X'DC') | SIGNED | 4 | DMCEXPDY | EXPIRE AFTER DAYS NO USE |
| 224(X'E0') | SIGNED | 4 | DMCEXDAT | EXPIRE DAYS SINCE CREATE OR |
| 224(X'E0') | UNSIGNED | 2 | DMCEYEAR | EXPIRE DATE SINCE CREATE |
| 226(X'E2') | UNSIGNED | 2 | DMCEDAY | SEE DMCRFMT FOR FORMAT |
| **DATA SET MIGRATION CRITERIA** | | | | |
| 228(X'E4') | BITSTRING | 1 | DMCMIGF | DATA SET MIGRATION FLAGS |

*Table 17. DCOLLECT Management Class Definition (Record Type 'MC') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| | 1... .... | | * | RESERVED |
| | .1.. .... | | * | RESERVED |
| | ..1. .... | | DMCL1NOL | MIN DAYS ON LVL 1 / LAST USE, 1=NOLIMIT, ELSE SEE DMCL1DY |
| | ...1 1111 | | * | RESERVED |
| 229(X'E5') | CHARACTER | 1 | * | RESERVED |
| 230(X'E6') | SIGNED | 2 | DMCPRDY | MIN DAYS ON PRIM / LAST USE |
| 232(X'E8') | SIGNED | 2 | DMCL1DY | MIN DAYS ON LVL 1 / LAST USE |
| 234(X'EA') | UNSIGNED | 1 | DMCCMAU | COMMAND OR AUTO MIGRATE -- SEE CONSTANTS BELOW |
| 235(X'EB') | CHARACTER | 1 | * | RESERVED |
| **DATA SET BACKUP CRITERIA** | | | | |
| 236(X'EC') | BITSTRING | 1 | DMCBKFLG | BACKUP FLAGS |
| | 1... .... | | DMCRBNOL | 1=>RETAIN DAYS ONLY BACKUP VERS = NOLIMIT 0=>SEE DMCBKNP FOR DAYS TO KEEP ONLY BACKUP |
| | .1.. .... | | DMCNPNOL | 1=>RETAIN DAYS EXTRA BACKUP VERS = NOLIMIT 0=>SEE DMCBKDY FOR DAYS TO KEEP EXTRA BACKUP |
| | ..1. .... | | * | RESERVED |
| | ...1 .... | | DMCAUTBK | 1=AUTO BACKUP ALLOWED, ELSE 0 |
| | .... 1... | | DMCCPYTF | COPY TECHNIQUE, 1=SEE DMCCPYTC 0=(DEFAULT)=STANDARD |
| | .... .111 | | * | RESERVED |
| 237(X'ED') | CHARACTER | 3 | * | RESERVED |
| 240(X'F0') | SIGNED | 2 | DMCBKFQ | BACKUP FREQUENCY |
| 242(X'F2') | SIGNED | 2 | DMCBKVS | NUMBER OF BACKUP VERSIONS |
| 244(X'F4') | SIGNED | 2 | DMCBVRD | NUM OF VERSIONS DS DELETED |
| 246(X'F6') | SIGNED | 2 | DMCBKDY | DAYS TO KEEP BACKUP VERSION |
| 248(X'F8') | SIGNED | 2 | DMCBKNP | DAYS TO KEEP ONLY BACKUP |
| 250(X'FA') | UNSIGNED | 1 | DMCBADU | ALLOW ADMIN OR USER BACKUP. SEE CONSTANTS BELOW |
| 251(X'FB') | UNSIGNED | 1 | DMCCPYTC | COPY TECHNIQUE - SEE CONSTANTS |
| 252(X'FC') | CHARACTER | 8 | DMCBKUDC | BACKUP DESTINATION CLASS |
| **MAXIMUM RETENTION CRITERIA** | | | | |
| 260(X'104') | BITSTRING | 1 | DMCMRETF | MAXIMUM RETENTION FLAGS |
| | 1... .... | | DMCRPNOL | RETPD (RETAIN PD) 1=NOLIMIT ELSE SEE DMCMRTDY |
| | .111 1111 | | * | RESERVED |
| 261(X'105') | CHARACTER | 1 | * | RESERVED |
| 262(X'106') | SIGNED | 2 | DMCMRTDY | MAXIMUM DAYS TO RETAIN |
| **CLASS TRANSITION CRITERIA** | | | | |

*Table 17. DCOLLECT Management Class Definition (Record Type 'MC') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 264(X'108') | BITSTRING | 1 | DMCTSCR | TIME SINCE CREATION FLAGS |
| | 1... .... | | DMCTCYR | YEARS SPECIFIED |
| | .1.. .... | | DMCTCMN | MONTHS SPECIFIED |
| | ..1. .... | | DMCTCDY | DAYS SPECIFIED |
| | ...1 1111 | | * | RESERVED |
| 265(X'109') | BITSTRING | 1 | DMCTSLU | TIME SINCE LAST USED FLAGS |
| | 1... .... | | DMCTSYR | YEARS SPECIFIED |
| | .1.. .... | | DMCTSMN | MONTHS SPECIFIED |
| | ..1. .... | | DMCTSDY | DAYS SPECIFIED |
| | ...1 1111 | | * | RESERVED |
| 266(X'10A') | BITSTRING | 1 | DMCPERD | PERIODIC FLAGS |
| | 1... .... | | DMCPEMN | MONTHLY SPECIFIED |
| | .1.. .... | | DMCPEQD | QUARTERLY ON DAY SPEC |
| | ..1. .... | | DMCPEQM | QUARTERLY ON MONTH SPEC |
| | ...1 .... | | DMCPEYD | YEARLY ON DAY SPEC |
| | .... 1... | | DMCPEYM | YEARLY IN MONTH SPEC |
| | .... .1.. | | DMCFIRST | FIRST DAY OF PERIOD SPEC |
| | .... ..1. | | DMCLAST | LAST DAY OF PERIOD SPEC |
| | .... ...1 | | * | RESERVED |
| 267(X'10B') | CHARACTER | 1 | * | RESERVED |
| 268(X'10C') | CHARACTER | 6 | DMCVSCR | TIME SINCE CREATION VALUES |
| 268(X'10C') | SIGNED | 2 | DMCVSCY | TIME SINCE CREATION YEARS |
| 270(X'10E') | SIGNED | 2 | DMCVSCM | TIME SINCE CREATION MONTHS |
| 272(X'110') | SIGNED | 2 | DMCVSCD | TIME SINCE CREATION DAYS |
| 274(X'112') | CHARACTER | 6 | DMCVSLU | TIME SINCE LAST USED VALUES |
| 274(X'112') | SIGNED | 2 | DMCVSUY | TIME SINCE LAST USED YEARS |
| 276(X'114') | SIGNED | 2 | DMCVSUM | TIME SINCE LAST USED MONTHS |
| 278(X'116') | SIGNED | 2 | DMCVSUD | TIME SINCE LAST USED DAYS |
| 280(X'118') | SIGNED | 2 | DMCVPRD | PERIODIC VALUES |
| 282(X'11A') | SIGNED | 2 | DMCVPMD | PERIODIC MONTHLY ON DAY |
| 284(X'11C') | CHARACTER | 4 | DMCVPQT | PERIODIC QUARTERLY VALUES |
| 284(X'11C') | SIGNED | 2 | DMCVPQD | PERIODIC QUARTERLY ON DAY |
| 286(X'11E') | SIGNED | 2 | DMCVPQM | PERIODIC QUARTERLY IN MONTH |
| 288(X'120') | CHARACTER | 4 | DMCVPYR | PERIODIC YEARLY VALUES |
| 288(X'120') | SIGNED | 2 | DMCVPYD | PERIODIC YEARLY ON DAY |
| 290(X'122') | SIGNED | 2 | DMCVPYM | PERIODIC YEARLY IN MONTH |
| 292(X'124') | CHARACTER | 16 | * | RESERVED |
| 308(X'134') | CHARACTER | | DMCDATAE | END OF DMCDATA |

## DCOLLECT Output

*Table 18. DCOLLECT Storage Group Definition (Record Type 'SG')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **STORAGE GROUP CONSTRUCT INFORMATION (RECORD TYPE 'SG')** | | | | |
| 24(X'18') | STRUCTURE | 824 | DSGDATA | STORAGE GROUP DEFINITION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 32 | DSGNMFLD | SPACE FOR NAME AND LENGTH |
| 24(X'18') | SIGNED | 2 | DSGNMLEN | LENGTH OF NAME |
| 26(X'1A') | CHARACTER | 30 | DSGNAME | NAME OF DATA CLASS |
| 56(X'38') | CHARACTER | 8 | DSGUSER | USERID OF LAST UPDATER |
| 64(X'40') | CHARACTER | 10 | DSGDATE | DATE OF LAST UPDATE |
| 74(X'4A') | CHARACTER | 2 | * | RESERVED |
| 76(X'4C') | CHARACTER | 8 | DSGTIME | TIME OF LAST UPDATE |
| 84(X'54') | CHARACTER | 120 | DSGDESC | DESCRIPTION |
| **STORAGE GROUP FLAG INFORMATION** | | | | |
| 204(X'CC') | CHARACTER | 1 | DSGFLAGS | FLAGS AND RESERVED |
| | 1... .... | | DSGFABUP | HSM AUTO BACKUP, 1=YES, 0=NO |
| | .1.. .... | | DSGFAMIG | AUTO MIGRATION, 1=YES, 0=NO |
| | ..1. .... | | DSGFADMP | AUTO DUMP, 1 = YES, 0 = NO |
| | ...1 .... | | DSGFTHRS | THRESHOLDS SPECIFIED 1 = YES, 0 = NO |
| | .... 1... | | DSGFGBKU | GUARANTEED BACKUP FREQ SPECIFIED, 1=YES, 0=NO |
| | .... .1.. | | DSGGBNOL | GUARANTEED BACKUP FREQ 1=NOLIMIT, 0=SEE DSGGBKUF |
| | .... ..1. | | DSGFIMIG | INTERVAL MIGRATION 1 = YES, 0 = NO |
| | .... ...1 | | DSGFPSM | PRIMARY SPACE MGMT 1 = YES, 0 = NO |
| 205(X'CD') | BITSTRING | 1 | DSGFLGDC | RESERVED |
| | 1... .... | | DSG32NAM | 0 = USE DSGFPRST, DSGCNFRM 1 = USE DSGSSTAT FLAG BIT ONLY; DOES NOT INDICATE NUMBER OF SYSTEMS. |
| 206(X'CE') | CHARACTER | 2 | * | RESERVED |
| **STORAGE GROUP ATTRIBUTES** | | | | |
| 208(X'D0') | UNSIGNED | 1 | DSGFTYPE | STORAGE GROUP TYPE -- SEE CONSTANTS BELOW |
| 209(X'D1') | UNSIGNED | 1 | DSGFHTHR | HIGH THRESHOLD - 0 TO 99 % |
| 210(X'D2') | UNSIGNED | 1 | DSGFLTHR | LOW THRESHOLD - 0 TO 99 % |
| 211(X'D3') | CHARACTER | 1 | * | RESERVED |
| 212(X'D4') | SIGNED | 4 | DSGFVMAX | VIO MAX DATA SET SIZE |
| 216(X'D8') | CHARACTER | 4 | DSGFVUNT | VIO UNIT TYPE |
| 220(X'DC') | CHARACTER | 8 | DSGDMPCL(5) | DUMP CLASSES FOR AUTODUMP |
| 260(X'104') | CHARACTER | 1 | DSGFPRST(8) | STATUS BY PROCESSOR |
| 260(X'104') | UNSIGNED | 1 | DSGSTAT | STATUS |
| 268(X'10C') | CHARACTER | 8 | DSGABSYS | AUTO BACKUP SYSTEM |
| 276(X'114') | CHARACTER | 8 | DSGADSYS | AUTO DUMP SYSTEM |

*Table 18. DCOLLECT Storage Group Definition (Record Type 'SG') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 284(X'11C') | CHARACTER | 8 | DSGAMSYS | AUTO MIGRATE SYSTEM |
| 292(X'124') | CHARACTER | 1 | DSGCNFRM(8) | CONFIRMED SMS STATUS FOR THIS STORAGE GROUP |
| 292(X'124') | UNSIGNED | 1 | DSGCSMSS | CONFIRMED SMS STATUS |
| 300(X'12C') | SIGNED | 4 | DSGGBKUF | GUARANTEED BACKUP FREQ |
| **STORAGE GROUP OAM ATTRIBUTES** | | | | |
| 304(X'130') | CHARACTER | 7 | DSGTBLGR | OAM TABLE SPACE ID GROUPNN |
| 311(X'137') | CHARACTER | 1 | * | RESERVED |
| 312(X'138') | BITSTRING | 1 | DSGOAMFL | OAM FLAGS |
| | 1... .... | | DSGFCYS | OAM CYCLE START/END GIVEN |
| | .1.. .... | | DSGFVLFT | VOLUME FULL THRESHOLD BIT |
| | ..1. .... | | DSGFDRST | DRIVE START THRESHOLD BIT |
| | ...1 .... | | DSGVFFER | VOL FULL @ WRITE ERROR GIVEN |
| | .... 1... | | DSGVFERR | VOL FULL @ WRITE ERROR BIT |
| | .... .111 | | * | RESERVED |
| 313(X'139') | CHARACTER | 1 | * | RESERVED |
| 314(X'13A') | UNSIGNED | 1 | DSGCYLST | OAM CYCLE START TIME (HRS) |
| 315(X'13B') | UNSIGNED | 1 | DSGCYLED | OAM CYCLE END TIME (HRS) |
| 316(X'13C') | SIGNED | 2 | DSGVOLFT | VOLUME FULL THRESHOLD BIT |
| 318(X'13E') | SIGNED | 2 | DSGDRVST | DRIVE START THRESHOLD BIT |
| 320(X'140') | CHARACTER | 32 | DSGOLIBS(8) | OPTICAL LIBRARIES |
| 320(X'140') | SIGNED | 2 | DSGOLBNL | OPTICAL LIBRARY NAME LENGTH |
| 322(X'142') | CHARACTER | 8 | DSGOLBNM | OPTICAL LIBRARY NAME |
| 330(X'14A') | CHARACTER | 22 | * | RESERVED |
| 576(X'240') | CHARACTER | 8 | DSGSSTAT(32) | STATUS BY PROCESSOR, CAN HAVE UP TO 32 SYSTEM STATUS ENTRIES. |
| 576(X'240') | UNSIGNED | 1 | DSGSYSST | REQUESTED SYSTEM STATUS |
| 577(X'241') | UNSIGNED | 1 | DSGCNSMS | CONFIRMED SMS STATUS |
| 578(X'242') | CHARACTER | 6 | * | RESERVED |
| 832(X'340') | CHARACTER | 16 | * | RESERVED |
| **IF DCUVERS IS TWO OR HIGHER** | | | | |
| 848(X'350') | UNSIGNED | 1 | DSGOFLOW | OVERFLOW |
| 849(X'351') | SIGNED | 2 | DSGEXNLN | LENGTH OF EXTEND NAME |
| 851(X'353') | CHARACTER | 30 | DSGEXNM | EXTEND STORAGE GROUP NAME |
| 881(X'371') | CHARACTER | | DSGDATAE | END OF STORAGE GROUP DATA |

*Table 19. DCOLLECT SMS Volume Information (Record Type 'VL')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **SMS VOLUME DEFINITION (RECORD TYPE 'VL')** | | | | |

*Table 19. DCOLLECT SMS Volume Information (Record Type 'VL') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 24(X'18') | STRUCTURE | 416 | DVLDATA | SMS VOLUME DEFINITION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 32 | DVLNMFLD | SPACE FOR NAME AND LENGTH |
| 24(X'18') | SIGNED | 2 | DVLNMLEN | LENGTH OF NAME -- SHOULD BE 6 |
| 26(X'1A') | CHARACTER | 6 | DVLVSER | VOLUME SERIAL NUMBER |
| 32(X'20') | CHARACTER | 24 | * | RESERVED FOR CONSISTENCY |
| 56(X'38') | CHARACTER | 8 | DVLUSER | USERID OF LAST UPDATER |
| 64(X'40') | CHARACTER | 10 | DVLDATE | DATE OF LAST UPDATE |
| 74(X'4A') | CHARACTER | 2 | * | RESERVED |
| 76(X'4C') | CHARACTER | 8 | DVLTIME | TIME OF LAST UPDATE |
| **VOLUME RECORD FLAG INFORMATION** | | | | |
| 84(X'54') | CHARACTER | 1 | DVLFLAGS | FLAGS AND RESERVED |
| | 1... .... | | DVLCONV | 1 = VOL IS IN CONVERSION |
| | .111 1111 | | * | RESERVED |
| 85(X'55') | BITSTRING | 1 | DVLFLGDC | DCOLLECT FLAGS |
| | 1... .... | | DVL32NAM | 0 = Use DVLNSTAT, DVLCSMSS, 1 = USE DVLSSTAT FLAG BIT ONLY; DOES NOT INDICATE NUMBER OF SYSTEMS. |
| | .111 1111 | | * | RESERVED |
| 86(X'56') | CHARACTER | 2 | * | RESERVED |
| **STORAGE GROUP ASSOCIATION AND STATUS INFORMATION** | | | | |
| 88(X'58') | CHARACTER | 32 | DVLSG | LENGTH AND NAME OF STORGRP |
| 88(X'58') | SIGNED | 2 | DVLSGLEN | LENGTH OF STORGRP NAME |
| 90(X'5A') | CHARACTER | 30 | DVLSTGRP | STORAGE GROUP OF THIS VOLUME |
| 120(X'78') | CHARACTER | 2 | DVLNSTAT(8) | STATUS BY SYSTEM (8 SYSTEMS) |
| 120(X'78') | UNSIGNED | 1 | DVLSMSS | SMS STATUS |
| 121(X'79') | UNSIGNED | 1 | DVLMVSS | MVS STATUS |
| 136(X'88') | UNSIGNED | 1 | DVLCSMSS(8) | CONFIRMED SMS STATUS FOR VOLUME (8 SYSTEMS) |
| **VOLUME ATTRIBUTES** | | | | |
| 144(X'90') | ADDRESS | 4 | DVLNUCBA | ADDRESS OF UCB IF KNOWN - OR 0 |
| 148(X'94') | UNSIGNED | 4 | DVLNTCPY | TOTAL CAPACITY IN MB |
| 152(X'98') | UNSIGNED | 4 | DVLNFREE | AMOUNT FREE SPACE IN MB |
| 156(X'9C') | UNSIGNED | 4 | DVLNLEXT | LARGEST FREE EXTENT IN MB |
| 160(X'A0') | SIGNED | 2 | DVLN0CNT | VOLUME LEVEL RESET COUNT |
| 162(X'A2') | UNSIGNED | 2 | DVLTRKSZ | VOLUME R1 TRACK CAPACITY |
| 164(X'A4') | SIGNED | 4 | DVLNLEVL | UPDATE LEVEL FOR VOLUME |
| 168(X'A8') | CHARACTER | 8 | DVLSSTAT(32) | STATUS BY PROCESSOR, CAN HAVE UP TO 32 SYSTEM STATUS ENTRIES. |
| 168(X'A8') | UNSIGNED | 1 | DVLSTSMS | SMS SYSTEM STATUS |
| 169(X'A9') | UNSIGNED | 1 | DVLSTMVS | MVS SYSTEM STATUS |

*Table 19. DCOLLECT SMS Volume Information (Record Type 'VL') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 170(X'AA') | UNSIGNED | 1 | DVLCNSMS | CONFIRMED SMS STATUS |
| 171(X'AB') | CHARACTER | 5 | * | RESERVED |
| 424(X'1A8') | CHARACTER | 16 | * | RESERVED |
| 440(X'1B8') | CHARACTER | | DVLDATAE | ROUND TO DWORD BOUNDARY |

*Table 20. DCOLLECT Base Configuration Information (Record Type 'BC')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **BASE CONFIGURATION INFORMATION (RECORD TYPE 'BC')** | | | | |
| 24(X'18') | STRUCTURE | 904 | DBCDATA | BASE CONFIGURATION INFORMATION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 32 | * | RESERVED |
| 56(X'38') | CHARACTER | 8 | DBCUSER | USERID OF LAST UPDATER |
| 64(X'40') | CHARACTER | 10 | DBCDATE | DATE OF LAST UPDATE |
| 74(X'4A') | CHARACTER | 2 | * | RESERVED |
| 76(X'4C') | CHARACTER | 8 | DBCTIME | TIME OF LAST UPDATE |
| 84(X'54') | CHARACTER | 120 | DBCDESC | DESCRIPTION |
| **BASE CONFIGURATION FLAGS** | | | | |
| 204(X'CC') | BITSTRING | 1 | DBCFLAGS | RESERVED |
| 205(X'CD') | BITSTRING | 1 | DBCFLGDC | DCOLLECT FLAGS |
| | 1... .... | | DBC32NAM | 0 = USE DBCFSYSN, 1 = USE DBCSYSDT FLAG BIT ONLY; DOES NOT INDICATE NUMBER OF SYSTEMS. |
| | .111 1111 | | * | RESERVED |
| 206(X'CE') | CHARACTER | 2 | * | RESERVED |
| **BASE CONFIGURATION DEFAULTS** | | | | |
| 208(X'D0') | CHARACTER | 32 | DBCDEFMC | DEFAULT MANAGEMENT CLASS |
| 208(X'D0') | SIGNED | 2 | DBCMCLEN | DEFAULT MC LENGTH OF NAME |
| 210(X'D2') | CHARACTER | 30 | DBCMCNAM | DEFAULT MANAGEMENT CLASS NAME |
| 240(X'F0') | CHARACTER | 8 | DBCDGEOM | DEFAULT DEVICE GEOMETRY |
| 240(X'F0') | SIGNED | 4 | DBCTRKSZ | TRACK SIZE IN BYTES |
| 244(X'F4') | SIGNED | 4 | DBCCYLCP | CYL CAPACITY (TRK/CYL) |
| 248(X'F8') | CHARACTER | 8 | DBCDUNIT | DEFAULT UNIT |
| **BASE CONFIGURATION INFORMATION** | | | | |
| 256(X'100') | CHARACTER | 8 | DBCSRST | SMS RESOURCE STATUS TOKEN |
| 264(X'108') | UNSIGNED | 1 | DBCSTAT | DATA SET STATUS -- SEE CONSTS |
| 265(X'109') | CHARACTER | 3 | * | RESERVED |
| 268(X'10C') | CHARACTER | 8 | DBCFSYSN(8) | SYSTEM NAMES (8 SYSTEMS) |
| 332(X'14C') | CHARACTER | 44 | DBCSCDSN | FOR ACDS ONLY, NAME OF SCDS FROM WHICH IT WAS ACTIVATED |
| **SYSTEM FEATURES** | | | | |

*Table 20. DCOLLECT Base Configuration Information (Record Type 'BC') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 376(X'178') | CHARACTER | 2 | DBCSFEAT(8) | SUPPORTED SYSTEM FEATURES (8 SYSTEMS) |
| 392(X'188') | UNSIGNED | 1 | DBCSYSNT(8) | TYPE OF SYSTEM NAMES. SEE CONSTANTS FOR TYPES. |
| 400(X'190') | CHARACTER | 16 | DBCSYSDT (32) | STATUS BY PROCESSOR, CAN HAVE UP TO 32 SYSTEM STATUS ENTRIES. |
| 400(X'190') | CHARACTER | 8 | DBCSYSNM | SYSTEM/GROUP NAME |
| 408(X'198') | CHARACTER | 2 | DBCSYSFT | SUPPORTED SYSTEM FEATURES |
| 410(X'19A') | CHARACTER | 2 | * | RESERVED |
| 412(X'19C') | UNSIGNED | 1 | DBCSNMTY | SYSTEM NAME TYPE FOR THIS ENTRY. SEE CONSTANTS. |
| 413(X'19D') | CHARACTER | 3 | * | RESERVED |
| 912(X'390') | CHARACTER | 16 | * | RESERVED |
| **IF DCUVERS IS TWO OR HIGHER,** | | | | |
| 928(X'3a0') | SIGNED | 2 | DBCSEPNL | SEPARATION NAME LENGTH |
| 930(X'3a2') | CHARACTER | 30 | DBSEPNM | SEPARATION NAME |
| 960(X'3C0') | CHARACTER | | DBCDATAE | END OF DBCDATA |

*Table 21. DCOLLECT Aggregate Group Definition (Record Type 'AG')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **AGGREGATE GROUP DEFINITION (RECORD TYPE 'AG')** | | | | |
| 24(X'18') | STRUCTURE | 616 | DAGDATA | AGGREGATE GROUP DEFINITION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 32 | DAGNMFLD | SPACE FOR NAME AND LENGTH |
| 24(X'18') | SIGNED | 2 | DAGNMLEN | LENGTH OF NAME |
| 26(X'1A') | CHARACTER | 30 | DAGNAME | NAME OF DATA CLASS |
| 56(X'38') | CHARACTER | 8 | DAGUSER | USERID OF LAST UPDATER |
| 64(X'40') | CHARACTER | 10 | DAGDATE | DATE OF LAST UPDATE |
| 74(X'4A') | CHARACTER | 2 | * | RESERVED |
| 76(X'4C') | CHARACTER | 8 | DAGTIME | TIME OF LAST UPDATE |
| 84(X'54') | CHARACTER | 120 | DAGDESC | DESCRIPTION |
| **AGGREGATE GROUP FLAG INFORMATION** | | | | |
| 204(X'CC') | BITSTRING | 1 | DAGFLAGS | |
| | 1... .... | | DAGTENQ | TOLERATE ENQ FAILURE, 1 = YES, 0 = NO |
| | .1.. .... | | DAGFRET | RETENTION PERIOD SPECIFIED, 1 = YES, 0 = NO |
| | ..1. .... | | DAGFNCPY | NUMBER OF COPIES SPECIFIED, 1 = YES, 0 = NO |
| | ...1 1111 | | * | RESERVED |
| 205(X'CD') | CHARACTER | 3 | * | RESERVED |
| **AGGREGATE GROUP ATTRIBUTES** | | | | |
| 208(X'D0') | SIGNED | 4 | DAGRETPD | RETENTION PERIOD |

*Table 21. DCOLLECT Aggregate Group Definition (Record Type 'AG') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 208(X'D0') | SIGNED | 2 | DAGEXPYR | EXPIRATION YEAR |
| 210(X'D2') | SIGNED | 2 | DAGEXPDY | ABSOLUTE DAY OF YEAR |
| 212(X'D4') | CHARACTER | 30 | DAGDEST | DESTINATION |
| 242(X'F2') | CHARACTER | 33 | DAGPREFX | OUTPUT DATA SET PREFIX |
| 275(X'113') | CHARACTER | 1 | * | RESERVED |
| 276(X'114') | CHARACTER | 52 | DAGIDSNM | INSTRUCTION DATA SET NAME |
| 276(X'114') | CHARACTER | 44 | DAGINDSN | DATA SET NAME |
| 320(X'140') | CHARACTER | 8 | DAGINMEM | MEMBER NAME, IF ANY, OR BLANK |
| 328(X'148') | CHARACTER | 52 | DAGDSNMS(5) | ARRAY OF DATA SET NAMES (5 NAMES) |
| 328(X'148') | CHARACTER | 44 | DAGDSN | DATA SET NAME |
| 372(X'174') | CHARACTER | 8 | DAGMEM | MEMBER NAME, IF ANY, OR BLANK |
| 588(X'24C') | CHARACTER | 32 | DAGMGMTC | MANAGEMENT CLASS |
| 588(X'24C') | SIGNED | 2 | DAGMCLEN | MANAGEMENT CLASS LENGTH |
| 590(X'24E') | CHARACTER | 30 | DAGMCNAM | MANAGEMENT CLASS NAME |
| 620(X'26C') | SIGNED | 4 | DAGNCOPY | NUMBER OF COPIES |
| 624(X'270') | CHARACTER | 16 | * | RESERVED |
| 640(X'280') | CHARACTER | | DAGDATAE | END OF DAGDATA |

*Table 22. DCOLLECT Optical Drive Information (Record Type 'DR')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **SMS OPTICAL DRIVE DEFINITION (RECORD TYPE 'DR')** | | | | |
| 24(X'18') | STRUCTURE | 400 | DDRDATA | SMS OPTICAL DRIVE DEFINITION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 32 | DDRNMFLD | EXTENDED FOR CONSISTENCY |
| 24(X'18') | SIGNED | 2 | DDRDVLEN | LENGTH OF NAME -- SHOULD BE 8 |
| 26(X'1A') | CHARACTER | 30 | DDRNAME | DRIVE NAME FIELD |
| 26(X'1A') | CHARACTER | 8 | DDRDNAME | DRIVE NAME |
| 34(X'22') | CHARACTER | 22 | * | RESERVED FOR CONSISTENCY |
| 56(X'38') | CHARACTER | 8 | DDRDUSER | USERID OF LAST UPDATER |
| 64(X'40') | CHARACTER | 10 | DDRDDATE | DATE OF LAST UPDATE |
| 74(X'4A') | CHARACTER | 1 | DDRFLAGS | FLAGS AND RESERVED |
| | 1... .... | | DDR32NAM | 0 = USE DDRNSTAT, 1 = USE DDRSTAT FLAG BIT ONLY; DOES NOT INDICATE NUMBER OF SYSTEMS. |
| | .111 1111 | | * | RESERVED |
| 75(X'4B') | CHARACTER | 1 | * | RESERVED |
| 76(X'4C') | CHARACTER | 8 | DDRDTIME | TIME OF LAST UPDATE |
| **LIBRARY NAME FIELDS** | | | | |
| 84(X'54') | CHARACTER | 32 | DDRLB | LENGTH AND NAME OF LIBRARY |
| 84(X'54') | SIGNED | 2 | DDRLBLEN | LENGTH OF LIBRARY NAME |
| 86(X'56') | CHARACTER | 30 | DDRLIBRY | LIBRARY FOR THIS DRIVE |

*Table 22. DCOLLECT Optical Drive Information (Record Type 'DR') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 86(X'56') | CHARACTER | 8 | DDRLBNM | LIBRARY NAME |
| 94(X'5E') | CHARACTER | 22 | * | RESERVED |
| **DRIVE STATUS BY SYSTEM** | | | | |
| 116(X'74') | CHARACTER | 4 | DDRNSTAT(8) | STATUS BY SYSTEM (8 SYSTEMS) |
| 116(X'74') | CHARACTER | 4 | DDROMST | STATUS OF EACH DRIVE |
| 116(X'74') | UNSIGNED | 1 | DDRSOUT | REQUESTED OAM STATUS |
| 117(X'75') | UNSIGNED | 1 | DDRCFCS | CURRENT OAM STATUS |
| 118(X'76') | CHARACTER | 2 | * | RESERVED |
| **MISCELLANEOUS INFORMATION** | | | | |
| 148(X'94') | UNSIGNED | 4 | DDRDCONS | CONSOLE ID |
| 152(X'98') | CHARACTER | 8 | DDRSTAT(32) | STATUS BY PROCESSOR, CAN HAVE UP TO 32 SYSTEM STATUS ENTRIES. |
| 152(X'98.') | CHARACTER | 4 | DDRSYSST | STATUS FOR THIS SYSTEM |
| 152(X'98') | UNSIGNED | 1 | DDRREQST | REQUESTED SYSTEM STATUS |
| 153(X'99') | UNSIGNED | 1 | DDRCURST | CURRENT SYSTEM STATUS |
| 154(X'9A') | CHARACTER | 2 | * | RESERVED |
| 156(X'9C') | CHARACTER | 4 | * | RESERVED |
| 408(X'198') | CHARACTER | 16 | * | RESERVED |
| 424(X'1A8') | CHARACTER | | DDRDATAE | END OF DDDRATA |

*Table 23. DCOLLECT Optical Library Information (Record Type 'LB')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **SMS OPTICAL LIBRARY DEFINITION (RECORD TYPE 'LB')** | | | | |
| 24(X'18') | STRUCTURE | 424 | DLBDATA | SMS OPTICAL LIBRARY DEFINITION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 32 | DLBNMFLD | EXTENDED FOR CONSISTENCY |
| 24(X'18') | SIGNED | 2 | DLBNMLEN | LENGTH OF LIBRARY NAME |
| 26(X'1A') | CHARACTER | 30 | DLBLNAME | LIBRARY NAME - LONG VERSION |
| 26(X'1A') | CHARACTER | 8 | DLBNAME | NAME OF OPTICAL LIBRARY |
| 34(X'22') | CHARACTER | 22 | * | RESERVED FOR CONSISTENCY |
| 56(X'38') | CHARACTER | 8 | DLBDUSER | USERID OF LAST UPDATER |
| 64(X'40') | CHARACTER | 10 | DLBDDATE | DATE OF LAST UPDATE |
| 74(X'4A') | CHARACTER | 1 | DLBFLAGS | RESERVED |
| | 1... .... | | DLB32NAM | 0 = USE DLBNSTAT, 1 = USE DLBSTAT FLAG BIT ONLY; DOES NOT INDICATE NUMBER OF SYSTEMS. |
| | .111 1111 | | * | RESERVED |
| 75(X'4B') | CHARACTER | 5 | * | RESERVED |
| 80(X'50') | CHARACTER | 8 | DLBDTIME | TIME OF LAST UPDATE |
| **OPTICAL LIBRARY STATUS BY SYSTEM** | | | | |
| 88(X'58') | CHARACTER | 4 | DLBNSTAT (X'8') | STATUS BY SYSTEM (8 SYSTEMS) |

*Table 23. DCOLLECT Optical Library Information (Record Type 'LB') (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 88(X'58') | CHARACTER | 4 | DLBOMST | STATUS FOR EACH LIBRARY |
| 88(X'58') | UNSIGNED | 1 | DLBSOUT | REQUESTED OAM STATUS |
| 89(X'59') | UNSIGNED | 1 | DLBCFCS | CURRENT OAM STATUS |
| 90(X'5A') | CHARACTER | 2 | * | RESERVED |
| **OPTICAL LIBRARY ATTRIBUTES** | | | | |
| 120(X'78') | UNSIGNED | 1 | DLBTYPE | REAL OR PSEUDO LIBRARY |
| 121(X'79') | CHARACTER | 2 | * | RESERVED |
| 123(X'7B') | UNSIGNED | 1 | DLBDTYPE | LIBRARY DEVICE TYPE |
| 124(X'7C') | UNSIGNED | 4 | DLBDCONS | LIBRARY CONSOLE ID |
| 128(X'80') | UNSIGNED | 1 | DLBEDVT | ENTRY DEFAULT USE ATTRIBUTE (TAPE ONLY) |
| 129(X'81') | UNSIGNED | 1 | DLBEJD | EJECT DEFAULT (TAPE ONLY) |
| 130(X'82') | CHARACTER | 5 | DLBLCBID | LIBRARY ID IN LIB. CONF. DB. (TAPE ONLY) |
| 135(X'87') | CHARACTER | 1 | * | RESERVED |
| 136(X'88') | CHARACTER | 8 | DLBEDUNM | ENTRY DEFAULT UNIT NAME (TAPE ONLY) |
| 144(X'90') | CHARACTER | 32 | DLBDEFDC | ENTRY DEFAULT DATA CLASS (TAPE ONLY) |
| 144(X'90') | SIGNED | 2 | DLBDCLEN | LENGTH OF ENTRY DEFAULT DATA CLASS |
| 146(X'92') | CHARACTER | 30 | DLBDCLNM | DEFAULT DATA CLASS LONG VERSION |
| 146(X'92') | CHARACTER | 8 | DLBDCNAM | NAME OF ENTRY DEFAULT DATA CLASS |
| 154(X'9A') | CHARACTER | 22 | * | RESERVED FOR CONSISTENCY |
| 176(X'B0') | CHARACTER | 8 | DLBSTAT(32) | STATUS BY PROCESSOR, CAN HAVE UP TO 32 SYSTEM STATUS ENTRIES. |
| 176(X'B0') | CHARACTER | 4 | DLBSYSST | STATUS FOR THIS SYSTEM |
| 176(X'B0') | UNSIGNED | 1 | DLBREQST | REQUESTED SYSTEM STATUS |
| 177(X'B1') | UNSIGNED | 1 | DLBCURST | CURRENT SYSTEM STATUS |
| 178(X'B2') | CHARACTER | 2 | * | RESERVED |
| 180(X'B4') | CHARACTER | 4 | * | RESERVED |
| 432(X'1B0') | CHARACTER | 16 | * | RESERVED |
| 448(X'1C0') | CHARACTER | | DLBDATAE | END OF DLBDATA |

*Table 24. DCOLLECT Cache Names (Record Type 'CN')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **SMS CACHE NAMES DEFINITION (RECORD TYPE 'CN')** | | | | |
| 24(X'18') | STRUCTURE | 152 | DCNDATA | SMS CACHE SET AND SES CACHE NAMES (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 8 | DCNCSNAM | CACHE SET NAME |
| 32(X'20') | CHARACTER | 16 | DCNSESNM (X'8') | SES CACHE NAME |
| 160(X'A0') | CHARACTER | 16 | * | RESERVED |
| 176(X'B0') | CHARACTER | | DCNDATAE | END OF DCNDATA |

## DCOLLECT Output

*Table 25. DCOLLECT Accounting Information (Record Type 'AI')*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **SMS ACCOUNTING INFORMATION DEFINITION (RECORD TYPE 'AI')** | | | | |
| 24(X'18') | STRUCTURE | 328 | DAIDATA | ACCOUNTING INFORMATION (DEFINED ON DCUDATA) |
| 24(X'18') | CHARACTER | 78 | DAIDRTN | DATA CLASS ROUTINE |
| 24(X'18') | CHARACTER | 10 | DAIDDATE | DATE LAST UPDATED |
| 34(X'22') | CHARACTER | 44 | DAIDDSNM | DATA SET NAME WHERE STORED |
| 78(X'4E') | CHARACTER | 8 | DAIDDSMR | MEMBER NAME IN DATA SET |
| 86(X'56') | CHARACTER | 8 | DAIDSRID | USERID OF LAST UPDATER |
| 94(X'5E') | CHARACTER | 8 | DAIDTIME | TIME LAST UPDATED |
| 102(X'66') | CHARACTER | 78 | DAIMRTN | MANAGEMENT CLASS ROUTINE |
| 102(X'66') | CHARACTER | 10 | DAIMDATE | DATE LAST UPDATED |
| 112(X'70') | CHARACTER | 44 | DAIMDSNM | DATA SET NAME WHERE STORED |
| 156(X'9C') | CHARACTER | 8 | DAIMDSMR | MEMBER NAME IN DATA SET |
| 164(X'A4') | CHARACTER | 8 | DAIMSRID | USERID OF LAST UPDATER |
| 172(X'AC') | CHARACTER | 8 | DAIMTIME | TIME LAST UPDATED |
| 180(X'B4') | CHARACTER | 78 | DAISRTN | STORAGE CLASS ROUTINE |
| 180(X'B4') | CHARACTER | 10 | DAISDATE | DATE LAST UPDATED |
| 190(X'BE') | CHARACTER | 44 | DAISDSNM | DATA SET NAME WHERE STORED |
| 234(X'EA') | CHARACTER | 8 | DAISDSMR | MEMBER NAME IN DATA SET |
| 242(X'F2') | CHARACTER | 8 | DAISSRID | USERID OF LAST UPDATER |
| 250(X'FA') | CHARACTER | 8 | DAISTIME | TIME LAST UPDATED |
| 258(X'102') | CHARACTER | 78 | DAIGRTN | STORAGE GROUP ROUTINE |
| 258(X'102') | CHARACTER | 10 | DAIGDATE | DATE LAST UPDATED |
| 268(X'10C') | CHARACTER | 44 | DAIGDSNM | DATA SET NAME WHERE STORED |
| 312(X'138') | CHARACTER | 8 | DAIGDSMR | MEMBER NAME IN DATA SET |
| 320(X'140') | CHARACTER | 8 | DAIGSRID | USERID OF LAST UPDATER |
| 328(X'148') | CHARACTER | 8 | DAIGTIME | TIME LAST UPDATED |
| 336(X'150') | CHARACTER | 16 | * | RESERVED |
| 352(X'160') | CHARACTER | | DAIDATAE | END OF DAIDATA |

The following constants are included in the DCOLLECT record mapping macro AMSDOUT. These constants are used to describe selected fields in the DCOLLECT records:

*Table 26. DCOLLECT Output Listing: CONSTANTS*

| Length | Type | Value | Name | Description |
|---|---|---|---|---|
| **VALUES FOR DCURCTYP—RECORD TYPE** | | | | |
| 2 | CHARACTER | D | DCUDATAT | DATA TYPE RECORD |
| 2 | CHARACTER | A | DCUASSOC | VSAM ASSOCIATION RECORD |
| 2 | CHARACTER | V | DCUVULUT | VOLUME TYPE RECORD |
| 2 | CHARACTER | DC | DCUDCDEF | DATA CLASS |

*Table 26. DCOLLECT Output Listing: CONSTANTS (continued)*

| Length | Type | Value | Name | Description |
|---|---|---|---|---|
| 2 | CHARACTER | SC | DCUSCDEF | STORAGE CLASS |
| 2 | CHARACTER | MC | DCUMCDEF | MANAGEMENT CLASS |
| 2 | CHARACTER | BC | DCUBCDEF | BASE CONFIGURATION |
| 2 | CHARACTER | SG | DCUSGDEF | STORAGE GROUP |
| 2 | CHARACTER | VL | DCUVLDEF | SMS VOLUME DEF |
| 2 | CHARACTER | AG | DCUAGDEF | AGGREGATE GROUP |
| 2 | CHARACTER | DR | DCUDRDEF | OPTICAL DRIVE |
| 2 | CHARACTER | LB | DCULBDEF | OPTICAL LIBRARY |
| 2 | CHARACTER | CN | DCUCNDEF | CACHE NAMES |
| 2 | CHARACTER | AI | DCUAIDEF | ACS INFORMATION |
| 2 | CHARACTER | M | UKTMIGR | MIGRATED DATA SET RECORD |
| 2 | CHARACTER | B | UKTBACK | BACKUP DATA SET RECORD |
| 2 | CHARACTER | C | UKCDASD | DASD CAPACITY PLANNING RECORD |
| 2 | CHARACTER | T | UKCTAPE | TAPE CAPACITY PLANNING RECORD |
| **VALUES FOR UPID AND UPVERS - PARMLIST ID AND VERSION** | | | | |
| 8 | CHARACTER | ARCUTILP | UPIDNAME | ID NAME |
| 1 | DECIMAL | 1 | UPVERNUM | CURRENT VERSION NUMBER |
| **VALUES FOR UMLEVEL—MIGRATION VOLUME LEVEL** | | | | |
| | BIT | 00 | UKLEVL0 | LEVEL 0 MIGRATION VOLUME |
| | BIT | 01 | UKLEVL1 | LEVEL 1 MIGRATION VOLUME |
| | BIT | 10 | UKLEVL2 | LEVEL 2 MIGRATION VOLUME |
| **VALUES FOR UMDEVCL—MIGRATION VOLUME DEVICE CLASS AND UBDEVCL—BACKUP VOLUME DEVICE CLASS** | | | | |
| 1 | CHARACTER | D | UKDASDV | DASD VOLUME |
| 1 | CHARACTER | T | UKTAPEV | TAPE VOLUME |
| **VALUES FOR UCLEVEL—VOLUME LEVEL** | | | | |
| | BIT | 00 | UKLEVL0 | LEVEL 0 |
| | BIT | 01 | UKLEVL1 | LEVEL 1 MIGRATION |
| **VALUES FOR UTSTYPE—TYPE OF TAPE CAPACITY PLANNING RECORD** | | | | |
| 1 | CHARACTER | B | UKBKTAPE | BACKUP TAPES |
| 1 | CHARACTER | D | UKDUTAPE | DUMP TAPES |
| 1 | CHARACTER | M | UKMGTAPE | MIGRATION TAPES |
| 2 | CHARACTER | DC | DCUDCDEF | DATA CLASS CONSTRUCT |
| 2 | CHARACTER | SC | DCUSCDEF | STORAGE CLASS CONSTRUCT |
| 2 | CHARACTER | MC | DCUMCDEF | MANAGEMENT CLASS CONSTRUCT |
| 2 | CHARACTER | BC | DCUBCDEF | BASE CONFIGURATION INFORMATION |
| 2 | CHARACTER | SG | DCUSGDEF | STORAGE GROUP CONSTRUCT |
| 2 | CHARACTER | VL | DCUVLDEF | SMS VOLUME INFORMATION |
| 2 | CHARACTER | AG | DCUAGDEF | AGGREGATE GROUP CONSTRUCT |
| 2 | CHARACTER | DR | DCUDRDEF | OPTICAL DRIVE INFORMATION |

*Table 26. DCOLLECT Output Listing: CONSTANTS (continued)*

| Length | Type | Value | Name | Description |
|---|---|---|---|---|
| 2 | CHARACTER | LB | DCULBDEF | OPTICAL LIBRARY INFORMATION |
| **VALUES FOR DCVPHYST—PHYSICAL STATUS OF VOLUME** | | | | |
| 1 | BIT | 00000011 | DCVMANGD | VOLUME IS MANAGED BY SMS |
| 1 | BIT | 00000001 | DCVINITL | IN CONVERSION TO SMS |
| 1 | BIT | 00000000 | DCVNMNGD | NON-SMS MANAGED VOLUME |
| **VALUES FOR DCDRECFM—RECORD FORMAT** | | | | |
| 1 | BIT | 10000000 | DCDRECFF | FIXED LENGTH RECORDS |
| 1 | BIT | 01000000 | DCDRECFV | VARIABLE LENGTH RECORDS |
| 1 | BIT | 11000000 | DCDRECFU | UNDEFINED LENGTH RCDS |
| **CONSTANTS FOR DDCRBIAS—RECORD ACCESS BIAS** | | | | |
| 4 | DECIMAL | 0 | DDCRABUS | USER |
| 4 | DECIMAL | 1 | DDCRABSY | SYSTEM |
| **CONSTANTS FOR DDCRCORG** | | | | |
| 4 | DECIMAL | 0 | DDCORGNL | RECORG IS NULL - SAM |
| 4 | DECIMAL | 1 | DDCORGKS | RECORG IS VSAM KSDS |
| 4 | DECIMAL | 2 | DDCORGES | RECORG IS VSAM ESDS |
| 4 | DECIMAL | 3 | DDCORGRR | RECORG IS VSAM RRDS |
| 4 | DECIMAL | 4 | DDCORGLS | RECORG IS VSAM LDS |
| **CONSTANTS FOR DDCRECFM** | | | | |
| 4 | DECIMAL | 0 | DDCFMNUL | RECFM IS NULL |
| 4 | DECIMAL | 1 | DDCFMU | RECFM IS UNDEFINED |
| 4 | DECIMAL | 2 | DDCFMV | RECFM IS VARIABLE |
| 4 | DECIMAL | 3 | DDCFMVS | RECFM IS VARIABLE SPANNED |
| 4 | DECIMAL | 4 | DDCFMVB | RECFM IS VARIABLE BLOCKED |
| 4 | DECIMAL | 5 | DDCFMVBS | RECFM IS VARIABLE BLOCKED SPANNED |
| 4 | DECIMAL | 6 | DDCFMF | RECFM IS FIXED |
| 4 | DECIMAL | 7 | DDCFMFS | RECFM IS FIXED STANDARD |
| 4 | DECIMAL | 8 | DDCFMFB | RECFM IS FIXED BLOCKED |
| 4 | DECIMAL | 9 | DDCFMFBS | RECFM IS FIXED BLOCKED SPANNED |
| **CONSTANTS FOR DDCCNTL** | | | | |
| 4 | DECIMAL | 1 | DDCCNTLA | CARRIAGE CONTROL IS ANSI |
| 4 | DECIMAL | 2 | DDCCNTLM | CARRIAGE CONTROL IS MACHINE |
| 4 | DECIMAL | 3 | DDCCNTLN | CARRIAGE CONTROL IS NULL |
| **CONSTANTS FOR DDCAVREC** | | | | |
| 1 | DECIMAL | 1 | DDCBYTES | AVGREC IS BYTES |
| 1 | DECIMAL | 2 | DDCKB | AVGREC IS KB |
| 1 | DECIMAL | 3 | DDCMB | AVGREC IS MB |
| **CONSTANTS FOR DDCDSNTY** | | | | |
| 1 | DECIMAL | 0 | DDCDSNUL | DSN TYPE IS NULL |
| 1 | DECIMAL | 1 | DDCDSPDS | DSN TYPE IS PDS |

*Table 26. DCOLLECT Output Listing: CONSTANTS (continued)*

| Length | Type | Value | Name | Description |
|---|---|---|---|---|
| 1 | DECIMAL | 2 | DDCDSLIB | DSN TYPE IS LIBRARY |
| 1 | DECIMAL | 3 | DDCDSHFS | DSN TYPE IS HFS |
| 1 | DECIMAL | 4 | DDCDSEXR | DSN TYPE IS EXTENDED(R) |
| 1 | DECIMAL | 5 | DDCDSEXC | DSN TYPE IS EXTENDED(C) |
| **CONSTANTS FOR DDCCOMP** | | | | |
| 4 | DECIMAL | 0 | DDCCNUL | NULL COMPACTION TYPE |
| 4 | DECIMAL | 1 | DDCNOCMP | NO COMPACTION |
| 4 | DECIMAL | 2 | DDCIDRC | IMPROVED DATA RECORDING CAPABILITY, COMPACTION |
| **CONSTANTS FOR DDCMEDIA** | | | | |
| 4 | DECIMAL | 0 | DDCMENUL | MEDIA TYPE IS NULL |
| 4 | DECIMAL | 1 | DDCMEDA1 | MEDIA 1 - CARTRIDGE SYSTEM |
| 4 | DECIMAL | 2 | DDCMEDA1 | MEDIA 2 - ENH CAP CART SYSTEM TAPE |
| 4 | DECIMAL | 3 | DDCMEDA3 | MEDIA 3 -HIGH PERFORMANCE |
| 4 | DECIMAL | 4 | DDCMEDA4 | MEDIA 4 -RESERVED FOR EXTENDED HIGH |
| **CONSTANTS FOR DDCRECTE** | | | | |
| 4 | DECIMAL | 0 | DDCRTNUL | DDCRECTE IS NULL |
| 4 | DECIMAL | 1 | DDC18TRK | DDCRECTE IS 18 TRACK |
| 4 | DECIMAL | 2 | DDC36TRK | DDCRECTE IS 36 TRACK |
| **CONSTANTS FOR DDCBWOTP :** | | | | |
| 4 | DECIMAL | 1 | DDCBWOC1 | BWO TYPE CICS |
| 4 | DECIMAL | 2 | DDCBWONO | BWO TYPE NONE |
| 4 | DECIMAL | 3 | DDCBWOIM | BWO TYPE IMS |
| **CONSTANTS FOR DDCLOGRC** | | | | |
| 4 | DECIMAL | 1 | DDCLOGNO | NON-RECOVERABLE SPHERE |
| 4 | DECIMAL | 2 | DDCLOGUN | UNDO - USE EXTERNAL LOG |
| 4 | DECIMAL | 3 | DDCLOGAL | ALL - (UNDO) AND FORWARD |
| **CONSTANTS FOR DDCSPAND** | | | | |
| 4 | DECIMAL | 0 | DDCSPANN | RECORD CAN NOT SPAN CI |
| 4 | DECIMAL | 1 | DDCSPANY | RECORD MAY SPAN CI |
| **CONSTANTS FOR DSCDIRB & DSCSEQB** | | | | |
| 4 | DECIMAL | 0 | DSCBIADC | BIAS = DON’T CARE |
| 4 | DECIMAL | 1 | DSCBIARD | BIAS = READ |
| 4 | DECIMAL | 2 | DSCBIAWR | BIAS = WRITE |
| **CONSTANTS FOR DSCAVAIL** | | | | |
| 4 | DECIMAL | 0 | DSCAVLDC | AVAILABILITY = DON’T CARE |
| 4 | DECIMAL | 1 | DSCAVLST | AVAILABILITY = STANDARD |
| 4 | DECIMAL | 2 | DSCAVLCN | AVAILABILITY = CONTINUOUS |
| 4 | DECIMAL | 3 | DSCAVLPR | AVAILABILITY = CONTINOUS PREFERRED |

*Table 26. DCOLLECT Output Listing: CONSTANTS (continued)*

| Length | Type | Value | Name | Description |
|---|---|---|---|---|
| **CONSTANTS FOR DSCACCES** | | | | |
| 4 | DECIMAL | 0 | DSCACCPR | ACCESSIBILITY = CONTINUOUS PREFERRED |
| 4 | DECIMAL | 1 | DSCACCRQ | ACCESSIBILITY = CONTINUOUS |
| 4 | DECIMAL | 2 | DSCACCST | ACCESSIBILITY = STANDARD |
| 4 | DECIMAL | 3 | NOPREF | ACCESSIBILITY = NO PREFERENCE |
| **CONSTANTS FOR DMCRFMT** | | | | |
| 4 | DECIMAL | 0 | DMCNULL | FIELD WAS NOT USED |
| 4 | DECIMAL | 1 | DMCFDATE | EXPIRE FORMAT DATE/CREATE |
| 4 | DECIMAL | 2 | DMCFDAYS | EXPIRE FORMAT DAYS/CREATE |
| **CONSTANTS FOR DMCCMAU** | | | | |
| 4 | DECIMAL | 0 | DMCMNONE | NO MIGRATION ALLOWED |
| 4 | DECIMAL | 1 | DMCMCMD | MIGRATE ON COMMAND ONLY |
| 4 | DECIMAL | 2 | DMCMBOTH | AUTO MIGRATE OR ON COMMAND |
| **CONSTANTS FOR DMCBADU** | | | | |
| 4 | DECIMAL | 0 | DMCBNONE | NO USER OR ADMIN BACKUP |
| 4 | DECIMAL | 1 | DMCBADM | ALLOW ADMIN COMMAND BACKUP |
| 4 | DECIMAL | 2 | DMCBBOTH | ALLOW ADMIN OR USER COMMAND |
| **CONSTANTS FOR DMCEXPAC** | | | | |
| 1 | DECIMAL | 0 | DMCEANUL | EXPIRE ACTION IS NULL |
| 1 | DECIMAL | 1 | DMCEADEL | EXPIRE ACTION IS DELETE |
| 1 | DECIMAL | 2 | DMCEAARC | EXPIRE ACTION IS ARCHIVE |
| **CONSTANTS FOR DMCRLF** | | | | |
| 0 | BIT | 10000000 | DMCRLFYE | PARTIAL RELEASE = YES, IMMEDIATE RELEASE = NO |
| 0 | BIT | 01000000 | DMCRLFCN | CONDITION PARTIAL RELEASE = YES, IMMEDIATE RELEASE = NO |
| 0 | BIT | 00000000 | DMCRLFNO | PARTIAL RELEASE = NO, IMMEDIATE RELEASE = NO |
| 0 | BIT | 10100000 | DMCRLFYI | PARTIAL RELEASE = YES, IMMEDIATE RELEASE = YES |
| 0 | BIT | 01100000 | DMCRLFCI | CONDITIONAL PARTIAL RELEASE = YES, IMMEDIATE CONDITIONAL RELEASE = YES |
| **CONSTANTS FOR DMCCPYTC** | | | | |
| 1 | DECIMAL | 0 | DMCCPYST | STANDARD |
| 1 | DECIMAL | 1 | DMCCPYPR | CONCURRENT PREFERRED |
| 1 | DECIMAL | 2 | DMCCPYRQ | CONCURRENT REQUIRED |
| **CONSTANTS FOR DSGFTYPE** | | | | |
| 4 | DECIMAL | 0 | DSGPOOL | STORAGE GROUP TYPE IS POOL |
| 4 | DECIMAL | 1 | DSGVIO | STORAGE GROUP TYPE IS VIO |
| 4 | DECIMAL | 2 | DSGDUMMY | STORAGE GROUP TYPE IS DUMMY |
| 4 | DECIMAL | 3 | DSGOBJ | STORAGE GROUP TYPE IS OBJECT |

*Table 26. DCOLLECT Output Listing: CONSTANTS (continued)*

| Length | Type | Value | Name | Description |
|---|---|---|---|---|
| 4 | DECIMAL | 4 | DSGOBJBK | STORAGE GROUP TYPE IS OBJECT BACKUP |
| 4 | DECIMAL | 5 | DSGTAPE | STORAGE GROUP TYPE IS TAPE |
| **CONSTANTS FOR DSGSTAT AND DSGSYSST** | | | | |
| 1 | DECIMAL | 0 | DSG0 | NO STATUS SPECIFIED |
| 1 | DECIMAL | 1 | DSGENBL | STORAGE GROUP IS ENABLED |
| 1 | DECIMAL | 2 | DSGQUI | STORAGE GROUP IS QUIESCED/ALL |
| 1 | DECIMAL | 3 | DSGQUIN | STORAGE GROUP IS QUIESCED/NEW |
| 1 | DECIMAL | 4 | DSGDIS | STORAGE GROUP IS DISABLED/ALL |
| 1 | DECIMAL | 5 | DSGDISN | STORAGE GROUP IS DISABLED/NEW |
| **SMS STATUS - DVLSMSS AND DVLSTSMS** | | | | |
| 1 | DECIMAL | 0 | DVL0 | NO STATUS GIVEN |
| 1 | DECIMAL | 1 | DVLENBL | SMS STATUS IS ENABLED |
| 1 | DECIMAL | 2 | DVLQUI | SMS STATUS IS QUIESCED/ALL |
| 1 | DECIMAL | 3 | DVLQUIN | SMS STATUS IS QUIESCED/NEW |
| 1 | DECIMAL | 4 | DVLDIS | SMS STATUS IS DISABLED/ALL |
| 1 | DECIMAL | 5 | DVLDISN | SMS STATUS IS DISABLED/NEW |
| **MVS STATUS - DVLMVSS AND DVLSTMVS** | | | | |
| 1 | DECIMAL | 1 | DVLONLN | MVS STATUS IS ONLINE |
| 1 | DECIMAL | 2 | DVLOFFLN | MVS STATUS IS OFFLINE |
| 1 | DECIMAL | 3 | DVLPOFF | MVS STATUS IS PENDING OFFLINE |
| 1 | DECIMAL | 4 | DVLBOXED | MVS STATUS IS BOXED |
| 1 | DECIMAL | 5 | DVLNRDY | MVS STATUS IS NOT READY |
| **CONSTANTS FOR DBCSTAT** | | | | |
| 4 | DECIMAL | 1 | DBCVALID | DATA SET IS VALID |
| 4 | DECIMAL | 2 | DBCINVAL | DATA SET IS NOT VALID |
| 4 | DECIMAL | 3 | DBCUNKWN | DATA SET STATUS IS UNKNOWN |
| **CONSTANTS FOR DBCSYSNT AND DBCSNMTY** | | | | |
| 4 | DECIMAL | 0 | DBCSYSNS | NAME TYPE NOT SPECIFIED |
| 4 | DECIMAL | 1 | DBCSYSTM | NAME TYPE IS SYSTEM NAME |
| 4 | DECIMAL | 2 | DBCSYSPL | NAME TYPE IS SYSTEM GROUP NAME |
| **CONSTANTS FOR DBCSYSFT** | | | | |
| 2 | HEX | X'80' | DBCASMS | ACTIVE SMS |
| 2 | HEX | X'40' | DBCPDSE | PDSE FEATURE |
| 2 | HEX | X'20' | DBCCDMP | SAM COMPRESSION |
| 2 | HEX | X'10' | DBCSESC | SES CACHE FEATURE |
| **MVS STATUS - DDRSOUT, DDRCFCS, DDRREQST, AND DDRCURST** | | | | |
| 1 | DECIMAL | 0 | DDRNOCON | OAM STATUS IS NO CONNECTIVITY |
| 1 | DECIMAL | 1 | DDRONLN | OAM STATUS IS ONLINE |
| 1 | DECIMAL | 2 | DDROFFLN | OAM STATUS IS OFFLINE |

*Table 26. DCOLLECT Output Listing: CONSTANTS (continued)*

| Length | Type | Value | Name | Description |
|---|---|---|---|---|
| 1 | DECIMAL | 3 | DDRNORST | NO OUTSTANDING REQUEST |
| **MVS STATUS - DLBSOUT, DLBCFCS, DLBREQST, AND DLBCURST** | | | | |
| 1 | DECIMAL | 0 | DLBNOCON | OAM STATUS IS NO CONNECTIVITY |
| 1 | DECIMAL | 1 | DLBONLN | OAM STATUS IS ONLINE |
| 1 | DECIMAL | 2 | DLBOFFLN | OAM STATUS IS OFFLINE |
| 1 | DECIMAL | 3 | DLBNORST | NO OUTSTANDING REQUEST (DLBSOUT ONLY) |
| 1 | DECIMAL | 4 | DLBLPENO | LIBRARY PENDING OFFLINE |
| **TYPE OF LIBRARY - DLBTYPE** | | | | |
| 1 | DECIMAL | 0 | DLBNOOPT | NOT OPTICAL LIBRARY |
| 1 | DECIMAL | 1 | DLBREAL | REAL LIBRARY |
| 1 | DECIMAL | 2 | DLBPSEUD | PSEUDO LIBRARY |
| **TYPE OF LIBRARY DEVICE - DLBDTYPE** | | | | |
| 1 | DECIMAL | 0 | DLBD9246 | IBM 9246 LIBRARY |
| 1 | DECIMAL | 1 | DLBD3995 | IBM 3995 LIBRARY |
| 1 | DECIMAL | 2 | DLBTAPE | TAPE LIBRARY |
| **ENTRY DEFAULT USE ATTRIBUTE - DLBEDVT(TAPE LIBRARY ONLY)** | | | | |
| 1 | DECIMAL | 1 | DLBPRVT | PRIVATE VOLUME |
| 1 | DECIMAL | 2 | DLBSCRT | SCRATCH VOLUME |
| **EJECT DEFAULT - DLBEJD** | | | | |
| 1 | DECIMAL | 1 | DLBPURGE | PURGE TCDB VOLUME RECORD |
| 1 | DECIMAL | 2 | DLBKEEP | KEEP TCDB VOLUME RECORD |

# DCOLLECT Output Record Field Descriptions

## Header Record Field

This is the header for all the record types. It contains all the common fields that are needed regardless of the type of data collected. All other output record data is appended to the header.

| Name | Description |
|---|---|
| **DCURDW** | This field is NOT the RDW for the record that an assembler program sees. Assembler programs see the true RDW (4 bytes) preceding DCURDW. High-level languages, such as PL/1, have the true RDW stripped and see DCURDW as the first field in the record. |
| **DCULENG** | Length of this record in bytes. |
| **DCURCTYP** | Record type for this record. Types are:<br>**D** Active Data Set Record<br>**A** VSAM Association Information<br>**V** Volume Information<br>**M** Migrated Data Set Information<br>**B** Backup Data Set Information<br>**C** DASD Capacity Planning Information |

**T** Tape Capacity Planning Information.

**DCUVERS** The version number for this record.

**DCUSYSID** Identification field of the system running DCOLLECT. This is the same as the SMF system ID for the system.

**DCUTMSTP** The timestamp of the DCOLLECT run. This timestamp is the same for all records collected during one invocation of DCOLLECT. The timestamp consists of:

**DCUTIME** The time in hundredths of a second from midnight (same format as in the SMF record header time stamp).

**DCUDATE** The date in CCYYDDDF format (packed decimal).

## Active Data Set Record Field

This is the section for data set information. These records are collected when the VOLUME, or STORAGEGROUP, or both of the parameters are selected, and the NODATAINFO parameter is *not* specified. One of these records is created for each data set encountered on every volume scanned. The record type for this record is D.

**Name** **Description**

**DCDDSNAM** 1-to-44 character data set name. For a VSAM data set, this is the component true name (that is, the data or index name).

**DCDERROR** This is the error indication byte. Each bit in this byte represents a distinct error encountered during processing. The following are indicated if the specified bit is 1.

**Note:** If DCOLLECT indicates errors in the VTOC/VVDS, use the DIAGNOSE command to get further information on the errors.

**DCDEMNGD** An inconsistency was found in the SMS indicators for this data set.

**DCDNOSPC** No space information was generated for this data set. The affected fields are:
- DCDALLSP
- DCDUSESP
- DCDSCALL
- DCDNMBLK

**DCDVSAMI** An inconsistency was found in the VSAM indicators for this data set.

**DCDNOFM1** A VTOC entry does not exist for this data set.

**DCDFLAG1** This is the first byte of flags. The following are indicated if the specified bit is 1.

**DCDRACFD** The data set is RACF defined.

**DCDSMSM** The data set resides on an SMS-managed volume.

**DCDTEMP** The data set is a temporary data set. This indicator is valid only for SMS-managed data sets.

**DCDPDSE** The data set is a partitioned extended data set.

**DCDGDS** The data set is a generation data group data set. This indicator is valid only for SMS-managed data sets.

**DCDREBLK** The data set can be reblocked.

**DCDCHIND** The data set has been opened for other than input since the last time a backup copy was made.

**DCDCKDSI** The data set is a Checkpoint/Restart Checkpoint data set.

**DCDFLAG2** This is the second byte of flags. The following are indicated if the specified bit is 1.

**DCDNOVVR** There is no VVDS entry (VVR) for this data set.

**DCDINTCG** This data set is an integrated catalog facility catalog.

**DCDINICF** This data set is cataloged in an integrated catalog facility catalog. This indicator is valid only for VSAM data sets.

**DCDALLFG** Allocated space information was returned from DASDCALC in field DCDALLSP.

**DCDUSEFG** Used space information was returned from DASDCALC in field DCDUSESP.

**DCDDECFG** Secondary space information was returned from DASDCALC in field DCDSCALL.

**DCDNMBFG** Space wasted information was returned from DASDCALC in field DCDNMBLK.

**DCDFLAG3** This is the third byte of flags. The following are indicated if the specified bit is 1.

**DCDPDSEX** Data set is an z/OS UNIX System Services MVS data set.

**DCDSTRP** Data set is an extended format data set.

**DCDDDMEX** Indicates that Distributed Data Management (DDM) information exists in the catalog for this data set.

**DCDDSORG** This field describes the data set organization. Only one bit will be set to 1.

**DCDDSGIS** Data set organization is indexed sequential.
**DCDDSGPS** Data set organization is physical sequential.
**DCDDSGDA** Data set organization is direct.
**DCDDSGPO** Data set organization is partitioned.
**DCDDSGU** Data set organization is immovable.
**DCDDSGGS** Data set organization is graphics.
**DCDDSGVS** Data set organization is VSAM.

**DCDRECRD** The record format byte for the data set.

**DCDRECFM** These two bits describe the record format for the data set. This field is mapped by constants DCDRECFV, DCDRECFF, and DCDRECFU.

**X'00'** Unused (undetermined)
**X'01'** Variable length records
**X'10'** Fixed length records

**X'11'** Undefined length records

**DCDRECFT** Blocks in the data set can use the hardware track overflow feature.

**DCDRECFB** The data set records are blocked. This bit should not be set if the record format is “Undefined”.

**DCDRECFS** If the record format is “Fixed”, the data set is using standard blocks, that is, there are no truncated blocks or unfilled tracks. If the record format is “Variable”, the records are spanned.

**DCDRECFA** The data set uses ANSI control characters.

**DCDRECFC** The data set uses machine control characters.

**DCDNMEXT** The number of extents the data set is using on this volume.

**DCDVOLSR** 6-character volume serial number the data set resides on.

**DCDBKLNG** The length of each block for this data set.

**DCDRCLNG** The length of each record for this data set.

**DCDALLSP** The number of tracks allocated, multiplied by the track capacity. The actual number of bytes available to the data set might be less because of non-optimal block sizes. The amount of space allocated to the data set is rounded to the nearest kilobyte. See DCDNMBLK below.

**DCDUSESP** The number of tracks containing data from the data set, multiplied by the optimal track capacity (space used). It does not indicate the number of bytes in a data set.

**Note:** This information cannot be provided for VSAM or ISAM data sets. For VSAM and ISAM data sets, this field is set to zeros.

**DCDSCALL** The amount of space for the secondary allocation of this data set in kilobytes (rounded to the nearest kilobyte).

**DCDNMBLK** The amount of space in kilobytes (rounded to the nearest kilobyte) that are wasted from non-optimal block size and from unused tracks on allocated cylinders. (This information cannot be provided for VSAM or ISAM data sets.)

**DCDCREDT** The creation date of the data set in packed decimal. The format is X'YYYYDDDF'.

**DCDEXPDT** The expiration date of the data set in packed decimal. The format is X'YYYYDDDF'. For VSAM data sets, this will always be a ’never expire’ date.

**DCDLSTRF** The last referenced date of the data set in packed decimal. The format is X'YYYYDDDF'.

**DCDDSSER** The data set serial number. The data set serial number identifies the first or only volume containing the data set.

**DCDVOLSQ** The volume sequence number.

**DCDLBKDT** The system timestamp when the data set was backed up last. This field is an 8-byte binary value; it is only valid for SMS data sets.

The format is STCK. See Enterprise Systems Architecture/390® (ESA/390) principles of operations under time of day clock.

**DCDDCLAS** Data class name[6]

**DCDDCLNG** The actual length of the data class name in DCDDATCL.

**DCDDATCL** The data class name field.

**DCDSCLAS** Storage class name field[6]

**DCDSCLNG** The actual length of the storage class name in DCDSTGCL.

**DCDSTGCL** The storage class name.

**DCDMCLAS** management class name field[6]

**DCDMCLNG** The actual length of the management class name in DCDMGTCL.

**DCDMGTCL** The Management class name.

**DCDSTOGP** Storage group name field[6]

**DCDSGLNG** The actual length of the storage group name in DCDSTGRP.

**DCDSTGRP** The storage group name.

**DCDCCSID** The coded character set identifier. This field is used to identify the coded character set to be used with this data set.

**DCDUDSIZ** Data set size before compression. This field is applicable to extended format data sets. If this value and DCDCUDSZ are both zero, it is an indication that the data set is not compressed.

**DCDCUDSZ** Data set size after compression. This field is applicable to extended format data sets only, and refers only to user data within the data set. That is, any system data written to the data set is not included here. If this value and DCDCUDSZ are both zero, it is an indication that the data set is not compressed.

**DCDBDSZ** Data set sizes that are not valid in either or both of DCDUDSIZ or DCDCUDSZ. These fields might contain non-zero values, but should not be used. This field is applicable to non-VSAM extended format data sets only.

**DCDOVERA** VSAM space available for release. This is the difference between the High Used Relative Byte Address (HURBA) and the High Allocated Relative Byte Address. This field is computed for all VSAM data sets but only applies to data sets eligible for partial release (VSAM Extended Format data sets). The space value is represented in bytes.

6. Only available for the D record of the dataset residing on the first volume of a multi-volume dataset.

## VSAM Association Record Field

This is the section for VSAM data set association information. This record ties data and index components to the sphere name and provides other VSAM-related information. The record type for this record is A.

| Name | Description |
|---|---|
| **DCADSNAM** | 1-to-44 character *component* name of the data or index component of the data set. |
| **DCAASSOC** | 1- to 44-character *sphere* name of the data set. This is also referred to as the *cluster* name. |
| **DCAFLAG1** | This is the first byte of information flags for VSAM data sets. The following are indicated if the specified bit is 1. |
| | **DCAKSDS** The data set is a VSAM key-sequenced data set. |
| | **DCAESDS** The data set is a VSAM entry-sequenced data set. |
| | **DCARRDS** The data set is a VSAM relative record data set. |
| | **DCALDS** The data set is a VSAM linear data set. |
| | **DCAKRDS** The data set is a key range data set. |
| | **DCAAIX** The data set is a alternate index data set. |
| | **DCADATA** This component is a data component. |
| | **DCAINDEX** This component is an index component. |
| **DCAFLAG2** | This is the second byte of information flags for VSAM data sets. The following are indicated if the specified bit is 1. |
| | **DCAKR1ST** This is the first segment of a key range data set. |
| | **DCAIXUPG** The data set is an AIX that is upgraded when the base cluster changes. |
| | **DCAVRRDS** The data set is a VSAM variable length relative record data set. |
| | **DCANSTAT** This record does not contain VSAM statistics. VSAM statistics are valid only for the first extent on the first volume for the data set. If this bit is set, the following fields will contain zeros:<br>• DCAHURBA<br>• DCAHARBA<br>• DCANLR<br>• DCADLR<br>• DCAINR<br>• DCAUPR<br>• DCARTR<br>• DCAASP<br>• DCACIS<br>• DCACAS<br>• DCAEXC<br>• DCARKP<br>• DCAKLN<br>• DCAHURBC<br>• DCAHARBC |
| | **DCASRCI** Relative CI. If this bit is set, fields DCAHURBC and |

DCAHARBC should be used rather than DCAHARBA and DCAHURBA.

**DCAG4G** If this bit is set, the data set can be extended addressability (greater than 4 gigabytes).

**DCAHURBA** The high-used relative byte address for the data set. This number represents the ″high water mark″ for the data set, and under normal circumstances, represents the current amount of space used by the data set.[7]

**DCAHARBA** The high-allocated relative byte address for the data set. This number represents the total amount of space that has been allocated to the data set across all extents.[7]

**DCANLR** The number of logical records that exist in the data set.[7]

**DCADLR** The number of logical records that have been deleted from the data set.[7]

**DCAINR** The number of logical records that have been inserted into the data set.[7]

**DCAUPR** he number of logical records that have been updated in the data set.[7]

**DCARTR** The number of logical records that have been retrieved from the data set.[7]

**DCAASP** The amount of space, in bytes, that is available (freespace) for the data set.[7]

**DCACIS** The number of control interval splits that have occurred for the data set.[7]

**DCACAS** The number of control area splits that have occurred for the data set.[7]

**DCAEXC** The number of execute channel program instructions that have been issued for the data set.[7]

**DCARKP** The offset of the key for the data set.[7]

**DCAKLN** The length of the key for the data set.[7]

**DCAHURBC** The high-used relative byte address for the data set. This number represents the high-water mark for the data set, and under normal circumstances, represents the current amount of space used by the data set. This value is calculated from CI size times the number of CIs if the DCASRCI bit is on.[7]

**DCAHARBC** The high-allocated relative byte address for the data set. This number represents the total amount of space that has been allocated to the data set across all extents. This value is calculated from CI size times the number of CIs if the DCASRCI bit is on.[7]

---

7. The value for this record is not valid in the following cases:
   - All integrated catalog facility catalogs and VVDSs allocated to the catalog address space
   - All record type “A” records generated for secondary extents of VSAM data sets.

## Volume Record Field

This is the section for volume information. One of these records is created for each volume scanned. These records are collected when the VOLUME, or STORAGEGROUP, or both parameters are selected, and the NOVOLUMEINFO parameter is *not* specified. The record type for this record is V.

| Name | Description |
|---|---|
| **DCVVOLSR** | 6-character volume serial number. |
| **DCVFLAG1** | This is the first byte of flags. The following are indicated if the specified bit is 1. |
| | **DCVINXEX** An index exists for the VTOC. |
| | **DCVINXEN** The index for the VTOC is active. |
| | **DCVUSPVT** The volume usage is private. |
| | **DCVUSPUB** The volume usage is public. |
| | **DCVUSSTO** The volume usage is storage. |
| | **DCVSHRDS** The device can be shared between multiple processors. |
| | **DCVPHYST** 2 bits to indicate the physical status of the volume. This field is mapped by constants DCVNMNGD, DCVINITL, and DCVMANGD.<br>**BB'00'** Non-SMS-managed volume<br>**BB'01'** In conversion to SMS<br>**BB'11'** Volume is SMS-managed |
| **DCVERROR** | This is the error indication byte. Each bit in this byte represents a distinct error encountered during processing. The following descriptions are indicated if the specified bit is 1. |
| | **DCVEVLCP** An error occurred during calculation of the volume capacity value. |
| | **DCVEBYTK** An error occurred during calculation of the value for bytes per track. This will impact the following values:<br>• DCVPERCT<br>• DCVFRESP<br>• DCVALLOC<br>• DCVLGEXT<br>• DCVVLCAP |
| | **DCVELSPC** An error occurred acquiring information from the format-4 DSCB. This will impact the following values:<br>• DCVINXEN<br>• DCVFRAGI<br>• DCVFREXT<br>• DCVFDSCB<br>• DCVFVIRS<br>• DCVPHYST<br>• Plus those indicated for DCVEBYTK |

**DCVPERCT** The percentage of free space remaining on the volume. This is the ratio of the amount of free space to volume capacity.

**DCVFRESP** The amount of free space remaining on the volume expressed in kilobytes. This is a summation of the total free cylinders and total additional free tracks on the volume.

**DCVALLOC** The amount of allocated space on the volume expressed in kilobytes.

**DCVVLCAP** The total capacity of this volume expressed in kilobytes.

**DCVFRAGI** The fragmentation index, which is a numeric representation of the relative size and distribution of free space on the volume. A large index value indicates a high degree of fragmentation.

**DCVLGEXT** The largest free extent on the volume, expressed in kilobytes.

**DCVFREXT** The number of free extents on the volume.

**DCVFDSCB** The number of free DSCB in the VTOC. This field might not be accurate for a large VTOC.

**DCVFVIRS** The number of available VTOC index records (VIRs).

**DCVDVTYP** The device type of the volume, for example 3390.

**DCVDVNUM** The device number (address) of the volume, for example 0A20 or 1D01.

**DCVSTGGP** Storage group name

- **DCVSGLNG** The actual length of the storage group name in DCVSGTCL.
- **DCVSGTCL** The storage group name.

## Data Class Construct Field

This is the section for Data Class Construct information. These records are collected when SMSDATA is selected, and Data Class constructs are defined to the control data set selected. The record type for this record is ’DC’.

| Name | Description |
|---|---|
| **DDCNMFLD** | The Data Class Construct name.<br>**DDCNMLEN** The length of this construct name.<br>**DDCNAME** The name of this construct. |
| **DDCUSER** | The USERID of the last person to make a change to this construct. |
| **DDCDATE** | The date that this construct was last changed. The format is ″YYYY/MM/DD″ in EBCDIC. |
| **DDCTIME** | The time that this construct was last changed. The format is ″HH:MM″ in EBCDIC. |
| **DDCDESC** | The description of this construct. |
| **DDCSPEC** | Data Class Parameters Specification Flags. The following are indicated if the specified bit is ’1’. |

- **DDCSPEC1** First byte of flags.
  - **DDCFRORG** Record organization specified.
  - **DDCFLREC** LRECL specified.

**DDCFRFM** RECFM specified.

**DDCFKLEN** KEYLEN specified.

**DDCFKOFF** KEYOFF specified.

**DDCFEXP** Expiration date attribute specified.

**DDCFRET** Retention period attribute specified.

**DDCFPSP** Primary space allocation specified.

**DDCSPEC2** Second byte of flags.

**DDCFSSP** Secondary space allocation specified.

**DDCFDIR** Directory blocks specified.

**DDCFAUN** Allocation unit specified.

**DDCFAVR** AVGREC specified.

**DDCFVOL** Volume count specified.

**DDCFCIS** VSAM CISIZE specified.

**DDCFCIF** Free CI % specified.

**DDCFCAF** Free CA % specified.

**DDCSPEC3** Third byte of flags.

**DDCFXREG** SHAREOPT XREGION specified.
**DDCFXSYS** SHAREOPT XSYSTEM specified.
**DDCFIMBD** VSAM IMBED specified.
**DDCFRPLC** VSAM REPLICATE specified.
**DDCFCOMP** Compaction specified.
**DDCFMEDI** Media Type specified.
**DDCFRECT** Recording Technology specified.

**DDCRCORG** This field describes how VSAM data sets allocated by this data class are organized and is mapped by the constants DDCORGKS, DDCORGES, DDCORGRR, DDCORGLS, and DDCORGNL.

**1** Record organization is VSAM Keyed Sequential Data Set.

**2** Record organization is VSAM Entry-Sequenced Data Set.

**3** Record organization is VSAM Relative Record Data Set.

**4** Record organization is VSAM Linear Space Data Set.

**0** Record organization is null - this data class is used for non-VSAM data sets having Partitioned Organization (PO) or Physical Sequential (PS) organization.

**DDCRECFM** This field describes the data set record format assigned to non-VSAM data sets and is mapped by constants DDCFMNUL, DDCFMU, DDCFMV, DDCFMVS, DDCFMVB, DDCFMVBS, DDCFMF, DDCFMFS, DDCFMFB, and DDCFMFBS.

**0** Record format is null.
**1** Record format is undefined format.

**2** Record format is variable.
**3** Record format is variable spanned.
**4** Record format is variable blocked.
**5** Record format is variable blocked spanned.
**6** Record format is fixed.
**7** Record format is fixed standard.
**8** Record format is fixed blocked.
**9** Record format is fixed blocked standard.

**DDCDSFLG** Data set flags for non-VSAM data sets.
**DDCBLK** 1=Blocked, 0=Unblocked/Null.
**DDCSTSP** 1=Standard or Spanned.

**DDCCNTL** This field describes the type of carriage control assigned to non-VSAM data sets and is mapped by DDCCNTLA, DDCCNTLM, and DDCCNTLN.
**1** Carriage control is ANSI carriage control.
**2** Carriage control is MACHINE carriage control.
**3** Carriage control is NULL.

**DDCRETPD** If DDCFRET is '1', this field is the retention period in days assigned to data sets by this Data Class. If DDCFEXP is '1' then this field should be interpreted by the two fields, DDCEXPYR and DDCEXPDY. Data sets are deleted or archived one day after the retention period or on the expiration date occurs.

**DDCEXPYR** Expiration Date - year assigned to data sets by this Data Class.

**DDCEXPDY** Expiration Date - absolute day of year assigned to data sets by this Data Class.

**DDCVOLCT** The maximum number of volumes that can be used to store your data set. Possible values range from 1 to 59.

**DDCDSNTY** This field indicates the format used to allocate data sets using this Data Class mapped by DDCDSNUL, DDCDSPDS, and DDCDSLIB.
**0** Field value is null.
**1** The system allocates the data sets as PDSs.
**2** The system allocates the data sets as PDSEs.

**DDCSPPRI** The value in this field is the primary space, and when multiplied by DDCAUNIT, determines the amount of space that this Data Class initially allocates for a data set.

**DDCSPSEC** The value in this field is the secondary space, and when multiplied by DDCAUNIT, determines the additional space that can be allocated by this Data Class for a data set.

**DDCDIBLK** The value in this field shows the number of blocks allocated for the directory of a partitioned data set.

**DDCAVREC** This field shows whether this Data Class allocates space in bytes, kilobytes, or megabytes and is mapped by DDCBYTES, DDCKB, and DDCMB.
**1** Space is allocated in bytes - U.
**2** Space is allocated in kilobytes - K.
**3** Space is allocated in megabytes - M.

**DDCAUNIT** This field shows the multiplication factor used to determine primary and secondary allocated space. Possible values range from 0 to 65,535.

**DDCBSZLM** This field shows the blocksize limit value specified in the BLKSZLIM parameter of the associated Data Class.

**DDCLRECL** This field shows, in bytes, the logical record length used when allocating data sets in this Data Class. The value is the length of fixed-length records or the maximum length of variable-length records.

**DDCCISZ** This field shows the number of bytes allocated by the Data Class for each control interval in the data portion, not the index portion, of a VSAM data set. This field only applies to ESDS, KSDS, or RRDS VSAM data sets.

**DDCFRSP** VSAM Control Interval and Control Area FREESPACE fields used by the Data Class. Possible values for either field range from 1 to 100.

- **DDCCIPCT** This field shows what percentage of each control interval in a key-sequenced VSAM data set should be set aside as free space.
- **DDCCAPCT** This field shows what percentage of each control area in a key-sequenced VSAM data set should be set aside as free space.

**DDCSHROP** These fields indicate VSAM Share Options assigned by the Data Class to VSAM data sets.

- **DDCXREG** This field shows how a VSAM data set can be shared among regions of one system, or across regions of multiple systems. Possible values are 1, 2, 3, and 4, if specified for the Data Class.
- **DDCXSYS** This field shows how a VSAM data set can be shared among systems. Possible values are 3 and 4, if specified for the Data Class.

**DDCVINDX** These fields indicate VSAM Options assigned by the Data Class to VSAM data sets.

- **DDCIMBED** This field indicates whether or not each sequence-set record is to be written as many times as possible on the first track of the data control area for key-sequenced VSAM data sets only. If specified, the following definitions apply.
  - **1** IMBED - Write each sequence-set record, as many times as possible, on the first track of the data control area
  - **0** NOIMBED - Put the sequence-set records on the same disk that contains the index records.
- **DDCREPLC** This field indicates whether or not VSAM will write each index record on one track of direct access (DASD) storage as many times as possible. If specified, the following interpretations apply.
  - **1** REPLICATE - VSAM will write each index record on a single track of DASD as many times as possible.

**0** NOREPLICATE - Each index record will appear on a track only once.

**DDCKLEN** The KEYLEN field shows, in bytes, the size of each record key in a non-VSAM data set, or the size of each key field in a key-sequenced VSAM data set. Possible values are 0 to 255 for non-VSAM data sets and 1 to 255 for VSAM data sets.

**DDCKOFF** The KEYOFF field applies only to key-sequenced VSAM data sets. The field shows, in bytes, the distance from the start of the record to the start of the key field. Possible values range from 0 to 32760.

**DDCCOMP** This field shows the data compaction type used for tape and is mapped by DDCCNUL, DDCNOCMP, and DDCIDRC. Compaction specifies whether or not mountable tape volumes associated with this data class are compacted. Compaction increases overall tape cartridge capacity.

**0** Null Compaction Type
**1** No Compaction.
**2** Improved Data Recording.

**DDCMEDIA** This field shows the type and format of the cartridges used for mountable tape data sets used with this data class. and is mapped by DDCMENUL, DDCMEDA1, and DDCMEDA2.

**0** Media type is null.

**1** Media 1 - Cartridge System

**2** Media 2 - Enhanced Capacity Cartridge System Tape

**DDCRECTE** This field indicates the number of recording tracks on the cartridge used for the mountable tape data sets associated with this data class.

**0** Recording Technology is not specified.
**1** Recording Technology is 18 track.
**2** Recording Technology is 36 track.

## Storage Class Construct Field

This is the section for Storage Class Construct information. These records are collected when SMSDATA is selected, and Storage Class constructs are defined to the control data set selected. The record type for this record is 'SC'.

| Name | Description |
|---|---|
| **DSCNMFLD** | The Storage Class Construct name. |
| | **DSCNMLEN** The length of this construct name. |
| | **DSCNAME** The name of this construct. |
| **DSCUSER** | The USERID of the last person to make a change to this construct. |
| **DSCDATE** | The date that this construct was last changed. The format is ″YYYY/MM/DD″ in EBCDIC. |
| **DSCTIME** | The time that this construct was last changed. The format is ″HH:MM″ in EBCDIC. |
| **DSCDESC** | The description of this construct. |
| **DSCFLAGS** | Storage Class Parameters Specification Flags. The following are indicated if the specified bit is '1'. |

**DSCDFGSP** This bit indicates that guaranteed space is to be allocated. Multi-volume data sets can be pre-allocated with the same or different amounts of space on more than one volume.

**DSCDFAVL** Availability options have been specified, see DSCAVAIL.

**DSCFDIRR** Direct access response time objective has been specified, see DSCDIRR.

**DSCFDIRB** Direct access bias has been specified, see DSCDIRB.

**DSCFSEQR** Sequential access response time objective has been specified, see DSCSEQR.

**DSCFSEQB** Sequential access bias has been specified, see DSCSEQR.

**DSCSYNCD** This bit indicates that the system should return from a BSAM CHECK issued for a WRITE against a PDSE member after (synchronized) the data has actually been written to a storage device.

**DSCFIAD** Initial access response time has been specified, see DSCIACDL.

**DSCFLAG2** Storage Class Parameters Specification Flags Byte 2. The following are indicated if the specified bit is '1'.

**DSCDFACC** Accessibility has been specified. See DSCACCES.

**DSCDFSDR** Striping Sustained Data Rate has been specified. See DSCSTSDR.

**DSCAVAIL** This field shows the availability options specified for the Storage Class and is mapped by DSCAVLDC, DSCAVLST, DSCAVLCN, and DSCAVLPR.

**0** Do not care about availability.
**1** Use standard availability.
**2** Use continuous availability.
**3** Continuous availability preferred.

**DSCDIRB** This field shows the direct access bias for data sets in this storage class. The direct access bias tells whether the majority of I/O scheduled for the data sets in this storage class is for READ, WRITE, or unknown. This field is mapped by DSCBIADC, DSCBIARD, and DSCBIAWR.

**0** Direct access bias is unknown.
**1** Direct access has read bias.
**2** Direct access has write bias.

**DSCSEQB** This field shows the sequential access bias for data sets in this Storage Class. The sequential access bias shows whether the majority of I/O scheduled for the data sets in this Storage Class is for READ, WRITE or unknown. This field is also mapped by DSCBIADC, DSCBIARD, and DSCBIAWR.

**0** Sequential access bias is unknown.
**1** Sequential access has read bias.
**2** Sequential access has write bias.

**DSCACCES** This field specifies whether the data sets in this storage class

should be allocated to volumes supported by concurrent copy. When used with the ABACKUP/BACKUP COPY TECHNIQUE attributes of the management class, this field determines if the data sets should retain continuous write access during backup.

**0** Continuous Preferred - The data set should be allocated to volumes supported by concurrent copy. If this cannot be done, a data set can be allocated to volumes that do not support concurrent copy.

**1** Continuous - The data set must be allocated to volumes supported by concurrent copy. The allocation is unsuccessful for data sets that cannot be allocated to such volumes.

**2** Standard - The data set should be allocated to volumes that do not support concurrent copy. If this cannot be done, a data set can be allocated to volumes that support concurrent copy.

**3** Nopref - The data set is allocated to volumes whether or not the volumes are concurrent copy capable.

**DSCIACDL** This field indicates the time required (in seconds) to locate, mount, and prepare media for data transfer.

**DSCDIRR** This field shows the direct access response time required for data sets in this storage class. The value is the number of milliseconds required to read or write a 4-kilobyte block of data.

**DSCSEQR** This field shows the sequential access response time required for data sets in this storage class. The value is the number of milliseconds required to read or write a 4-kilobyte block of data.

**DSCSTSDR** This field shows the sustained data transfer rate for data sets in this storage class. The system uses this value to determine the number of stripes it will try to allocate for the data set.

## Management Class Construct Field

This is the section for Management Class Construct information. These records are collected when SMSDATA is selected, and Management Class constructs are defined to the control data set selected. The record type for this record is 'MC'.

| Name | Description |
|---|---|
| **DMCNMFLD** | The Management Class Construct name.<br>**DMCNMLEN** The length of this construct name.<br>**DMCNAME** The name of this construct. |
| **DMCUSER** | The USERID of the last person to make a change to this construct. |
| **DMCDATE** | The date that this construct was last changed. The format is ″YYYY/MM/DD″ in EBCDIC. |
| **DMCTIME** | The time that this construct was last changed. The format is ″HH:MM″ in EBCDIC. |
| **DMCDESC** | The description of this construct. |
| **DMCSPEC1** | First byte of Management Class Parameters Specification Flags. The following are indicated if the specified bit is '1'. |

**DMCFBVER** The maximum number of backup versions for data sets in this Management Class has been specified.

**DMCFBVRD** The maximum number of backup versions for data sets in this Management Class to be retained after the data set has been deleted has been specified.

**DMCFRBK** The number of days to keep additional backup versions of data sets managed by this Management Class has been specified.

**DMCFRNP** The time period to retain the most recent backup copy of a data set after that data set has been deleted has been specified.

**DMCFEXDT** The expiration date for data sets or objects in this Management Class beginning with the creation date has been specified.

**DMCFEXDY** The number of days until the data sets or objects expire in this Management Class beginning with the creation date has been specified.

**DMCFPRDY** The number of days the data sets must remain unreferenced before they become eligible for migration in this Management Class has been specified.

**DMCSPEC2** Second byte of Management Class Parameters Specification Flags. The following are indicated if the specified bit is '1'.

**DMCFL1DY** Minimum days on Level 1 storage has been specified for data sets managed by this Management Class. See DMCL1DY.

**DMCFRLMG** Action for rolled-off GDS has been specified for this Management Class. See DMCRLOMG.

**DMCFPELE** The number of generation data group elements that can occupy primary storage in for this Management Class has been specified. See DMCPELEM.

**DMCFBKFQ** The backup frequency for data sets associated with this Management Class has been specified. See DMCBKFQ.

**DMCRLF** The partial release criteria for non-VSAM data sets in this Management Class. The bit combinations in this field show whether or not data sets in this Management Class can have unused space automatically released and the release conditions. This field is mapped by constants DMCRLFYE, DMCRLFCN, DMCRLFNO, DMCRLFYI, and DMCRLFCI. The following are indicated if the specified bit is '1'.

**DMCPREL** Unused space is released unconditionally.

**DMCPRCN** Unused space is released only if the data set has a secondary allocation.

**DMCPRIM** This bit indicates that the release is to be done either during the Space Management cycle or when the data set is closed, whichever comes first.

**DMCGDGFL** Generation Data Group (GDG) Attribute Flags.

**DMCRLOMG** This flag denotes the action to be done on a Generation Data Set (GDS) when it is rolled-off.

**1** The GDS is to be migrated after being removed from the GDG.

**0** The GDS is to expire after being removed from the GDG.

**DMCPELEM** This field shows how many of the newest generations of a Generation Data Group (GDG) can occupy primary storage. Any generations older than this set of newest generations are eligible to migrate. Possible values range from 0 to 255.

**DMCPEXPF** Data set expiration criteria flags. The following is indicated if the specified bit is '1'.

**DMCARNOL** This flag indicates that the number of days to keep the backup is unlimited.

**DMCEXPAC** This field indicates the action taken when a data set expires in this Management Class. It is mapped by DMCEANUL, DMCEADEL and DMCEARC.

**0** Expiration action is null.
**1** Expiration action is delete.
**2** Expiration action is backup.

**DMCRDARC** This field specifies the number of days to retain the backup copy of a data set managed by this Management Class.

**DMCRETF** Data set retention flags.

**DMCDYNOL** This flag indicates whether or not an expiration limit has been specified.

**1** No limit.

**0** See DMCEXPDY for expiration value.

**DMCDTNOL** This flag indicates whether or not an expiration date has been specified.

**1** No limit.

**0** See DMCEXDAT for expiration date.

**DMCRFMT** This field shows the format used by DMCEXDAT and is mapped by DMCNULL, DMCFDATE, and DMCFDAYS.

**0** Field was not used.
**1** Expire format: date/create.
**2** Expire format: days/create.

**DMCEXPDY** This field shows how many days an unaccessed data set or object in this Management Class can exist before expiring. Data sets or objects become eligible for expiration when the number of days since last access reaches the value in this field.

**DMCEXDAT** This field shows the expiration date for data sets or objects in this Management Class, or the number of days until the data sets or objects expire, beginning with the creation date.

**DMCEYEAR** Expire date since create. See DMCRFMT for format.

**DMCEDAY** Expire days since create. See DMCRFMT for format.

**DMCMIGF** Data set migration flags.

**DMCL1NOL** This flag indicates whether or not a limit has been specified for the number of days a data set can remain unaccessed before becoming eligible to migrate from Level 1 to Level 2.

- **1** No limit.
- **0** See DMCL1DY for this value.

**DMCPRDY** This field shows when the data sets in this Management Class become eligible for automatic migration. A value of 0 indicates that they are eligible upon creation. A value greater than 0 is the number of days the data sets must remain unreferenced before they become eligible for migration.

**DMCL1DY** This field is the number of days a data set must remain unaccessed before becoming eligible to migrate from Level 1 to Level 2.

**DMCCMAU** This field shows whether data sets in this Management Class can migrate between storage levels. The field also shows how migration, if allowed, can be initiated. It is mapped by DMCMNONE, DMCMCMD and DMCMBOTH.

- **0** Data sets cannot migrate between storage levels.
- **1** Data sets can migrate by command only.
- **2** Data sets can migrate either automatically or by command.

**DMCBKFLG** Data set backup flags.

**DMCRBNOL** This flag indicates whether or not a limit has been specified for how long the most recent backup copy of a data set is kept after the data set is deleted.

- **1** No limit.
- **0** See DMCBKNP for this value.

**DMCNPNOL** This flag indicates whether or not a limit has been specified for how long to keep backups of a data set that pre-date the most recent backup.

- **1** No limit.
- **0** See DMCBKDY for this value.

**DMCAUTBK** This flag shows whether or not automatic backup is allowed for data sets or objects in this Management Class.

- **1** Automatic backup is allowed.
- **0** Automatic backup is not allowed.

**DMCCPYTF** This flag indicates whether or not a backup copy technique had been specified for this Management Class.

- **0** No Copy Technique has been specified. Standard is assumed.
- **1** A Copy Technique had been specified. See DMCCPYTC.

**DMCBKFQ** This field represents the minimum number of days between

backups for data sets associated with this Management Class. A new backup of a data set can be made after this period of days only if the data set is changed during that period.

**DMCBKVS** This field shows whether automatic backups of an existing data set is kept. A value of 1 or higher represents the maximum number of such backups that can be kept at any one time. Only the most recent automatic backups can be kept. Each backup of a given data set will contain a different version of the data set. Possible values range from 1 to 13.

**DMCBKRD** This field shows whether automatic backups of a data set will be kept after the data set is deleted. A value of 0 means that no such backups are kept. A value of 1 or higher represents the maximum number that can be kept. Each automatic backup of a deleted data set contains a different version of the data set. Only the most recent backups are kept.

**DMCBKDY** This field shows how long to keep backups of a data set that pre-date the most recent backup. Each of these older backups will be kept for the period specified, regardless of whether the data set exists or has been deleted.

**DMCBKNP** This field shows how long the most recent backup copy of a data set is kept after the data set is deleted. A numeric value represents a specific number of days.

**DMCBADU** This field indicates who is authorized to perform command backups against the data sets in this Management Class. It is mapped by DMCBNONE, DMCBADM, and DMCMBOTH.

**0** Neither end users nor the Storage Administrator can perform command backups.

**1** Only the Storage Administrator can perform command backups.

**2** Both end users and the Storage Administrator can perform command backups.

**DMCCPYTC** This field indicates whether the concurrent copy technique should be used for the incremental backups of data sets associated with this Management Class. This attribute works in association with DSCASSOC to determine if the data set should retain write access during backup. This field is mapped by DMCCPYST, DMCCPYPR, and DMCCPYRQ.

**0** Standard - Indicates that all data sets are backed up without the concurrent copy technique.

**1** Concurrent Preferred - Indicates that the concurrent copy technique should be used for backup. A data set is backed up without the concurrent copy technique if it does not reside on a volume supported by concurrent copy or is otherwise unavailable for concurrent copy.

**2** Concurrent Required - Indicates that the concurrent copy technique must be used for backup. The backup is unsuccessful for data sets that do not reside on volumes supported by concurrent copy or are otherwise unavailable for concurrent copy.

**DMCBKUDC** The name of the backup destination class.

**DMCMRETF** Maximum retention flags.

**DMCRPNOL** This flag indicates whether or not a retention limit exists for data sets in this Management Class.

**1** No limit. This allows an unlimited retention period or expiration date.

**0** See DMCMRTDY for this value.

**DMCMRTDY** This field shows whether the Storage Management Subsystem (SMS) will use the retention period (RETPD) or expiration date (EXPDT) that a user or Data Class specifies for a data set. If the value is 0, SMS will not use the specified retention period or expiration date.

**DMCTSCR** Time since creation flags. The following is specified if the specified bit is '1'.

**DMCTCYR** The number of years that must pass since the creation date before class transition occurs has been specified.

**DMCTCMN** The number of months that must pass since the creation date before class transition occurs has been specified.

**DMCTCDY** The number of days that must pass since the creation date before class transition occurs has been specified.

**DMCTSLU** Time since last used flags. The following is specified if the specified bit is '1'.

**DMCTSYR** The number of years that must pass since the last reference date before class transition occurs has been specified.

**DMCTSMN** The number of months that must pass since the last reference date before class transition occurs has been specified.

**DMCTSDY** The number of days that must pass since the last reference date before class transition occurs has been specified.

**DMCPERD** Period and day on which class transition occurs flags. The following is indicated if the specified bit is '1'.

**DMCPEMN** The day of each month on which transition occurs has been specified.

**DMCPEQD** The day of each quarter on which transition occurs has been specified.

**DMCPEQM** The month of each quarter on which transition occurs has been specified.

**DMCPEYD** The day of each year on which transition occurs has been specified.

**DMCPEYM** The month of each year on which transition occurs has been specified.

**DMCFIRST** The first day of each period on which transition occurs has been specified.

**DMCLAST** The last day of each period on which transition occurs has been specified.

**DMCVSCR** Time since creation fields.

**DMCVSCY** This field indicates the number of years that must pass since the creation date before class transition occurs.

**DMCVSCM** This field indicates the number of months that must pass since the creation date before class transition occurs.

**DMCVSCD** This field indicates the number of days that must pass since the creation date before class transition occurs.

**DMCVSLU** Time since last used fields.

**DMCVSUY** This field indicates the number of years that must pass since the last reference date before class transition occurs.

**DMCVSUM** This field indicates the number of months that must pass since the last reference date before class transition occurs.

**DMCVSUD** This field indicates the number of days that must pass since the last reference date before class transition occurs.

**DMCVPRD** Periodic values.

**DMCVPMD** This field shows the day of each month that class transition occurs.

**DMCVPQT** Periodic quarterly values.

**DMCVPQD** This field shows the day of each quarter that class transition occurs. If the DMCVPQM field is also specified, this field indicates the day of the month in each quarter that class transition occurs.

**DMCVPQM** This field shows the month of each quarter that class transition occurs.

**DMCVPYR** Yearly values.

**DMCVPYD** This field shows the day of each year that class transition occurs. If the DMCVPYM field is also specified, this field indicates the day of the month in each quarter that class transition occurs.

**DMCVPYM** This field shows the month of each year that class transition occurs.

## Base Configuration Field

This is the section for Base Configuration information. Only one of these records is collected when SMSDATA is selected. The record type for this record is ’BC’.

| Name | Description |
|---|---|
| **DBCUSER** | The USERID of the last person to make a change to this configuration |
| **DBCDATE** | The date that this configuration was last changed. The format is ″YYYY/MM/DD″ in EBCDIC. |
| **DBCTIME** | The time that this configuration was last changed. The format is ″HH:MM″ in EBCDIC. |
| **DBCDESC** | The description of this configuration. |
| **DBCFLAGS** | Flags used for base configuration information. These flags are all reserved. |
| **DBCDEFMC** | This field identifies the default management class. DFHSM uses the default management class for expiration, migration, class transition and backup information for data sets that do not have a management class assigned.<br>**DBCMCLEN** This field contains the length of the default management class name.<br>**DBCMCNAM** Name of the default management class. |
| **DBCDGEOM** | This field contains the default device geometry. The default device geometry isolates the user from the actual physical device where SMS places their data sets.<br>**DBCTRKSZ** Bytes per track. This value represents the number of bytes per track that SMS uses on allocations.<br>**DBCCYLCP** Tracks per cylinder. This value represents the number of tracks per cylinder that SMS uses on allocations. |
| **DBCDUNIT** | Default Unit. This field is an esoteric or generic device name, such as SYSDA or 3390 that applies to data sets that are not managed by SMS. |
| **DBCSRST** | The SMS Resource Status Token for this configuration. |
| **DBCSTAT** | The status of the SCDS. The possible values are:<br>**1** SCDS is valid.<br>**2** SCDS is not valid.<br>**3** SCDS status is unknown. |
| **DBCFSYSN** | This field indicates the global resource serialization systems defined to the complex. This field contains eight system names. |
| **DBCSCDSN** | The name of the SCDS from which this ACDS was activated. This field will only contain a name when the ACDS parameter has been specified. |
| **DBCSEPNL** | The length of the separation name profile, from 0 to 54, valid only when DCUVERS is two or higher for this record. |
| **DBCSEPNM** | The separation profile name, valid only when DCUVERS is two or higher and DBCSEPNL is nonzero. |
| **DBCSFEAT** | The value in this field shows the supported system features of the eight systems named in DBCFSYSN. |
| **DBCSYSDT** | This field shows status by processor, and can have up to 32 system status entries. |

**DBCSYSNM** System/group names, up to 32. Similiar to DBCSYSN.
**DBCSYSFT** Supported system features, up to 32. Similar to DBCSFEAT.
**DBCSNMTY** System name type, up to 32. Similiar to DBCSYSNT.

## Aggregate Group Construct Field

This is the section for Aggregate Group Construct information. These records are collected when SMSDATA is selected, and Aggregate Group constructs are defined to the control data set selected. The record type for this record is 'AG'.

| Name | Description |
|---|---|
| **DAGNMFLD** | The Aggregate Group Construct name.<br>**DAGNMLEN** The length of this construct name.<br>**DAGNAME** The name of this construct. |
| **DAGUSER** | The USERID of the last person to make a change to this construct. |
| **DAGDATE** | The date that this construct was last changed. The format is ″YYYY/MM/DD″ in EBCDIC. |
| **DAGTIME** | The time that this construct was last changed. The format is ″HH:MM″ in EBCDIC. |
| **DAGDESC** | The description of this construct. |
| **DAGFLAGS** | Aggregate Group Construct Flags.<br>**DAGTENQ** This bit indicates whether or not SMS should tolerate enqueue errors. '1'X indicates that the error should be tolerated.<br>**DAGFRET** Retention period attribute specified.<br>**DAGFNCPY** Number of copies attribute specified (DAGNCOPY). |
| **DAGRETPD** | Retention period in days assigned to backup versions by this Aggregate Group if DAGFRET is set to '0'. If DDCFEXP is '1' then this field should be interpreted using the two following fields. Backup versions are deleted or archived either one day after the retention period or on the expiration date.<br>**DAGEXPYR** Expiration Year - year assigned to backup versions by this Aggregate Group.<br>**DAGEXPDY** Expiration Day - absolute day of year assigned to backup versions by this Aggregate Group. |
| **DAGDEST** | The remote location of the backup volumes. |
| **DAGPREFX** | The prefix of the output data sets allocated by the backup process. The output data sets allocated are generation data groups. The system will append one of the following suffixes to the name specified:<br>**.D.G000n.Vnn** for data sets.<br>**.C.G000n.Vnn** for control data sets. |
| **DAGIDSNM** | The name of the data set containing instructions, commands, and so on, that are copied into the control file volume after the backup control file.<br>**DAGINDSN** The name of the instruction data set. |

**DAGINMEM** The member name, if any.

**DAGDSNMS** The name of the data set containing lists of data sets to be included in the application backup. There can be up to five selection data sets.
**DAGDSN** The name of one of the selection data sets.
**DAGMEM** The member name, if any.

**DAGMGMTC** This field shows the name of the Management Class from which the aggregate group backup attributes are obtained.
**DAGMCLEN** The length of the Management Class name
**DAGMCNAM** The Management Class Name

**DAGNCOPY** This field specifies how many copies of the aggregate backup output files are to be created. The aggregate backup output file consists of an instruction activity log file, a control file, and one or more data files.

## Storage Group Construct Field

This is the section for Storage Group Construct information. These records are collected when SMSDATA is selected, and Storage Group constructs are defined to the control data set selected. The record type for this record is 'SG'.

| Name | Description |
|---|---|
| **DSGNMFLD** | The Storage Group Construct name.<br>**DSGNMLEN** The length of this construct name.<br>**DSGNAME** The name of this construct. |
| **DSGUSER** | The USERID of the last person to make a change to this construct. |
| **DSGDATE** | The date that this construct was last changed. The format is ″YYYY/MM/DD″ in EBCDIC. |
| **DSGTIME** | The time that this construct was last changed. The format is ″HH:MM″ in EBCDIC. |
| **DSGDESC** | The description of this construct. |
| **DSGEXNLN** | Length of storage group extend name from 0 to 30. Valid only when DCUVERS is two or higher. |
| **DSGEXNM** | Storage group extend name, valid only when DCUVERS is two or higher, and DSGEXNLN is nonzero. |
| **DSGFLAGS** | Storage Group Flags. The following are indicated if the specified bit is '1'.<br>**DSGFABUP** This bit indicates that the data sets on the volumes in this Storage Group are eligible for automatic backup.<br>**DSGFAMIG** This bit indicates that the data sets on the volumes in this Storage Group are eligible for automatic migration.<br>**DSGFADMP** This bit indicates that this Storage Group can be automatically backed up using DFSMShsm, or a comparable product. Dumping whole volumes instead of individual data sets speeds up the process of restoring data sets to volumes. |

**DSGFTHRS** This flag indicates that thresholds have been specified for this Storage Group.

**DSGFGBKU** This flag indicates that the maximum number of days between backups has been specified.

**DSGGBNOL** This flag indicates that the maximum number of days between backups has no limit.

**DSGFIMIG** This flag indicates that the data sets in this Storage Group are eligible for interval migration. DSGFHTHR and DSGFLTHR must be specified.

**DSGFPSM** This flag indicates systems features. See constants for DBCSYSFT in Table 26 on page 438.

**DSGFTYPE** This field denotes the type of storage group to which the volumes belong. This field is mapped by the constants DSGPOOL, DSGVIO, DSGDUMMY, DSGOBJ, and DSGOBJBK.

**0** Storage Group type is POOL.
**1** Storage Group type is VIO.
**2** Storage Group type is DUMMY.
**3** Storage Group type is OBJECT.
**4** Storage Group type is OBJECT BACKUP.

**DSGFHTHR** This field is the high threshold, or the percentage of a single volume in the Storage Group at which DFSMShsm will migrate data sets off all the volumes if any of them meet or exceed this value. Processing continues on each volume until either the volume occupancy meets or goes below the value in DSGFLTHR or no more data sets on the volume are eligible for migration. This value is not used during automatic migration. Possible values range from 0 to 99.

**DSGFLTHR** This field is the low threshold value. During interval and automatic migration, this value is used as the target for the percentage of space allocated on each volume in the storage group. DFSMShsm will migrate eligible data sets off a volume until either the space allocated on the volume drops to or below this value, or no more data sets on the volume are eligible to be migrated. This value is ignored if DSGFAMIG is ’0’. Possible values range from 0 to 99.

**DSGFVMAX** This field shows the largest size of a virtual input/output (VIO) data set, in kilobytes, that you can create for this storage group. You cannot allocate data sets that exceed this field in this Storage Group. This value applies to VIO storage groups only.

**DSGFVUNT** The value in this field shows the type of physical device that will be simulated by the Storage Group. At least one unit of the device type shown must be physically connected to each system that has access to the storage group. This value appears for VIO Storage Groups only.

**DSGDMPCL** The five elements in this array show the names of the dump classes assigned to this Storage Group.

**DSGFRPST** The eight elements in this array show status of this Storage Group by processor. Each element can be referenced by using DSGSTAT.

**DSGSTAT** This field shows the status of the Storage Group on

a given processor and is mapped by the constants DSG0, DSGENBL, DSGQUI, DSGQUIN, DSGDIS and DSGDISN.

**0** No status is specified.

**1** Storage Group status is enabled. A relationship that allows a system to allocate and access data sets in a VIO Storage Group, a pool Storage Group, or individual volumes within a pool Storage Group.

**2** Storage Group status is quiesce all. A relationship that prevents a system from scheduling jobs that allocate or access data sets in a VIO Storage Group, a pool Storage Group, or individual volumes within a pool Storage Group.

**3** Storage Group status is quiesce new. A relationship that prevents a system from scheduling jobs that allocate new data sets or modify existing ones in a VIO Storage Group, a pool Storage Group, or individual volumes within a pool Storage Group.

**4** Storage Group status is disable all. A relationship that prevents a system from allocating or accessing data sets in a VIO Storage Group, a pool Storage Group, or individual volumes within a pool Storage Group.

**5** Storage Group status is disable new. A relationship that prevents a system from allocating new data sets in a VIO Storage Group, a pool Storage Group, or individual volumes within a pool Storage Group.

**DSGABSYS** This field shows the name of the system on which automatic backup of the volumes in this Storage Group will take place.

**DSGADSYS** This field shows the name of the system on which automatic dumping of the volumes in this Storage Group will take place.

**DSGAMSYS** This field shows the name of the system which will perform automatic migration and space management of the volumes in this Storage Group.

**DSGCNFRM** The eight elements of this array show the confirmed SMS status of this Storage Group. Each element can be referenced by DSGCSMSS.

**DSGCSMSS** This field shows the confirmed SMS status of the storage group and maps to the same values as DSGSTAT.

**DSGGBKUF** The value in this field indicates the maximum number of days between backups. During this backup period, a copy of each data set in the Storage Group is available. This field is valid only for Pool Storage Groups.

**DSGTBLGR** This field shows the OAM table space identifier for this Storage Group in the form GROUPnn.

**DSGOAMFL** This field shows the OAM flags for this Storage Group.

**DSGFCYS** This flag indicates whether or not the start and end times when the OAM Storage Management Component (OSMC) can automatically start its storage management processing for this Storage Group.

**1** These values have been given, see DSGCYLST and DSGCYLED.

**0** These values have not been given.

**DSGFVLFT** This flag indicates whether or not the number of free sectors required for an optical volume within this Storage Group has been specified.

**1** This value has been given, see DSGVOLFT.

**0** This value has not been given.

**DSGFDRST** This flag indicates whether or not the maximum number of object write requests outstanding for an optical drive in this Storage Group has been given.

**1** This value has been given, see DSGDRVST.

**0** This value has not been given.

**DSGVFFER** This flag indicates whether or not a ″mark volume full on first write failure″ criteria has been specified for this Storage Group.

**1** This value has been specified, see DSGVFERR.

**0** This value has not been specified.

**DSGVFERR** This flag indicates whether or not a ″mark volume full on first write failure″ criteria applies to optical volumes in this Storage Group.

**1** This value indicates that OAM marks an optical volume full the first time an attempt to write an object on the optical volume is unsuccessful because there is not enough space remaining.

**0** This value indicates that OAM marks an optical volume full only

when the number of available sectors in the user data area falls below the volume full threshold specified in DSGVOLFT.

**DSGCYLST** This field shows the beginning of a window of time in which the Object Access Method can begin storage management processing. The actual value this field represents is an hour of the day on a 24-hour scale. This value is valid only for the OBJECT Storage Group type. Possible values range from 0 to 23.

**DSGCYLED** This field shows the end of a window of time in which the Object Access Method can begin storage management processing. The actual value this field represents is an hour of the day on a 24-hour scale. This value is valid only for the OBJECT Storage Group type. Possible values range from 0 to 23.

**DSGVOLFT** This field shows the number of free sectors required for an optical volume within this storage group. When the number of free sectors falls below the threshold, the Object Access Method marks the optical volume as full. This value is valid only for the OBJECT and OBJECT BACKUP Storage Group types.

**DSGDRVST** This field shows the maximum number of object write requests outstanding for an optical drive in this storage group. When the number of object write requests to this storage group divided by the number of optical disk drives currently processing requests for this storage group exceeds this threshold, the Object Access Method attempts to start an additional optical disk drive. This value is valid only for the OBJECT and OBJECT BACKUP Storage Group types.

**DSGOLIBS** These eight array elements list the library names that represent defined optical drive configurations available for this storage group, or a pseudo-library name that represents stand-alone optical drives and shelf-resident optical volumes. This array is valid for OBJECT and OBJECT BACKUP Storage Group types only. Each element can be referenced by DSGOLBNL and DSGOLBNM.

**DSGOLBNL** Optical library name length.
**DSGOLBNM** Optical library name.

**DSGSSTAT** This field shows status by processor, and can have up to 32 system status entries.

**DSGSYSST** Requested system status. Similiar to DSGSTAT.
**DSGCNSMS** Confirmed SMS status. Similar to DSGCSMSS.

## Volume Definition Field

This is the section for SMS Volume Definition information. These records are collected when SMSDATA is selected, and SMS Volume Definitions are defined to the control data set selected. The record type for this record is 'VL'.

| Name | Description |
|---|---|
| **DVLNMFLD** | The Storage Group Construct name.<br>**DVLNMLEN** The length of this construct name.<br>**DVLNAME** The name of this construct. |
| **DVLUSER** | The USERID of the last person to make a change to this construct. |
| **DVLDATE** | The date that this construct was last changed. The format is ″YYYY/MM/DD″ in EBCDIC. |

**DVLTIME** The time that this construct was last changed. The format is ″HH:MM″ in EBCDIC.

**DVLFLAGS** Volume Definition Flags.

- **DVLCONV** Volume conversion flag. If this flag is '1' then the volume is in conversion.

**DVLSG** This area shows the name of the Storage Group this volume belongs to, if any.

- **DVLSGLEN** The length of the Storage Group name.
- **DVLSGGRP** The Storage Group name.

**DVLNSTAT** The eight elements in this array show volume status by system. Each element can be referenced by DVLSMSS and DVLMVSS.

- **DVLSMSS** The field shows the SMS status of the volume for a given system. It is mapped by DVL0, DVLENBL, DVLQUI, DVLQUIN, DVLDIS, and DVLDISN.
  - **0** No status is given.
  - **1** Full access enabled by SMS.
  - **2** Job access disabled by SMS.
  - **3** New job access disabled by SMS.
  - **4** Job access disabled by SMS.
  - **5** New allocation disabled by SMS.
- **DVLMVSS** The field shows the SMS status of the volume for a given system. It is mapped by DVLONLN, DVLOFFLN, DVLPOFF, DVLBOXED, and DVLNRDY.
  - **1** Online.
  - **2** Offline.
  - **3** Pending offline.
  - **4** Boxed.
  - **5** Not ready.

**DVLCSMSS** This eight element array shows the confirmed SMS status of the volume by system and maps to the same values as DVLSMSS.

**DVLNUCBA** This field shows the address of this volume's Unit Control Block (UCB) if known. Otherwise this field is equal to 0.

**DVLNTCPY** This field shows the total capacity of the volume in megabytes.

**DVLNFREE** This field shows the total amount of free space in megabytes.

**DVLNLEXT** This field shows the largest free extent in megabytes.

**DVLN0CNT** This field shows the volume level reset count.

**DVLTRKSZ** This field shows the volume R1 track capacity.

**DVLNLEVL** This field shows the update level for the volume.

**DVLSSTAT** This field shows status by processor, and can have up to 32 system status entries.

- **DVLSTSMS** SMS system status. Similar to DVLSMSS.
- **DVLSTMVS** MVS system status. Similar to DVLMVSS.
- **DVLCNSMS** Confirmed SMS status. Similar to DVLCSMSS.

## Optical Drive Information Field

This is the section for Optical Drive Information. These records are collected when SMSDATA is selected, and Optical Drives are defined in the control data set selected. The record type for this record is 'DR'.

| Name | Description |
|---|---|
| **DDRNMFLD** | The Optical Drive name. |
| **DDRDVLEN** | The length of this construct name. |
| **DDRNAME** | The full field for the name. |
| **DDRDNAME** | The eight character name of the Optical Drive |
| **DDRDUSER** | The USERID of the last person to make a change to this construct. |
| **DDRDDATE** | The date that this construct was last changed. The format is ″YYYY/MM/DD″ in EBCDIC. |
| **DDRDTIME** | The time that this construct was last changed. The format is ″HH:MM″ in EBCDIC. |
| **DDRLB** | The one to eight character name of the library to which the drive is assigned. For stand-alone drives, this field is the name of a pseudo library.<br>**DDRLBLEN** The length of the library name.<br>**DDRLIBRY** The full field for the library name.<br>**DDRLBNM** The one to eight character name of the Library. |
| **DDRNSTAT (8)** | This field contains status information for the drive for all eight possible systems.<br>**DDROMST** This field contains the status information for each drive.<br>**DDRSOUT** The requested OAM status.<br>**0** No Connectivity<br>**1** Online<br>**2** Offline<br>**3** No Outstanding Request<br>**DDRCFCS** The current OAM status.<br>**0** No Connectivity<br>**1** Online<br>**2** Offline<br>**3** No Outstanding Request |
| **DDRDCONS** | This field specifies the name of the MVS console that is associated with the optical drive. |
| **DDRSTAT** | This field shows status by processor, and can have up to 32 system status entries.<br>**DDRSYSST** System status.<br>**DDRREQST** Requested status. Similar to DDRSOUT.<br>**DDRCURST** Current status. Similar to DDRCFCS. |

## Library Information Field

This is the section for Library Information. These records are collected when SMSDATA is selected, and Libraries are defined in the control data set selected. The record type for this record is 'LB'.

**Name** **Description**

**DLBNMFLD** The Optical or Tape Library name.
- **DLBNMLEN** The length of this construct name.
- **DDLBLNAME** The full field for the name.
- **DLBNAME** The eight character name of the Optical or Tape Library

**DLBDUSER** The USERID of the last person to make a change to this construct.

**DLBDDATE** The date that this construct was last changed. The format is ″YYYY/MM/DD″ in EBCDIC.

**DLBDTIME** The time that this construct was last changed. The format is ″HH:MM″ in EBCDIC.

**DLBNSTAT (8)**
This field contains status information for the optical library for all eight possible systems.

- **DLBOMST** This field contains the status information for each library.
  - **DLBSOUT** The requested OAM status.
    - **0** No Connectivity
    - **1** Online
    - **2** Offline
    - **3** No Outstanding Request
  - **DLBCFCS** The current OAM status.
    - **0** No Connectivity
    - **1** Online
    - **2** Offline

**DLBTYPE** Specifies the library type. This field contains either REAL or PSEUDO. A real library is a physical library containing from 1 to 4 drives, while a pseudo library is a library consisting of stand-alone drives only.
- **0** Real Library
- **1** Pseudo Library

**DLBDTYPE** The library device type for this library
- **0** The IBM 9426 Library
- **1** The IBM 3995 Library
- **2** Tape Library

**DLBDCONS** This field specifies the name of the MVS console that is associated with the library. Associating a console with a manual tape library allows MVS to direct messages for that library to its console. MVS continues to use the normal MVS routing code information for automated tape libraries and manual tape libraries with no specified console name.

**DLBEDVT** This field specifies the default volume type for inserted tape cartridges. A value of PRIVATE means the tape cartridge can only

be used by referencing its volume serial number. A value of SCRATCH means you can use the tape cartridge to satisfy a non-specific volume request. This field is mapped by the constants DLBPRVT and DLBSCRT.

**1** Private
**2** Scratch

**DLBEJD** This field specifies the default action for the Tape Configuration Data Base volume record when a tape cartridge is ejected from this library. A value of PURGE means the volume record is deleted from the Tape Configuration Data Base. A value of KEEP means the volume record is not deleted from the Tape Configuration Data Base. This field is mapped by the constants DLBPURGE and DLBKEEP.

**1** Purge TCDB Volume Record
**2** Keep TCDB Volume Record

**DLBLCBID** This field specifies the EBCDIC representation of the five digit hexadecimal library sequence number returned by the tape control unit in response to a Read Device Characteristics command - the value placed in the library hardware at the time it was installed. The LIBRARY ID connects the library name to the library hardware.

**DLBEDUNM** This field specifies the entry default unit name of the library.

**DLBDEFDC** This field specifies the default data class for inserted tape cartridges. The data class name must identify a valid data class whose recording technology, media type and compaction parameters are used as the default values if the cartridge entry installation exit does not supply them. All other data class parameters are ignored.

**DLBDCLEN** This field specifies the length of the data class name.

**DLBDCLNM** This field contains the entire 32 characters for the data class name. Only eight characters are used in today's environment.

**DLBDCNAM** This field contains the eight character data class name.

**DLBSTAT** This field shows status by processor, and can have up to 32 system status entries.

**DLBSYSST** System status.
**DLBREQST** Requested system status. Similar to DLBSOUT.
**DLBCURST** Current system status. Similar to DLBCFCS.

## Migrated Data Set Record Field

This is the section for migrated data set information. If migrated data set information records are requested, one record is created for each migrated data set represented in the MCDS. The record type for this record is M.

| Name | Description |
|---|---|
| **UMDSNAM** | Identifies the original name of this data set. |
| **UMLEVEL** | Identifies the migration level where this migrated data set is currently residing. |
| **UMCHIND** | Indicates, when the flag bit is set to a 1, that this data set has changed since the last time it was backed up. |

**UMDEVCL** Identifies whether the migrated data set is currently residing on DASD or tape.

**UMDSORG** Data set organization.

**UMDSIZE** Indicates the size in kilobytes of the migrated data set. If compaction is used, this value represents the compacted size.

**UMMDATE** Contains the time and date that the data set was migrated from a level 0 volume. The format is packed decimal. The date (*yyyydddF*) indicates the year(*yyyy*) and day(*ddd*). The time (*hhmmssth*) indicates the hours(*hh*), minutes(*mm*), and seconds(*ss*) including tenths(*t*) and hundredths(*h*) of a second.

**UMCLASS** Contains the SMS data class, storage class, and management class of the data set at the time the data set was migrated from a level 0 volume.

**Note:** If UMSCLNG, which contains the length of the storage class name, is zero, the data set is not SMS-managed. If the field is non-zero, the data set is SMS-managed.

**UMRECRD** Record format of this data set.

**UMRECOR** VSAM organization of this data set.

**UMBKLNG** Block length of this data set.

**UMRACFD** Indicates, when the flag bit is set to a 1, that this data set is RACF-indicated (protected by a discrete RACF profile).

**UMGDS** Indicates, when the flag bit is set to a 1, that this data set is an SMS-managed generation data set.

**UMREBLK** Indicates, when the flag bit is set to a 1, that this data set is an SMS-managed reblockable data set.

**UMPDSE** Indicates, when the flag bit is set to a 1, that this data set is a partitioned data set extended (PDSE) data set (DSNTYPE=LIBRARY).

**UMCOMPR** Indicates, when the flag is set to 1, that this data set is in compressed format.

**UMNMIG** Contains the number of times that a data set has migrated from a user volume.

**UMALLSP** Indicates the space (in kilobytes) that was originally allocated when this data set was migrated from a level 0 volume.

**UMUSESP** Indicates the space (in kilobytes) that actually contained data when this data set was migrated from a level 0 volume.

**UMRECSP** Indicates the estimated space (in kilobytes) required if this data set is recalled to a level 0 volume of

| | |
|---|---|
| | similar geometry, using a similar blocking factor. Actual space used will depend on blocking factor and device geometry. |
| **UMCREDT** | Contains the date (*yyyyddd*F) on which this data set was created on a level 0 volume. This field is valid only for SMS-managed data sets. |
| **UMEXPDT** | Contains the date (*yyyyddd*F) on which this data set expires. |
| **UMLBKDT** | Contains the date on which this data was last backed up (STCK format). This field is valid only for SMS-managed data sets. |
| **UMLRFDT** | Contains the date (*yyyyddd*F) in which this data set was last referred to. |
| **UM_USER_DATASIZE** | Contains, when UMCOMPR is set to 1, the size (in kilobytes) this data set would be if it were not compressed. |
| **UM_COMP_DATASIZE** | Contains, when UMCOMPR is set to 1, the actual compressed size (in kilobytes) of the data set. |

## Backup Data Set Record Field

This is the section for backup data set information. If backup information records are requested, one record is created for each backup version represented in the BCDS. The record type for this record is B.

| Name | Description |
|---|---|
| **UBDSNAM** | Identifies the name of the data set to which this backup version applies. |
| **UBINCAT** | Indicates the cataloged version of a data set name that is used by multiple data sets. For example, two different data sets (one cataloged and the other not in the catalog) can have the same name. This bit indicates which data set is being referred to with the data set name. When this bit is set to a 1, this backup refers to the cataloged data set. |
| **UBNOENQ** | Because DFSMShsm was directed not to serialize, the data set was unserialized while it was backed up. |
| **UBBWO** | Indicates that this data set was backed up while the data set was a backup-while-open candidate. |
| **UBNQN1** | The data set was unserialized while it was backed up. Even though the serialization failed on the first attempt, DFSMShsm was directed to accept the backup result without a retry. |
| **UBNQN2** | The data set was unserialized while it was backed up. The first backup attempt failed because the data set was in use and DFSMShsm was directed to retry. DFSMShsm accepted the result of the retry, even though serialization failed again. |

**UBDEVCL** Identifies the device on which the backup version is currently residing (DASD or tape).

**UBDSORG** Indicates the data set organization.

**UBDSIZE** Indicates the size (in kilobytes) of the backup version. If compaction is used, this value represents the compacted size.

**UBBDATE** Contains the time and date that the backup version was made for the data set. The format is packed decimal. The date (*yyyyddd*F) indicates the year(*yyyy*) and day(*ddd*). The time (*hhmmssth*) indicates the hours(*hh*), minutes(*mm*), and seconds(*ss*), including tenths(*t*) and hundredths(*h*) of a second.

**UBCLASS** Contains the SMS data class, storage class, and management class of the data set at the time the backup version was made for the data set.

**Note:** If UBSCLNG, which contains the length of the storage class name, is zero, the data set was not SMS-managed at the time that this backup copy was made. If UBSCLNG is non-zero, the data set was SMS-managed when this backup copy was made.

**UBRECRD** Indicates the data set record format.

**UBRECOR** Indicates the VSAM data set organization.

**UBBKLNG** Indicates the block length of this data set.

**UBRACFD** Indicates, when this flag bit is set to a 1, that this data set was RACF-indicated at the time that it was backed up.

**UBGDS** Indicates, when this flag bit is set to a 1, that this is a backup copy of an SMS-managed generation data set (GDS).

**UBREBLK** Indicates, when this flag bit is set to a 1, that this is a copy of an SMS-managed system-reblockable data set.

**UBPDSE** Indicates, when this flag bit is set to a 1, that this is a backup copy of a partitioned data set extended (PDSE) data set (DSNTYPE=LIBRARY).

**UBCOMPR** Indicates, when the flag is set to 1, that the data set is in compressed format.

**UBALLSP** Indicates the space (in kilobytes) that was originally allocated on a level 0 volume when this data set was backed up from that volume. If backup is done while the data set is migrated, this value represents the size of the migrated copy of the data set. If the data set is compacted during migration, the size might be smaller than the original level 0 data set.

| | |
|---|---|
| **UBUSESP** | Indicates the space (in kilobytes) of actual data in the data set at the time that the data set was backed up. |
| **UBRECSP** | Indicates the estimated space (in kilobytes) required if this data set is recovered to a level 0 volume of similar geometry, using a similar blocking factor. Actual space used will depend on blocking factor and device geometry. |
| **UB_USER_DATASIZE** | Contains, when UBCOMPR is set to 1, the size (in kilobytes) this data set would be if not compressed. |
| **UB_COMP_DATASIZE** | Contains, when UBCOMPR is set to 1, the actual compressed size (in kilobytes) of the data set. |

## DASD Capacity Planning Record Field

This is the section for DASD capacity planning information. If DASD capacity planning records are requested, one record is created for each level 0 and level 1 volume for each day there was activity. For example, if five volumes had DFSMShsm activity for seven days, there would be 35 DASD capacity planning records. The number of days that volume statistics are kept to create these records can be controlled by the MIGRATIONCLEANUPDAYS parameter of the DFSMShsm SETSYS command. The record type for this record is C.

| Name | Description |
|---|---|
| **UCVOLSR** | Identifies the serial number of the volume. |
| **UCCOLDT** | Identifies the date on which statistics were collected for this volume. Only the days where there was activity on this volume are recorded. Records are not created for the current day, as the statistics are incomplete and do not represent a full 24-hour period. |
| **UCLEVEL** | Identifies the volume level. Only level 0 and migration level 1 volumes are recorded. |
| **UCTOTAL** | Indicates the total capacity of this volume in kilobytes. |
| **UCTGOCC** | Indicates the target occupancy, or low threshold, assigned to this volume. This is the percentage of the volume that you want to contain data after migration processing. The percentage can range from 0 to 100. This field does not apply to migration level 1 volumes. |
| **UCTROCC** | Indicates the high threshold assigned to this volume. When this percentage of the volume is filled with data, it indicates that you should run interval migration except for volumes in storage groups with AM=I.<br><br>For storage groups with AM=I, the percentage of the volume is the midpoint between UCTGOCC and UCTROCC that indicates when you should run interval migration. |
| **UCBFOCC** | Indicates the occupancy of the volume before it has been processed by either automatic primary or automatic secondary space management. For level 0 (user) volumes, this processing is done during automatic primary space management. For migration level 1 volumes, this processing is done during automatic secondary space management. This is a percentage value ranging from 0 to 100. |

**UCAFOCC** Indicates the occupancy of the volume after it has been processed by either automatic primary or automatic secondary space management. For level 0 (user) volumes, this processing is done during automatic primary space management. For migration level 1 volumes, this processing is done during automatic secondary space management. This is a percentage value ranging from 0 to 100.

**UCNOMIG** Indicates the excess eligible data occupancy for level 0 and level 1 volumes.

**For level 0 volumes:** This is the percentage of the level 0 volume that contains data eligible for migration (based on type and date-last-referenced) that did not migrate because the desired volume occupancy was met without it being migrated. This percentage, ranging from 0 to 100, can be considered the safety margin for automatic primary space management.

**For level 1 volumes:** This is the percentage of the level 1 volume that contains data eligible for migration that did not migrate:

The percentage value is *valid* only if one of the following conditions is met:

- The automatic secondary space management window is too short and MGCFDCOL field is patched to X'FF' to allow DFSMShsm to collect the DCOLLECT data. This percentage value will indicate the percentage of the migrated data sets that are eligible to be migrated but are not migrated because the secondary space management window is too short.
- None of the level 1 volumes trigger the level 1 to level 2 migration because none of their thresholds is met or exceeded. No data movement is initiated from any level 1 volume. This percentage value will indicate the percentage of the migrated data sets that are eligible to be migrated but are not migrated because of the thresholds settings.
- The automatic secondary space management, including migration cleanup and level 1 to level 2 migration, runs to completion. In this case, all data sets eligible for migration are migrated and hence the percentage of the migrated data sets that are eligible to be migrated, but are not migrated, will be zero.

The percentage value is *not valid* if one of the following conditions occurs:

- Automatic secondary space management does not run for the day because it is not scheduled to run, N-day in cycle. In this case, the percentage of the migrated data sets that are eligible to be migrated, but are not migrated, will be zero.
- Automatic secondary space management runs but does not complete because:
  - DFSMShsm is shut down
  - DFSMShsm is in emergency mode
  - automatic migration is held
  - a target migration volume is not available
  - the maximum number of unsatisfactory or unexpected CDS records is encountered

– Secondary space management ending time is reached and DFSMShsm is not asked to collect the data by patching the MGCFDCOL field to X'FF'. In this case, the percentage of the migrated data sets that are eligible to be migrated, but are not migrated, will have partial results depending on where the SSM stopped.

**UCNINTV** Contains the number of times interval migration has processed this volume on this day. This field does not apply to migration level 1 volumes.

**UCINTVM** Contains the number of times interval migration has run and successfully reached the desired target occupancy. This field does not apply to migration level 1 volumes.

## Tape Capacity Planning Record Field

This is the section for tape capacity planning information. If tape capacity planning records are requested, one record is created for each of the following types of DFSMShsm tapes:

- Migration level 2 tapes
- Incremental backup tapes
- Full volume dump tapes.

Both the MCDS and BCDS are needed to create these records. If backup availability is not enabled in the installation, the BCDS DD statement in the job must be specified with a DD DUMMY value.

| Name | Description |
|---|---|
| **UTSTYPE** | Identifies the type of DFSMShsm tapes summarized in this record (backup, dump, or migration level 2). |
| **UTFULL** | Contains the number of tapes that are marked full. |
| **UTPART** | Contains the number of tapes that are not marked full, but do contain data. |
| **UTEMPTY** | Contains the number of tapes that are empty. |

# Appendix H. Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen-readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

## Using assistive technologies

Assistive technology products, such as screen-readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using it to access z/OS interfaces.

## Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Volume I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

© Copyright IBM Corp. 1973, 2002

# Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

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

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:**
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

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

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

© Copyright IBM Corp. 1973, 2002

IBM Corporation
Information Enabling Requests
Dept. DZWA
5600 Cottle Road
San Jose, CA 95193 U.S.A.

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

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

## Programming Interface Information

This publication is intended to help you use DFSMS.

This publication also documents General-use Programming Interface and Associated Guidance Information provided by DFSMS.

General-use programming interfaces allow the customer to write programs that obtain the services of DFSMS.

General-use Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a topic or section or by the following marking:

General-use Programming Interface and Associated Guidance Information...

## Trademarks

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

| | |
|---|---|
| AIX | IBM |
| DB2 | MVS |
| CICS | MVS/ESA |
| DFSMSdfp | OS/390 |
| DFSMShsm | RACF |
| DFSMS/MVS | RAMAC |
| DFSORT | SP |
| Enterprise System Architecture/390 | System/390 |
| | z/OS |

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

# Glossary

This glossary defines technical terms and abbreviations used in DFSMS documentation. If you do not find the term you are looking for, refer to the index of the appropriate DFSMS manual or view the *IBM Dictionary of Computing Terms* located at:

http://www.ibm.com/networking/nsg/nsgmain.htm

This glossary includes terms and definitions from:

- The *American National Standard Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The *Information Technology Vocabulary* developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published part of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.
- The *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994.

The following cross-reference is used in this glossary:

**See:** This refers the reader to (a) a related term, (b) a term that is the expanded form of an abbreviation or acronym, or (c) a synonym or more preferred term.

## A

**ACB.** See *access method control block*

**access method control block (ACB).** A control block that links an application program to VSAM or VTAM programs.

**access method services.** A multifunction service program that manages VSAM and nonVSAM data sets, as well as integrated catalog facility (ICF). Access method services provides the following functions:

- defines and allocates space for data sets and catalogs
- converts indexed-sequential data sets to key-sequenced data sets
- modifies data set attributes in the catalog
- reorganizes data sets
- facilitates data portability among operating systems
- creates backup copies of data sets
- assists in making inaccessible data sets accessible
- lists the records of data sets and catalogs
- defines and builds alternate indexes
- converts CVOLS and VSAM catalogs to ICF catalogs

**ACDS.** See*active control data set*

**ACS routine.** See *automatic class seletion routine*

**active control data set (ACDS).** A VSAM linear data set that contains an SCDS that has been activated to control the storage management policy for the installation. When activating an SCDS, you determine which ACDS will hold the active configuration (if you have defined more than one ACDS). The ACDS is shared by each system that is using the same SMS configuration to manage storage. See also *source control data set, communications data set*.

**active data.** (1) (1) Data that can be accessed without any special action by the user, such as data on primary storage or migrated data. Active data also can be stored on tape volumes. (2) (2) For tape mount management, application data that is frequently referenced, small in size, and managed better on DASD than on tape. (3) Contrast with *inactive data*.

**ADSP.** See *automatic data set protection*

**aggregate group.** A collection of related data sets and control information that have been pooled to meet a defined backup or recovery strategy.

**AIX.** See *alternate index*

**alias.** An alternative name for an entry or for a member of a partitioned data set (PDS).

**alias entry.** An entry that relates an alias to the real entryname of a user catalog or non-VSAM data set.

**allocation.** Generically, the entire process of obtaining a volume and unit of external storage, and setting aside space on that storage for a data set.

**alternate index (AIX).** In systems with VSAM, a key-sequenced data set containing index entries

© Copyright IBM Corp. 1973, 2002

organized by the alternate keys of its associated base data records. It provides an alternate means of locating records in the data component of a cluster on which the alternate index is based.

**alternate index cluster.** The data and index components of an alternate index.

**alternate-index entry.** In VSAM, a catalog entry that contains information about an alternate index. An alternate index entry points to a data entry and an index entry to describe the alternate index's components, and to a cluster entry to identify the alternate index's base cluster. See also *cluster entry*.

**alternate-index record.** In VSAM, a collection of items used to sequence and locate one or more data records in a base cluster. Each alternate-index record contains an alternate-key value and one or more pointers. When the alternate index supports a key-sequenced data set, the pointer is the prime key value of each data record. When the alternate index supports an entry-sequenced data set (ESDS), the RBA value of the data records is the pointer. See also *alternate index entry*, *alternate key*, *base cluster*, and *key*.

**alternate key.** One or more characters within a data record used to identify the data record or control its use. Unlike the prime key, the alternate key can identify more than one data record. It is used to build an alternate index or to locate one or more base data records with an alternate index. See also *key* and *prime key*.

**application.** The use to which an access method is put or the end result that it serves; contrasted to the internal operation of the access method.

**automated tape library data server.** A device consisting of robotic components, cartridge storage areas, tape subsystems, and controlling hardware and software, together with the set of tape volumes that reside in the library and can be mounted on the library tape drives. Contrast with *manual tape library*. See also *tape library*.

**automatic backup.** (1) (1) In DFSMShsm, the process of automatically copying data sets from primary storage volumes or migration volumes to backup volumes. (2) (2) In OAM, the process of automatically copying a primary copy of an object from DASD, optical, or tape volume to a backup volume contained in an object backup storage group.

**automatic class selection (ACS) routine.** A procedural set of ACS language statements. Based on a set of input variables, the ACS language statements generate the name of a predefined SMS class, or a list of names of predefined storage groups, for a data set.

**automatic data set protection (ADSP).** In z/OS, a user attribute that causes all permanent data sets created by the user to be automatically defined to RACF with a discrete RACF profile.

**automatic secondary space management.** In DFSMShsm, the process of automatically deleting expired migrated data sets, deleting expired records from the migration control data sets, and migrating eligible data sets from migration level 1 volumes to migration level 2 volumes.

**availability.** For a storage subsystem, the degree to which a data set or object can be accessed when requested by a user.

# B

**backup control data set (BCDS).** In DFSMShsm, a VSAM key-sequenced data set that contains information about backup versions of data sets, backup volumes, dump volumes, and volumes under control of the backup and dump functions of DFSMShsm.

**backup-while-open (BWO).** This makes a backup copy of a data set while the data set is open for update. The backup copy can contain partial updates.

**base cluster.** A key-sequenced data set or entry-sequenced data set over which one or more alternate indexes are built. See also *alternate index entry* and *path*.

**base configuration.** The part of an SMS configuration that contains general storage management attributes, such as the default management class, default unit, and default device geometry. It also identifies the systems or system groups that an SMS configuration manages.

**basic catalog structure (BCS).** The name of the catalog structure in the integrated catalog facility environment. See also *integrated catalog facility catalog*.

**BCDS.** See *backup control data set*

**BCS.** See *basic catalog structure*

**buffer.** A routine or storage used to compensate for a difference in rate of flow of data, or time of occurrence of events, when transferring data from one device to another.

**BWO.** See *backup-while-open*

# C

**CA.** See *control area*

**cache set.** A parameter on storage class and defined in the base configuration information that maps a logical name to a set of CF cache structure names.

**cache set.** A parameter on storage class and defined in the base configuration information that maps a logical name to a set of CF cache structure names.

**capacity planning.** The process of forecasting and calculating the appropriate amount of physical computing resources required to accommodate an expected workload.

**Cartridge System Tape.** The base tape cartridge media used with 3480 or 3490 Magnetic Tape Subsystems. Contrast with *Enhanced Capacity Cartridge System Tape*.

**catalog.** A data set that contains extensive information required to locate other data sets, to allocate and deallocate storage space, to verify the access authority of a program or operator, and to accumulate data set usage statistics. See *master catalog* and *user catalog*.

**catalog cleanup.** A process that allows the deletion of entries if their volume is no longer available; catalog cleanup also allows deletion of a catalog even though it is not empty. Catalog cleanup is a function of the DELETE command.

**catalog connector.** A catalog entry, called either a user catalog entry or a catalog connector entry, in the master catalog that points to a user catalog's volume (that is, it contains the volume serial number of the direct access volume that contains the user catalog).

**catalog recovery area (CRA).** An entry-sequenced data set that exists on each volume owned by a recoverable catalog, including the volume on which the catalog resides. The CRA contains copies of the catalog's records and can be used to recover a damaged catalog.

**CCSID.** See *coded character set identifier*

**cell.** (1) An occurrence of information such as passwords, volume information, or associations. (2) A single cartridge storage location within a 3495.

**CF.** See *coupling facility*

**CI.** See *control interval*

**class (SMS).** See *SMS class*.

**class transition.** An event that brings about change to an object's service-level criteria, causing OAM to invoke ACS routines to assign a new storage class or management class to the object.

**cluster.** In VSAM, a named structure consisting of a group of related components. For example, when the data is key-sequenced, the cluster contains both the data and the index components; for data that is entry-sequenced, the cluster contains only a data component.

**cluster entry.** A catalog entry that contains information about a key-sequenced or entry-sequenced VSAM cluster: ownership, cluster attributes, and the cluster's passwords and protection attributes. A key-sequenced cluster entry points to both a data entry and an index entry. An entry-sequenced cluster entry points to a data entry only.

**Coded Character Set Identifier (CCSID).** A 16-bit number that identifies a specific encoding scheme identifier, character set identifiers, code page identifiers, and additional coding required information. The CCSID uniquely identifies the coded graphic character representation used.

**communications data set (COMMDS).** The primary means of communication among systems governed by a single SMS configuration. The COMMDS is a VSAM linear data set that contains the name of the ACDS and current utilization statistics for each system-managed volume, which helps balance space among systems running SMS. See also *active control data set* and *source control data set*.

**compaction.** See *improved data recording capability.*

**compatibility mode.** The mode of running SMS in which no more than eight names—representing systems, system groups, or both—are supported in the SMS configuration.

**component.** (1) A named, cataloged collection of stored records. A component, the lowest member of the hierarchy of data structures that can be cataloged, contains no named subsets. (2) In this book, the components of an object are usually referred to as the object's data component and index component. Also, the cluster, data, or index fields of a subrecord.

**compress.** (1) (1) To reduce the amount of storage required for a given data set by having the system replace identical words or phrases with a shorter token associated with the word or phrase. (2) (2) To reclaim the unused and unavailable space in a partitioned data set that results from deleting or modifying members by moving all unused space to the end of the data set.

**compressed format.** A particular type of extended-format data set specified with the (COMPACTION) parameter of data class. VSAM can compress individual records in a compressed-format data set. SAM can compress individual blocks in a compressed-format data set. See *compress*.

**concurrent copy.** A function to increase the accessibility of data by enabling you to make a consistent backup or copy of data concurrent with the usual application program processing.

**construct.** One of the following: data class, storage class, management class, storage group, aggregate group, base configuration.

**control area (CA).** A group of control intervals used as a unit for formatting a data set before adding records to it. Also, in a key-sequenced data set, the set of control intervals pointed to by a sequence-set index record;

used by VSAM for distributing free space and for placing a sequence-set index record adjacent to its data.

**control interval (CI).** A fixed-length area of auxiliary storage space in which VSAM stores records. It is the unit of information (an integer multiple of block size) transmitted to or from auxiliary storage by VSAM.

**CRA.** See *catalog recovery areat*

**coupling facility (CF).** The hardware that provides high-speed caching, list processing, and locking functions in a Parallel Sysplex.

# D

**DASD volume.** A DASD space identified by a common label and accessed by a set of related addresses. See also *volume primary storage*, *migration level 1, migration level 2*.

**data class.** A collection of allocation and space attributes, defined by the storage administrator, that are used to create a data set.

**data component.** The part of a VSAM data set, alternate index, or catalog that contains the object's data records.

**data entry.** A catalog entry that describes the data component of a cluster, alternate index, page spaces, or catalog. A data entry contains the data component's attributes, allocation and extent information, and statistics. A data entry for a cluster's or catalog's data component can also contain the data component's passwords and protection attributes.

**Data Facility Sort.** An IBM licensed program that is a high-speed data processing utility. DFSORT provides an efficient and flexible way to handle sorting, merging, and copying operations, as well as providing versatile data manipulation at the record, field, and bit level.

**data set.** In DFSMS, the major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access. In non-z/OS UNIX System Services/MVS environments, the terms *data set* and *file* are generally equivalent and sometimes are used interchangeably. See also *file*.

**data set control block (DSCB).** A control block in the VTOC that describes data set characteristics.

**DBCS.** See *double-byte character set*

**default device geometry.** Part of the SMS base configuration, it identifies the number of bytes per track and the number of tracks per cylinder for converting space requests made in tracks or cylinders into bytes, when no unit name has been specified.

**default management class.** Part of the SMS base configuration, it identifies the management class that should be used for system-managed data sets that do not have a management class assigned.

**default unit.** Part of the SMS base configuration, it identifies an esoteric (such as SYSDA) or generic (such as 3390) device name. If a user omits the UNIT parameter on the JCL or the dynamic allocation equivalent, SMS applies the default unit if the data set has a disposition of MOD or NEW and is *not* system-managed.

**DFSMS environment.** An environment that helps automate and centralize the management of storage. This is achieved through a combination of hardware, software, and policies. In the DFSMS environment for MVS, this function is provided by DFSMS, DFSORT, and RACF. See also *system-managed storage*.

**DFSMSdfp.** A DFSMS functional component or base element of z/OS, that provides functions for storage management, data management, program management, device management, and distributed data access.

**DFSMShsm.** A DFSMS functional component or base element of z/OS, used for backing up and recovering data, and managing space on volumes in the storage hierarchy.

**DFSMShsm-managed volume.** (1) (1) A primary storage volume, which is defined to DFSMShsm but which does not belong to a storage group. (2) (2) A volume in a storage group, which is using DFSMShsm automatic dump, migration, or backup services. Contrast with *system-managed volume, DFSMSrmm-managed volume*.

**DFSMSrmm-managed volume.** A tape volume that is defined to DFSMSrmm. Contrast with *system-managed volume, DFSMShsm-managed volume*.

**DFSORT.** See *data facility sort*

**dictionary.** A table that associates words, phrases, or data patterns to shorter tokens. The tokens replace the associated words, phrases, or data patterns when a data set is compressed.

**direct access.** The retrieval or storage of data by a reference to its location in a data set rather than relative to the previously retrieved or stored data. See also *addressed direct access* and *keyed direct access*.

**double-byte character set (DBCS).** A 2-byte hexadecimal value which can be used to represent a single character for languages that contain too many characters or symbols for each to be assigned a 1-byte hexadecimal value.

**DSCB.** See *data set control block*

**dual copy.** A high availability function made possible by nonvolatile storage in some models of the IBM 3990 Storage Control. Dual copy maintains two functionally identical copies of designated DASD volumes in the logical 3990 subsystem, and automatically updates both copies every time a write operation is issued to the dual copy logical volume.

**dummy storage group.** A type of storage group that contains the serial numbers of volumes no longer connected to a system. Dummy storage groups allow existing JCL to function without having to be changed. See also *storage group*.

**dynamic allocation.** The allocation of a data set or volume by the use of the data set name or volume serial number rather than by the use of information contained in a JCL statement.

# E

**Enhanced Capacity Cartridge System Tape.** Cartridge system tape with increased capacity that can only be used with 3490E tape subsystems. Contrast with *Cartridge System Tape.*

**entry.** A collection of information about a cataloged object in a master or user catalog. Each entry resides in one or more 512-byte records.

**entry name.** A unique name for each component or object as it is identified in a catalog. The entryname is the same as the dsname in a DD statement that describes the object.

**entry sequence.** The order in which data records are physically arranged (according to ascending RBA) in auxiliary storage, without respect to their contents. Contrast to *key sequence*.

**entry-sequenced data set (ESDS).** In VSAM, a data set whose records are loaded without respect to their contents, and whose RBAs cannot change. Records are retrieved and stored by addressed access, and new records are added at the end of the data set.

**ESDS.** See *entry-sequenced data set*

**esoteric unit name.** A name used to define a group of devices having similar hardware characteristics, such as TAPE or SYSDA. Contrast with *generic unit name*.

**expiration.** (1) The process by which data sets or objects are identified for deletion because their expiration date or retention period has passed. On DASD, data sets and objects are deleted. On tape, when all data sets have reached their expiration date, the tape volume is available for reuse. (2) In DFSMSrmm, all volumes have an expiration date or retention period set for them either by vital record specification policy, by user-specified JCL when writing a data set to the volume, or by an installation default. When a volume reaches its expiration date or retention period, it becomes eligible for release.

**export.** To create a backup or portable copy of a VSAM cluster, alternate index, or integrated catalog facility user catalog.

**extended addressability.** The ability to create and access a VSAM data set that is greater than 4 GB in size. Extended addressability data sets must be allocated with DSNTYPE=EXT and EXTENDED ADDRESSABILITY=Y.

**extended format.** The format of a data set that has a data set name type (DSNTYPE) of EXTENDED. The data set is structured logically the same as a data set that is not in extended format but the physical format is different. Data sets in extended format can be striped or compressed. Data in an extended format VSAM KSDS can be compressed. See also *striped data set, compressed format*.

**extent.** A continuous space allocated on a DASD volume occupied by a data set or portion of a data set. An extent of a data set contains a whole number of control areas.

# F

**field.** In a record or a control block, a specified area used for a particular category of data or control information.

**filtering.** The process of selecting data sets based on specified criteria. These criteria consist of fully or partially-qualified data set names or of certain data set characteristics.

# G

**GDG.** See *generation data group*

**GDS.** See *generation data set*

**generic unit name.** A name assigned to a class of devices with the same geometry (such as 3390). Contrast with *esoteric unit name*.

**global resource serialization (GRS).** A component of z/OS used for serializing use of system resources and for converting hardware reserves on DASD volumes to data set enqueues.

**GRS.** See *global resource serialization*

# H

**hardware configuration definition (HCD).** An interactive interface in MVS that enables an installation to define hardware configurations from a single point of control.

**hardware configuration definition (HCD).** An interactive interface in z/OS that enables an installation to define hardware configurations from a single point of control.

# I

**IDRC.** See *improved data recording capability*

**improved data recording capability (IDRC).** A recording mode that can increase the effective cartridge data capacity and the effective data rate when enabled and invoked. IDRC is always enabled on the 3490E tape subsystem.

**inactive data.** (1) (1) A copy of active data, such as vital records or a backup copy of a data set. Inactive data is never changed, but can be deleted or superseded by another copy. (2) (2) In tape mount management, data that is written once and never used again. The majority of this data is point-in-time backups. (3) (3) Objects infrequently accessed by users and eligible to be moved to the optical library or shelf. (4) Contrast with *active data*.

**indexed VTOC.** A volume table of contents with an index that contains a list of data set names and free space information, which allows data sets to be located more efficiently.

**index entry.** A catalog entry that describes the index component of a key-sequenced cluster, alternate index, or catalog. An index entry contains the index component's attributes, passwords and protection attributes, allocation and extent information, and statistics.

**index level.** A set of index records that order and give the location of records in the next lower level or of control intervals in the data set that it controls.

**index record.** A collection of index entries that are retrieved and stored as a group. Contrast with *data record*.

**index set.** The set of index levels above the sequence set. The index is comprised of the index set and the sequence set.

**integrated catalog facility.** The name of the catalog in DFSMSdfp that is a functional replacement for OS CVOLs and VSAM catalogs.

**integrated catalog facility catalog.** A catalog that is composed of a basic catalog structure (BCS) and its related volume tables of contents (VTOCs) and VSAM volume data sets (VVDSs). See also *basic catalog structure, VSAM volume data set*.

**integrity.** See *data integrity*.

**Interactive Storage Management Facility (ISMF).** The interactive interface of DFSMS that allows users and storage administrators access to the storage management functions.

**interval migration.** In DFSMShsm, automatic migration that occurs when a threshold level of occupancy is reached or exceeded on a DFSMShsm-managed volume, during a specified time interval. Data sets are moved from the volume, largest eligible data set first, until the low threshold of occupancy is reached.

**ISMF.** See *interactive storage manage facility*

# K

**key.** One or more characters within an item of data that are used to identify it or control its use. As used in this publication, one or more consecutive characters taken from a data record, used to identify the record and establish its order with respect to other records. See also *prime key*.

**key sequence.** The collating sequence of data records, determined by the value of the key field in each of the data records. It can be the same as, or different from, the entry sequence of the records.

**key-sequenced data set (KSDS).** A VSAM data set whose records are loaded in key sequence and controlled by an index. Records are retrieved and stored by keyed access or by addressed access, and new records are inserted in the data set in key sequence because of free space allocated in the data set. Relative byte addresses of records can change because of control interval or control area splits.

**KSDS.** See *key-sequence data set*

# L

**LDS.** See *linear data set*

**level 0 volume.** A primary volume or a user volume not managed by DFSMShsm.

**level 1 volume.** A volume owned by DFSMShsm containing data sets that migrated from a level 0 volume.

**level 2 volume.** A volume under control of DFSMShsm containing data sets that migrated from a level 1 volume, or from a volume not managed by DFSMShsm.

**library.** A partitioned data set containing a related collection of named members.

**linear data set (LDS).** A VSAM data set that contains data but no control information. A linear data set can be accessed as a byte-addressable string in virtual storage.

**LRECL.** *Logical record length*

# M

**management class.** A collection of management attributes, defined by the storage administrator, used to control the release of allocated but unused space; to control the retention, migration, and back up of data sets; to control the retention and back up of aggregate groups, and to control the retention, back up, and class transition of objects.

**master catalog.** A catalog that contains extensive data set and volume information that VSAM requires to locate data sets, to allocate and deallocate storage space, to verify the authorization of a program or operator to gain access to a data set, and accumulate usage statistics for data sets.

**MCDS.** See *Migration control data set*

**MEDIA2.** Enhanced Capacity Cartridge System Tape

**MEDIA3.** High Performance Cartridge Tape

**MEDIA4.** Extended High Performance Cartridge Tape

**migration.** The process of moving unused data to lower cost storage in order to make space for high-availability data. If you wish to use the data set, it must be recalled. See also *migration level 1, migration level 2*.

**migration control data set (MCDS).** In DFSMShsm, a VSAM key-sequenced data set that contains statistics records, control records, user records, records for data sets that have migrated, and records for volumes under migration control of DFSMShsm.

**migration level 1.** DFSMShsm-owned DASD volumes that contain data sets migrated from primary storage volumes. The data can be compressed. See also *storage hierarchy*. Contrast with *primary storage, migration level 2*.

**migration level 2.** DFSMShsm-owned tape or DASD volumes that contain data sets migrated from primary storage volumes or from migration level 1 volumes. The data can be compressed. See also *storage hierarchy*. Contrast with *primary storage, migration level 1*.

**MLA.** See *Multilevel alias facility*

**mount.** A host-initiated operation that results in a tape cartridge being physically inserted into a tape drive by the 3495 robot. The drive access window is also closed by the robot.

**multilevel alias facility.** A function in catalog address space that allows integrated catalog facility catalog selection based on one to four data set name qualifiers.

**MVS configuration program (MVSCP).** A single-step, batch program that defines the input/output configuration to z/OS.

**MVS/ESA.** Multiple Virtual Storage/Enterprise Systems Architecture. A z/OS operating system environment that supports ESA/390.

**MVS/ESA.** Multiple Virtual Storage/Enterprise Systems Architecture. A z/OS operating system environment that supports ESA/390.

# N

**non-SMS volume.** A volume that is not controlled by SMS.

**non-VSAM entry.** A catalog entry that describes a non-VSAM data set. A non-VSAM entry contains the data set's volume serial number and device type. If the data set resides on a magnetic tape volume, the entry can also identify the data set's file number. When the data set resides on a direct access device, the operating system obtains further information by examining the data set's DSCB (data set control block) in the volume's VTOC (volume table of contents).

**non-VSAM volume record (NVR).** A VVDS record which contains SMS-related information about a non-VSAM, system-managed data set.

**NVR.** See *non-VSAM volume record*

# O

**OAM managed volumes.** Optical or tape volumes controlled by the object access method (OAM)

**object.** A named byte stream having no specific format or record orientation.

**object access method (OAM).** An access method that provides storage, retrieval, and storage hierarchy management for objects and provides storage and retrieval management for tape volumes contained in system-managed libraries.

**object backup storage group.** A type of storage group that contains optical or tape volumes used for backup copies of objects. See also *storage group*.

**object directory tables.** A collection of DB2 tables that contain information about the objects that have been stored in an object storage group.

**object storage group.** A type of storage group that contains objects on DASD, tape, or optical volumes. See also *storage group*.

**object storage hierarchy.** A hierarchy consisting of objects stored in DB2 table spaces on DASD, on optical

or tape volumes that reside in a library, and on optical or tape volumes that reside on a shelf. See also *storage hierarchy.*

**OpenEdition MVS.** See *z/OS UNIX System Services.*

**optical disk drive.** The mechanism used to seek, read, and write data on an optical disk. An optical disk drive can be operator-accessible or library-resident.

**optical library.** A storage device that houses optical drives and optical cartridges, and contains a mechanism for moving optical disks between a cartridge storage area and optical disk drives.

**optical volume.** Storage space on an optical disk, identified by a volume label. See also *volume*.

# P

**page space.** A system data set that contains pages of virtual storage. The pages are stored into and retrieved from the page space by the auxiliary storage manager.

**partitioned data set (PDS).** A data set on direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

**partitioned data set extended (PDSE).** A system-managed data set that contains an indexed directory and members that are similar to the directory and members of partitioned data sets. A PDSE can be used instead of a partitioned data set.

**password.** A unique string of characters stored in a catalog that a program, a computer operator, or a terminal user must supply to meet security requirements before the program gains access to a data set.

**path.** A named, logical entity composed of one or more clusters (an alternate index and its base cluster, for example).

**path entry.** A catalog entry that contains information about a path, and that points to the path's related objects.

**PDS.** See *partitioned data set*

**PDSE.** See *partitioned data set extended*

**performance.** (1) (1) A measurement of the amount of work a product can produce with a given amount of resources. (2) (2) In a system-managed storage environment, a measurement of effective data processing speed with respect to objectives set by the storage administrator. Performance is largely determined by throughput, response time, and system availability.

**permanent data set.** A user-named data set that is normally retained for longer than the duration of a job or interactive session. Contrast with *temporary data set*.

**plaintext.** A data set or key which is not enciphered (with the cryptographic option). A data set or key is plaintext before it is enciphered and after it is deciphered.

**pool storage group.** A type of storage group that contains system-managed DASD volumes. Pool storage groups allow groups of volumes to be managed as a single entity. See also *storage group*.

**portability.** The ability to use VSAM data sets with different operating systems. Volumes whose data sets are cataloged in a user catalog can be removed from storage devices of one system, moved to another system, and mounted on storage devices of that system. Individual data sets can be transported between operating systems using access method services.

**portable data set.** A data set that can be transported between systems using access method services.

**primary space allocation.** Amount of space requested by a user for a data set when it is created. Contrast with *secondary space allocation*.

**primary storage.** A DASD volume available to users for data allocation. The volumes in primary storage are called primary volumes. See also *storage hierarchy*. Contrast with *migration level 1, migration level 2*.

**prime index.** The index component of a key-sequenced data set.

**prime key.** One or more characters within a data record used to identify the data record or control its use. A prime key must be unique.

**private.** The state of a tape volume which contains user-written data. A private volume is requested by specifying the volume serial number.

**private volume.** A tape volume which has been assigned the private use attribute by the software. If the cartridge resides in a 3495, it is assigned to the private category.

# Q

**qualified name.** A data set name consisting of a string of names segmented by periods; for example TREE.FRUIT.APPLE is a qualified name.

# R

**RACF authorization.** (1) The facility for checking a user's level of access to a resource against the user's desired access. (2) The result of that check.

**RBA.** See *relative byte address*

**record.** A set of data treated as a unit. See *index record*, *data record*, *stored record*.

**RLS.** *Record level sharing*

**recovery.** The process of rebuilding data after it has been damaged or destroyed, often by using a backup copy of the data or by reapplying transactions recorded in a log.

**recovery volume.** The first volume of a prime index if the VSAM data set is a key-sequenced cluster; the first volume of the data set if entry-sequenced.

**relative byte address (RBA).** The displacement of a data record or a control interval from the beginning of the data set to which it belongs; independent of the manner in which the data set is stored.

**relative record data set (RRDS).** A VSAM data set whose records are loaded into fixed-length, or variable-length slots.

**Resource Access Control Facility (RACF).** An IBM licensed program that is included in z/OS Security Server and is also available as a separate program for the z/OS and VM environments. RACF provides access control by identifying and verifying the users to the system, authorizing access to protected resources, logging detected unauthorized attempts to enter the system, and logging detected accesses to protected resources.

**retained lock.** A lock protecting transaction updates when a problem delays transaction recovery of the updates. The retained status is cleared when transaction recovery completes.

**PDS.** See *relative record data set*

# S

**SCDS.** See *source control data sett*

**SDSP.** See *small data set packing*

**scratch.** The state of a tape volume which is available for general use. A scratch volume is requested by omitting the volume serial number.

**scratch tape.** See *scratch volume.*

**scratch volume.** A tape volume which has been assigned the scratch use attribute by the software. If the cartridge resides in a 3495, it is assigned to the scratch category.

**secondary space allocation.** Amount of additional space requested by the user for a data set when primary space is full. Contrast with *primary space allocation*.

**security.** See *data security*.

**sequence set.** The lowest level of the index of a key-sequenced data set; it gives the locations of the control intervals in the data set and orders them by the key sequence of the data records they contain. The sequence set and the index set together comprise the index.

**shelf.** A place for storing removable media, such as tape and optical volumes, when they are not being written to or read.

**shelf location.** (1) A single space on a shelf for storage of removable media. (2) In DFSMSrmm, a shelf location is defined in the removable media library by a rack number, and in a storage location, it is defined by a bin number. See also *rack number, bin number*.

**slot.** A single cartridge storage location within a 3495. See also *cell.*

**small-data-set packing (SDSP).** In DFSMShsm, the process used to migrate data sets that contain equal to or less than a specified amount of actual data. The data sets are written as one or more records into a VSAM data set on a migration level 1 volume.

**small-data-set-packing data set.** In DFSMShsm, a VSAM key-sequenced data set allocated on a migration level 1 volume and containing small data sets that have been migrated.

**PDS.** See *storage management subsystem*

**SMS class.** A list of attributes that SMS applies to data sets having similar allocation (data class), performance (storage class), or backup and retention (management class) needs.

**SMS configuration.** A configuration base, Storage Management Subsystem class, group, library, and drive definitions, and ACS routines that the Storage Management Subsystem uses to manage storage. See also *configuration, base configuration, source control data set* .

**source control data set (SCDS).** A VSAM linear data set containing an SMS configuration. The SMS configuration in an SCDS can be changed and validated using ISMF. See also *active control data set, communications data set*.

**spanned record.** A logical record whose length exceeds control interval length, and as a result crosses (or spans) one or more control interval boundaries within a single control area.

**sphere record.** A collection of logically related subrecords in one VSAM logical record.

**storage administrator.** A person in the data processing center who is responsible for defining, implementing, and maintaining storage management policies.

**storage class.** A collection of storage attributes that identify performance goals and availability requirements, defined by the storage administrator, used to select a device that can meet those goals and requirements.

**storage control.** The component in a storage subsystem that handles interaction between processor channel and storage devices, runs channel commands, and controls storage devices.

**storage group.** A collection of storage volumes and attributes, defined by the storage administrator. The collections can be a group of DASD volumes or tape volumes, or a group of DASD, optical, or tape volumes treated as a single object storage hierarchy. See also *VIO storage group, pool storage group, tape storage group, object storage group, object backup storage group, dummy storage group*.

**storage hierarchy.** An arrangement of storage devices with different speeds and capacities. The levels of the storage hierarchy include main storage (memory, DASD cache), primary storage (DASD containing uncompressed data), migration level 1 (DASD containing data in a space-saving format), and migration level 2 (tape cartridges containing data in a space-saving format). See also *primary storage, migration level 1, migration level 2, object storage hierarchy*.

**storage location.** A location physically separate from the removable media library where volumes are stored for disaster recovery, backup, and vital records management.

**storage management.** The activities of data set allocation, placement, monitoring, migration, backup, recall, recovery, and deletion. These can be done either manually or by using automated processes. The Storage Management Subsystem automates these processes for you, while optimizing storage resources. See also *Storage Management Subsystem*.

**Storage Management Subsystem (SMS).** A DFSMS facility used to automate and centralize the management of storage. Using SMS, a storage administrator describes data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements to the system through data class, storage class, management class, storage group, and ACS routine definitions.

**storage subsystem.** A storage control and its attached storage devices. See also *tape subsystem*.

**stored record.** A data record, together with its control information, as stored in auxiliary storage.

**stripe.** In DFSMS, the portion of a striped data set, such as an extended format data set, that resides on one volume. The records in that portion are not always logically consecutive. The system distributes records among the stripes such that the volumes can be read from or written to simultaneously to gain better performance. Whether it is striped is not apparent to the application program.

**striped data set.** In DFSMS, an extended-format data set consisting of two or more stripes. SMS determines the number of stripes to use based on the value of the SUSTAINED DATA RATE in the storage class. Striped data sets can take advantage of the sequential data striping access technique. See*stripe, striping*.

**striping.** A software implementation of a disk array that distributes a data set across multiple volumes to improve performance.

**subrecord.** The user definition level of a sphere, such as an alternate index, cluster, or generation data set.

**system data.** The data sets required by z/OS or its subsystems for initialization and control.

**system group.** All systems that are part of the same Parallel Sysplex and are running the Storage Management Subsystem with the same configuration, minus any systems in the Parallel Sysplex that are explicitly defined in the SMS configuration.

**system-managed data set.** A data set that has been assigned a storage class.

**system-managed storage.** Storage managed by the Storage Management Subsystem. SMS attempts to deliver required services for availability, performance, and space to applications. See also *system-managed storage environment*.

**system-managed tape library.** A collection of tape volumes and tape devices, defined in the tape configuration database. A system-managed tape library can be automated or manual. See also *tape library.*

**system-managed volume.** A DASD, optical, or tape volume that belongs to a storage group. Contrast with *DFSMShsm-managed volume, DFSMSrmm-managed volume*.

# T

**tape configuration database.** One or more volume catalogs used to maintain records of system-managed tape libraries and tape volumes.

**tape library.** A set of equipment and facilities that support an installation's tape environment. This can include tape storage racks, a set of tape drives, and a

set of related tape volumes mounted on those drives. See also *system-managed tape library, automated tape library data server*.

**Tape Library Dataserver.** A hardware device that maintains the tape inventory that is associated with a set of tape drives. An automated tape library dataserver also manages the mounting, removal, and storage of tapes.

**tape storage group.** A type of storage group that contains system-managed private tape volumes. The tape storage group definition specifies the system-managed tape libraries that can contain tape volumes. See also *storage group*.

**tape subsystem.** A magnetic tape subsystem consisting of a controller and devices, which allows for the storage of user data on tape cartridges. Examples of tape subsystems include the IBM 3490 and 3490E Magnetic Tape Subsystems.

**tape volume.** A tape volume is the recording space on a single tape cartridge or reel. See also *volume*.

**TCDB.** See *tape configuration data base*

**temporary data set.** An uncataloged data set whose name begins with & or &&, that is normally used only for the duration of a job or interactive session. Contrast with *permanent data set*.

**TSO.** See *time sharing option*

**threshold.** A storage group attribute that controls the space usage on DASD volumes, as a percentage of occupied tracks versus total tracks. The *low migration threshold* is used during primary space management and interval migration to determine when to stop processing data. The *high allocation threshold* is used to determine candidate volumes for new data set allocations. Volumes with occupancy lower than the high threshold are selected over volumes that meet or exceed the high threshold value.

# U

**use attribute.** (1) The attribute assigned to a DASD volume that controls when the volume can be used to allocate new data sets; use attributes are *public*, *private*, and *storage*. (2) For system-managed tape volumes, use attributes are *scratch* and *private*.

**user catalog.** An optional catalog used in the same way as the master catalog and pointed to by the master catalog. It lessens the contention for the master catalog and facilitates volume portability.

**user catalog connector.** See *catalog connector*.

**USVR.** *User-security-verification routine*

# V

**validate.** To check the completeness and consistency of an individual ACS routine or an entire SMS configuration.

**VIO.** *Virtual input/output*

**virtual input/output (VIO) storage group.** A type of storage group that allocates data sets to paging storage, which simulates a DASD volume. VIO storage groups do not contain any actual DASD volumes. See also *storage group*.

**volume.** The storage space on DASD, tape, or optical devices, which is identified by a volume label. See also *DASD volume, optical volume, tape volume*.

**volume status.** In the Storage Management Subsystem, indicates whether the volume is fully available for system management:

- “Initial” indicates that the volume is not ready for system management because it contains data sets that are ineligible for system management.
- “Converted” indicates that all of the data sets on a volume have an associated storage class and are cataloged in an integrated catalog facility catalog.
- “Non-system-managed” indicates that the volume does not contain any system-managed data sets and has not been initialized as system-managed.

**volume table of contents (VTOC).** A table on a direct access volume that describes each data set on the volume.

**VSAM.** *Virtual storage access method*

**VSAM catalog.** A special key-sequenced data set with an index containing extensive data set and volume information that VSAM requires to locate data sets or files, allocate and deallocate storage space, verify the authorization of a program or operator to gain access to a file, and accumulate usage statistics for data sets or files. VSAM catalogs have been functionally replaced by integrated catalog facility catalogs. VSAM catalogs are not supported by the Storage Management Subsystem.

**VSAM record-level sharing (VSAM RLS).** An extension to VSAM that provides direct record-level sharing of VSAM data sets from multiple address spaces across multiple systems. Record-level sharing uses the z/OS Coupling Facility to provide cross-system locking, local buffer invalidation, and cross-system data caching.

**VSAM sphere.** The base cluster of a VSAM data set and its associated alternate indexes.

**VSAM volume control record (VVCR).** The first logical record in the VVDS that contains information to manage DASD space and the BCS back pointers.

**VSAM volume data set (VVDS).** A data set that describes the characteristics of VSAM and system-managed data sets residing on a given DASD volume; part of an integrated catalog facility catalog. See also *basic catalog structure, integrated catalog facility catalog*.

**VSAM volume record (VVR).** A VSAM logical record within a VVDS.

**VTOC.** See *volume table of contents*

**VVCR.** See *VSAM volume control record*

**VVDS.** See *VSAM volume data set*

**VVR.** See *VSAM volume record*

# Index

## A


access method services 81
  attribute selection order 14
  coding
    command structure xxi
    continuing commands xxv
    parameters xxi
    subparameters xxii, xxv
  command structure xxi
  commands 25
  invoking
    described 4
    from a PL/I program 396
    from JCL 5
    from TSO 5
    from user program 393
  parameter
    keyword xxi
    positional xxi
accessibility 483
ACCODE parameter
  ALLOCATE command 32
ACCOUNT parameter
  DEFINE command
    CLUSTER 57, 145
ADDVOLUMES parameter
  ALTER command 57
ALC (allocation group)
  keyword fields, LISTCAT output 335
alias
  aliasname subparameter, coding xxiii
  entry keywords, LISTCAT output 328
  multilevel facility 9
    generic catalog selection 13
  non-VSAM data set 117
  user catalog 117
ALIAS
  requirement 257
ALIAS parameter
  DEFINE command 117
  DELETE command 208
  IMPORT command 251
  IMPORT CONNECT command 263
  LISTCAT command 268
ALL parameter
  LISTCAT command
    description 273
    output listings 356
Allocate Command
  PDSE 34
ALLOCATE command
  examples 47, 53
  functional command format 27
  parameters
    optional 32, 47
    required 32
  restrictions 28
ALLOCATE command *(continued)*
  return codes 30
  TSO naming convention 27
allocating SMS-managed data sets 28
allocation
  direct, using JCL 3, 4
  dynamic 2
ALLOCATION parameter
  LISTCAT command
    description 273
    output listings 368
alphanumeric characters xxv
ALTER
  important 77
ALTER command
  ACCOUNT
    optional parameters 57
  catalog search order 9
  entry types that can be altered 54, 56
  examples 73, 75
  format 53
  parameters
    optional 57, 72
    required 56
  RLS (record-level sharing) 58
ALTER LIBRARYENTRY command
  examples 80
  format 77
  parameters
    optional 77, 80
    required 77
ALTER VOLUMEENTRY command
  examples 85, 86
  format 81
  parameters 81
    optional 81, 85
    required 81
  required 81
alternate index
  BLDINDEX command 87
  data and index components 135
  defining 121, 122
  defining with RECATALOG 137
  deleting 208
  entry keywords, LISTCAT 328
  exporting 239
  path 187
  record size 129
  SMS-managed 135
  virtual storage space 90
alternate key 128
alternate target data set
  specifying for listing 4
ALTERNATEINDEX parameter
  DEFINE command 122
  DELETE command 208
  LISTCAT command 268


© Copyright IBM Corp. 1973, 2002

ALTFILE parameter
  ALLOCATE command 33
AMSDUMP DD statement 3
APF 393
APPEND parameter
  DCOLLECT command 110
AREAS parameter
  PARM command 20
argument lists
  invoking macros 393
ASN (associations group)
  keyword fields, LISTCAT output 336
ATT (attributes group)
  keyword fields, LISTCAT output 337
ATTACH macro 393
attribute
  nullifying protection 66
  specifying selection order 14
AUTHORIZATION parameter
  ALTER command 66
authorized program facility (APF) 393
auxiliary list 394
auxiliary storage management
  defining page spaces 185
AVBLOCK parameter
  ALLOCATE command 45
AVGREC parameter
  ALLOCATE command 33

## B

backup-while-open 58
BCS (basic catalog structure)
  diagnose examples 230, 234
BFALN parameter
  ALLOCATE command 33
BFTEK parameter
  ALLOCATE command 33
binary form subparameters, coding xxiii
BINDDATA command 403
BLDINDEX
  requirement 87
BLDINDEX command
  catalog selection order 10
  examples 93, 94
  format 87
  parameters
    optional 88, 90
    required 87, 88
BLKSIZE parameter
  ALLOCATE command 33
BLOCK parameter
  ALLOCATE command 45
buffer space
  altering 57
  data 194
  index 194
BUFFERSPACE parameter
  ALTER command 57
  DEFINE command
    ALTERNATEINDEX 125
BUFFERSPACE parameter *(continued)*
  DEFINE command *(continued)*
    CLUSTER 145
    USERCATALOG 194
BUFL parameter
  ALLOCATE command 35
BUFND parameter
  ALTER command 58
    RLS (record-level sharing) 58
  DEFINE command
    USERCATALOG 194
BUFNI parameter
  ALTER command 58
  DEFINE command
    USERCATALOG 194
BUFNO parameter
  ALLOCATE command 35
BUFOFF parameter
  ALLOCATE command 35
building, alternate index 87
BWO (backup-while-open)
  ALTER command 58, 66
  DEFINE command
    CLUSTER 146
  LISTCAT command 329

## C

cache device 403
CALL macro 393, 394
CANCEL command
  description 19
  example 19
CAPPLANDDATA parameter
  DCOLLECT command 109
catalog
  alias, user 117
  connecting to master 263
  copying 303
  defining, user 191
  entry 267
    altering 274
    deleting 208
    field names 327
    listing, example 275
    non-VSAM data set 173
    recreating info from VVDS 153
  generic selection 13
  JOBCAT DD statement 9
  listing 267
  RACF security authorization 321
  search order
    ALTER 9
    DELETE 10
    LISTCAT 12
  selection order
    BLDINDEX 10
    DEFINE 10
    EXPORT DISCONNECT 12
    specifying 9
  sharing 198

CATALOG parameter
ALLOCATE command 37
ALTER command 59
BLDINDEX command 88
DEFINE command
ALIAS 118
ALTERNATEINDEX 125
CLUSTER 146
GENERATIONDATAGROUP 169
NONVSAM 176
PAGESPACE 182
PATH 188
USERCATALOG 194
DELETE command 211
DIAGNOSE command 229, 230
EXPORT command 247
IMPORT command 252
IMPORT CONNECT command 264
LISTCAT command 269
catalog selection, generic 13
CCSID parameter
ALTER command 59
CFRESET command
SHCDS command
fall back 316
CFRESETDS command
SHCDS command
fall back 316
CHAIN parameter 20
CHARACTER parameter
PRINT command 278
CHECKPOINT parameter
ALTER VOLUMEENTRY command 81
CREATE VOLUMEENTRY command 101
CIMODE parameter
EXPORT command 240
relation to IMPORT command 240
CIPHERUNIT parameter
REPRO command 297
classes, SMS 28
cleanup, volume 67
cluster
altering attributes 73
altering entry names 73
components 160
data organization 150
defining 139
entry-sequenced 162
linear data set, example 168
relative record 163
specifying parameters 141
deleting 209
entry keywords, LISTCAT output 328
exporting 239
importing 257
listing catalog entry 268
migrating
DB2 to linear data set, example 74
path definition 187
printing contents 277
verifying end-of-file 319
CLUSTER parameter
DEFINE command 141
DELETE command 209
LISTCAT command 268
CODE parameter
ALTER command 66
DEFINE command
PAGESPACE 183
code subparameter, coding xxiii
coding
access method services commands xxi
characters xxv
characters, special xxiii
continuing commands xxii, xxv
keyword parameters xxi
modal commands, continuation errors 23
parameters xxv
positional parameters xxi
subparameters xxiii, xxv
terminator xxvi
collection of objects
defining catalog entry 173
COLLECTION parameter
DEFINE command
NONVSAM 176
command
ALLOCATE 30
ALTER 53
ALTER LIBRARYENTRY 77
ALTER VOLUMEENTRY 81
BLDINDEX 87
CREATE LIBRARYENTRY 95
CREATE VOLUMEENTRY 101
DCOLLECT 107
DEFINE
ALIAS 117
ALTERNATEINDEX 121
CLUSTER 139
GENERATIONDATAGROUP 169
MASTERCATALOG 191
NONVSAM 173
PAGESPACE 181
PATH 187
USERCATALOG 191
DELETE 207
DIAGNOSE 227
EXAMINE 235
EXPORT 239
EXPORT DISCONNECT 247
IMPORT 249
IMPORT CONNECT 263
LISTCAT 267
PRINT 277
REPRO 287
SHCDS 311
VERIFY 319
COMPACTION parameter
ALTER VOLUMEENTRY command 82
CREATE VOLUMEENTRY command 102
COMPAREDD parameter
DIAGNOSE command 228

COMPAREDS parameter
DIAGNOSE command 228
condition codes
IF-THEN-ELSE tests 23
processor, access method services 399
resetting 15
CONNECT parameter
IMPORT command 263
connecting a catalog 263
CONSOLENAME parameter
ALTER LIBRARYENTRY command 77
CREATE LIBRARYENTRY command 96
continuation errors, modal commands 23
continuing commands xxv
control area
preformatting
alternate index 133, 158
control interval
crossing boundaries 158
CONTROLINTERVALSIZE parameter
DEFINE command
ALTERNATEINDEX 126
CLUSTER 146
USERCATALOG 195
copying
catalog 300, 303
dummy records 289
ISAM records 289
linear data set 288, 291, 295
SAM data set 291
COUNT parameter
PRINT command 282
REPRO command 296
CREATE LIBRARYENTRY command
examples 98, 99
format 95
parameters
optional 96, 97
required 95
CREATE VOLUMEENTRY command
examples 105, 106
format 101
parameters
optional 101, 105
required 101
CREATION parameter
LISTCAT command
description 270
output listings 378
cross-region sharing
ALTER command 68
DEFINE command
ALTERNATEINDEX 130
CLUSTER 156
USERCATALOG 198
cross-system sharing
ALTER command 69
DEFINE command
ALTERNATEINDEX 131
CLUSTER 157
USERCATALOG 198
cryptographic parameters
REPRO command
description 297, 300
examples 305, 311
CYLINDERS parameter
ALLOCATE command 45
DEFINE command
ALTERNATEINDEX 123
CLUSTER 141
PAGESPACE 181
USERCATALOG 192

# D

data buffers 194
data class
attribute selection order 14
description 28
data component
alternate index 121, 135
cluster
control interval size 147
record size 154
specifying attributes 139, 160
examining, example 237
user catalog 200
data encrypting key
cryptographic option 297
establishing 298
data entry keywords, LISTCAT output 329
data integrity
sharing catalogs 198
sharing data sets
cluster 156
DEFINE ALTERNATEINDEX command 130
data organization, specifying 150
DATA parameter
LISTCAT command 268
data set
altering expiration date 74
copying 287
defining cluster
description 141
examples 160, 168
identifying 1
organization 150
page space
entry definition 181
scanning for diagnosis 227
security
definition 3
RACF authorization 322, 324
sharing
ALTER command 68
cluster 156
DEFINE ALTERNATEINDEX command 130
type 150
DATACLAS parameter
ALLOCATE command 35

DATACLASS parameter
DEFINE command
ALTERNATEINDEX 126
CLUSTER 148
PAGESPACE 183
USERCATALOG 195
DATAKEYFILE parameter
REPRO command 298, 299
DATAKEYVALUE parameter
REPRO command 298, 300
DATASET parameter
ALLOCATE command 32
VERIFY command 319
DATATEST parameter
EXAMINE command 236
DB2 (Database 2)
migration to linear data set, example 74
DBCS (double-byte character set) parameter
PRINT command 279
REPRO command 289
DCOLLECT command
batch environment 112
output record field description
active data set record 445
backup data set record 477
DASD capacity planning record 479
header record 444
migrated data set record 475
tape capacity planning record 481
volume record 451
VSAM association record 449
output record structure 411
active data set record 413
backup data set record 419
DASD capacity planning record 420
header record 413
migrated data set record 417
tape capacity planning record 421
volume information record 417
VSAM association record 415
output record types 411
parameters
optional 109, 112
required 108, 109
security considerations 108
APF authorization 108
RACF authorization 108
SMS configuration information 107
syntax 108
user exit, writing 405
DCOLLECT data collection
main collection areas
active data sets 107
capacity planning 107
inactive data 107
SMS Configuration Data Set Information 107
volume information 107
VSAM data sets 107
DD statement
sort work file statements 92
DD statements, examples 3, 4
DDM indicator 446
DDNAME parameter
ALLOCATE command 32
debugging tool
PARM command 20
decimal form subparameters, coding xxiii
DECIPHER parameter
REPRO command 299, 306
decipher, example
using private keys 308
using system keys 306
DEFINE command
ACCOUNT
optional parameters 145
ALIAS
examples 118
format 117
optional parameters 118
required parameters 117
ALTERNATEINDEX
data component 121
examples 135, 137
format 121, 122
index component 122
optional parameters 125, 135
required parameters 122, 125
attribute selection order 14
catalog selection order 10
CLUSTER
data component 139
data organization 150
examples 160, 168
format 139, 140
index component 140
optional parameters 160
required parameters 141, 145
GENERATIONDATAGROUP
format 169
optional parameters 169, 171
required parameters 169
MASTERCATALOG
optional parameters 200
required parameters 192, 194
NONVSAM
examples 178, 179
format 173
optional parameters 176, 178
required parameters 174, 176
PAGESPACE
examples 185, 186
format 181
optional parameters 182
required parameters 181
PATH
examples 189, 190
format 187
optional parameters 187
required parameters 187
USERCATALOG
components 200
examples 200, 205

DEFINE command *(continued)*
USERCATALOG *(continued)*
format 191
optional parameters 194, 200
required parameters 192
Delete command
OAM entry delete 225
DELETE command
catalog search order 10
examples 217, 226
format 207
generic catalog selection 13
parameters
optional 208, 217
required 207
delimiting a range
PRINT command 279
REPRO command 291
DEN parameter
ALLOCATE command 36
DENYNONRLSUPDATE parameter
SHCDS command 315
DESCRIPTION parameter
ALTER LIBRARYENTRY command 78
CREATE LIBRARYENTRY command 96
device
type translate table 346
DEVICETYPES parameter
DEFINE command
NONVSAM 174
IMPORT CONNECT command 263
DFSMS Data Collection Facility
output data set 411
DFSMShsm (DFSMS Hierarchical Storage Manager)
delete function 215, 217
DIAGNOSE command
examples 230, 234
format 227
optional parameters 228, 230
required parameters 227
DIAGNS parameter
ALLOCATE command 36
DIR parameter
ALLOCATE command 36
disability 483
DISCONNECT parameter
EXPORT command 247
disconnecting catalog 247
Distributed Data Management (DDM) 446
DO command 15, 18
DO-END command sequence 18
DSNTYPE parameter
ALLOCATE command 36
DSORG parameter
ALLOCATE command 36
dummy records
copying 289
DUMP parameter
DIAGNOSE command 229
PRINT command 278
dump, after abnormal termination 3
dynamic allocation
non-VSAM data set 2
volume 2
VSAM data set 2
DYNAMNBR parameter
ALLOCATE command
description 27
example 47

# E

ECSHARING parameter
ALTER command 59
DEFINE USERCATALOG command 195
ELSE parameter
IF command 17
EMPTY parameter
ALTER command 60
DEFINE command 170
ENCIPHER parameter
REPRO command 297, 305
using private keys 307
using system keys 305
enciphering records 297
END command 15, 18
ENTEREJECTDATE parameter
ALTER VOLUMEENTRY command 82
CREATE VOLUMEENTRY command 102
ENTRIES parameter
DIAGNOSE command 229, 230
LISTCAT command
description 270
output listings 378
REPRO command 290
entry-sequenced cluster
defining 151
example
copying and printing records 283
defining reusable 164
defining with expiration beyond 1999 167
defining with model 166
exporting using CIMODE 245
importing 259
entryname subparameter
ALTER LIBRARYENTRY command 77
ALTER VOLUMEENTRY command 81
coding xxiii
VVDS 141
entrypoint subparameter, coding xxiv
ENVIRONMENT parameter
REPRO command 289
ERASE parameter
ALTER command 60
DEFINE command
ALTERNATEINDEX 127
CLUSTER 148
DELETE command 212
EXPORT command 241
IMPORT command 252
EROPT parameter
ALLOCATE command 37

ERRORLIMIT parameter
DCOLLECT command 109
DIAGNOSE command 229
EXAMINE command 236
REPRO command 290
estimating primary space 411
EXAMINE command
examples 236, 237
format 235
parameters
optional 235, 236
required 235
exception, I/O error 60
EXCEPTIONEXIT parameter
ALTER command 60, 66
DEFINE command
ALTERNATEINDEX 127
CLUSTER 148
EXCLUDE parameter
DIAGNOSE command 229
EXCLUDEVOLUMES parameter
DCOLLECT command 109
EXEC statement 5
execution control commands
CANCEL command 19
DO-END command sequence 18
IF-THEN-ELSE sequence 15
null command 17
PARM command 19
SET command 18
EXITNAME parameter
DCOLLECT command 109
EXPDT parameter
ALLOCATE command 37
expiration date
example
altering, data set 74
defining, entry-sequenced cluster 167
EXPIRATION parameter
LISTCAT command
description 271
output listings 378
EXPIRATIONDATE parameter
ALTER VOLUMEENTRY command 82
CREATE VOLUMEENTRY command 102
explicitly specified attribute
selection order 14
EXPORT
restriction 130, 155
EXPORT command
examples 243, 245
format 239
parameters
optional 240, 243
required 239, 240
EXPORT DISCONNECT command
catalog selection order 12
example 248
format 247
required parameters 247
external file key
cryptographic option 297
EXTERNALKEYNAME parameter
REPRO command 297
EXTERNALSORT parameter
BLDINDEX command 88

# F

fall back
SHCDS command
CFRESET 316
CFRESETDS 316
field names, LISTCAT listings 327, 335
FILE
restriction 127, 149
FILE parameter
ALLOCATE command 32
ALTER command 60
DEFINE command
ALTERNATEINDEX 127
CLUSTER 148
PAGESPACE 183
USERCATALOG 196
DELETE command 212
IMPORT command 253
LISTCAT command 271
REPRO command 291
VERIFY command 319
FILEDATA parameter
ALTER command 61
Network File System Server 61
FILESEQUENCENUMBERS parameter
DEFINE command
NONVSAM 176
filtering
attribute selection order 14
fixed-length records
defining 154
FOR parameter
ALTER command 71
DEFINE command
ALTERNATEINDEX 133
CLUSTER 160
GENERATIONDATAGROUP 171
NONVSAM 178
PAGESPACE 185
PATH 189
USERCATALOG 199
FORCE parameter
DELETE command 213
FRBIND parameter
SHCDS command 314
FREESPACE parameter
ALTER command 61
DEFINE command
ALTERNATEINDEX 127
CLUSTER 149
USERCATALOG 196
FROMADDRESS parameter
PRINT command 279

FROMADDRESS parameter *(continued)*
REPRO command 291
FROMKEY parameter
PRINT command 279
REPRO command 291
FROMNUMBER parameter
PRINT command 280
REPRO command 292
FRRESETRR parameter
SHCDS command 314
FRSETRR parameter
SHCDS command 314
FRUNBIND parameter
SHCDS command 314
FULL parameter
PARM command 20
function, summary of 25

## G

GDG
restriction 172
GDG (generation data group)
defining 169
deleting with recovery 223
example 223
altering attributes 74
defining 171, 172
deleting 222
LISTCAT output
base entry keyword fields 340
base entry keywords 333
listing entries 222
GDS (generation data set)
cataloging maximum number 60
renaming 64
roll-in, example 73
ROLLIN parameter
ALTER command 68
SMS restriction 68
uncataloging 68
GENERATIONDATAGROUP parameter
DEFINE command 169
DELETE command 209
LISTCAT command 268
generic key
specified in PRINT command 279
generic name
altering 73
device type 174
listing catalog entries, example 275
global resource serialization 156
graphic character set 20
GRAPHICS parameter
PARM command 20
group names
LISTCAT output field names 335
LISTCAT output keywords 327
GRS (global resource serialization)
ALTER command 68
GRS (global resource serialization) *(continued)*
defining
alternate index 131
cluster 156
user catalog 198
guaranteed space
data set allocation 2, 47
example 201
specifying storage class 193

## H

HEX parameter
PRINT command 278
hexadecimal form subparameters, coding xxiii
HIS (history group)
keyword fields, LISTCAT output 340
HISTORY parameter
LISTCAT command 272
output listings 374

## I

I/O
error, exception 60
user routines 399
ICFCATALOG parameter
DEFINE command
USERCATALOG 196
DIAGNOSE command 227
IDCAMS commands
security
RACF authorization 325
IDCAMS program, invoking 4, 6, 396
IF command 16, 17
IF-THEN-ELSE command sequence
null command 17
specifying 15
IGG.CATLOCK
requirement 258
IMBED
NOIMBED xxvii
IMPORT command
examples 257, 259
format 249
parameters
optional 251, 257
required 249
IMPORT CONNECT command
example 264
format 263
parameters
optional 263, 264
required 263
important
ALTER 77
INCLUDE parameter
DIAGNOSE command 229
INDATASET parameter
BLDINDEX command 87
DIAGNOSE command 228

INDATASET parameter *(continued)*
IMPORT command 250
PRINT command 277
REPRO command 289
index
alternate
components 135
defining 122
buffer 194
cluster 140, 160
entry keywords 331
EXAMINE command 236
user catalog 200
INDEX parameter
LISTCAT command 268
INDEXED parameter
DEFINE command
CLUSTER 150
INDEXTEST parameter
EXAMINE command 235
INFILE parameter
BLDINDEX command 87
DIAGNOSE command 228
EXPORT command 241
IMPORT command 250
PRINT command 277
REPRO command 289
INHIBIT parameter
ALTER command 62
INHIBITSOURCE parameter
EXPORT command 242
INHIBITTARGET parameter
EXPORT command 242
INSERTALL parameter
PRINT command 281
REPRO command 292
INSERTSHIFT parameter
PRINT command 280
REPRO command 292
integrated catalog facility catalog
example
defining 200
deleting 218
examining 236
importing 258
merging user catalogs 302
exporting
description 239
example 243
locking and unlocking
ALTER 63
DEFINE USERCATALOG 197
IMPORT 253
restoring 249
internal file key
cryptographic option 297
INTERNALKEYNAME parameter
REPRO command 297
INTERNALSORT parameter
BLDINDEX command 89
interpreting output 413
interpreting SHCDS commands
LISTDS examples 383
INTOEMPTY parameter
IMPORT command 252
invoking access method services 4

# J

JCL (job control language)
access method services, invoking 5
allocating VSAM data sets 4
DCOLLECT command
examples 112
LISTCAT jobs 347
JOB statement 5
job step
canceling 19
JOBCAT DD statement
SMS restriction 2

# K

KEEP parameter
ALLOCATE command 37
key
cryptographic option 297
field 128
pointer pair 91
value 134
key-sequenced cluster
data set 150
example
copying records 300
defining 160, 163, 165
deleting 220
examining 236
exporting 244, 245
importing 259
listing 273
printing data records 283
specifying data and index parameters 161
keyboard 483
KEYLEN parameter
ALLOCATE command 37
KEYOFF parameter
ALLOCATE command 38
KEYRANGE xxvii
KEYS parameter
ALTER command 62
DEFINE command
ALTERNATEINDEX 128
CLUSTER 151
keyword
LISTCAT output 327, 346
parameters xxi
KILOBYTES parameter
DEFINE command
ALTERNATEINDEX 123
CLUSTER 141
PAGESPACE 181
USERCATALOG 192

# L


LABEL parameter
  ALLOCATE command 38
LASTCC
  condition code 399
  parameter
    IF command 16
    modal command execution example 18
    replacing value 18
    SET command 18
length, alternate key 128
LEVEL parameter
  DIAGNOSE command 229, 230
  LISTCAT command
    description 270
    output listings 378
  REPRO command 290
LIBDEVTYPE parameter
  ALTER LIBRARYENTRY command 78
  CREATE LIBRARYENTRY command 96
LIBRARY parameter
  LISTCAT command 271
LIBRARYENTRIES parameter
  LISTCAT command 269
LIBRARYENTRY parameter
  ALTER LIBRARYENTRY command 77
  CREATE LIBRARYENTRY command 95
  DELETE command 209
LIBRARYID parameter
  ALTER LIBRARYENTRY command 78
  CREATE LIBRARYENTRY command 95
LIBRARYNAME parameter
  ALTER VOLUMEENTRY command 82
  CREATE VOLUMEENTRY command 102
LIKE parameter
  ALLOCATE command 38
LIMCT parameter
  ALLOCATE command 39
LIMIT parameter
  ALTER command 63
  DEFINE command
    GENERATIONDATAGROUP 169
linear data set
  altering 54
  cluster
    data organization 150
    specifying 150
  copying 288, 291, 295
  example
    defining 168
    migrating from DB2 74
    printing 285
  exporting 239, 240
  printing 279, 281
LINEAR parameter
  ALTER command 71
  DEFINE command
    CLUSTER 150
LINK macro 393
LIST parameter
  DIAGNOSE command 230
LISTALL parameter
  SHCDS command 314
LISTCAT command
  catalog search order 12
  examples 273, 346
  format 267
  generic catalog selection 13
  interpreting output 327
  non-VSAM tailored compressed dataset 367
  output keywords 327
  output listings 346
    ALL 356
    ALLOCATION 368
    CREATION/EXPIRATION 378
    device type translate table 346
    ENTRIES 378
    HISTORY 374
    LEVEL 378
    NAMES 351
    no parameters specified 349
    VOLUME 351
  output messages 348
  parameters
    optional 268, 273
    required 267, 268
  RACF security authorization 324
  TSO environment 381
LISTDATA command
  UNITNUMBER subparameter 403
LISTDS parameter
  SHCDS command 312
LISTRECOVERY parameter
  SHCDS command 313
LISTSUBSYS parameter
  SHCDS command 313
LISTSUBSYSDS parameter
  SHCDS command 313
LOAD macro 393, 394
local system queue area 185
LOCATION parameter
  ALTER VOLUMEENTRY command 83
  CREATE VOLUMEENTRY command 102
LOCK parameter
  ALTER command 63
  DEFINE command
    USERCATALOG 197
  IMPORT command 253
LOG parameter
  Alter command 62
  ALTER command
    Nullify command 66
  DEFINE CLUSTER command 151
  LISTCAT command 329
LOGICALTYPE parameter
  ALTER LIBRARYENTRY command 78
  CREATE LIBRARYENTRY command 96
LOGSTREAMID parameter
  Alter command 63
  ALTER command
    Nullify command 66

LOGSTREAMID parameter *(continued)*
DEFINE command
CLUSTER 152
LISTCAT command 329
LRECL parameter
ALLOCATE command 39
LSQA (local system queue area)
defining page spaces 185

# M

macro
ATTACH 393
CALL 394
invoking instructions 393
LINK 393
LOAD 394
management class
description 28
MANAGEMENTCLASS parameter
ALTER command 64
DEFINE command
CLUSTER 152
PAGESPACE 183
USERCATALOG 197
IMPORT command 254
MARGINS parameter
PARM command 21
margins, coding xxi
master catalog
defining 192
MASTERCATALOG parameter
DEFINE command
USERCATALOG 192
MAXCC parameter
IF-THEN-ELSE command sequence 16
modal command execution example 19
replacing value 19
SET command 18
MAXVOL parameter
ALLOCATE command 40
MEDIATYPE parameter
ALTER VOLUMEENTRY command 83
CREATE VOLUMEENTRY command 103
MEGABYTES parameter
DEFINE command
ALTERNATEINDEX 123
CLUSTER 141
PAGESPACE 181
USERCATALOG 192
MERGECAT parameter
REPRO command 293
merging
integrated catalog facility catalog 290
messages, LISTCAT output 348
MGMTCLAS ACS routine
Newname parameter 64
MGMTCLAS parameter
ALLOCATE command 40
MIGRATEDATA parameter
DCOLLECT command 110
migration
DB2 to linear data set 74
MIGRSNAPALL parameter
DCOLLECT command 110
modal command, execution
condition codes 23
continuation errors 23
defining 15
examples 23
null command 17
PARM command 24
TSO restrictions 6
MODEL parameter
DEFINE command
ALTERNATEINDEX 128
CLUSTER 152
PAGESPACE 183
PATH 188
USERCATALOG 197
model, example of using
defining entry-sequenced cluster 166
modeled attribute, selection order 14
MODULE parameter
ALTER command 66
MOUNTDATE parameter
ALTER VOLUMEENTRY command 83
CREATE VOLUMEENTRY command 103
multilevel alias facility
generic catalog selection 13
search order 9

# N

NAME parameter
CREATE LIBRARYENTRY command 95
CREATE VOLUMEENTRY command 101
DEFINE command
ALIAS 117
ALTERNATEINDEX 122
CLUSTER 141
GENERATIONDATAGROUP 169
NONVSAM 174
PAGESPACE 181
PATH 187
USERCATALOG 192
EXAMINE command 235
LISTCAT command
description 272
output listings 351
national characters, description xxv
NCP parameter
ALLOCATE command 41
nested IF commands 17
NEW parameter
ALLOCATE command 40
NEWNAME
restriction 254
NEWNAME parameter
ALTER command 64
IMPORT command 254
newname subparameter, coding xxiv

NOALIAS parameter
IMPORT command 251
NOCHECKPOINT parameter
ALTER VOLUMEENTRY command 82
CREATE VOLUMEENTRY command 102
NODATAINFO parameter
DCOLLECT command 110
NODATATEST parameter
EXAMINE command 236
NODBCSCHECK parameter
PRINT command 281
REPRO command 295
NODUMP parameter
DIAGNOSE command 229
NOECSHARING parameter
ALTER command 59
DEFINE USERCATALOG command 196
NOEMPTY parameter
ALTER command 60
DEFINE command
GENERATIONDATAGROUP 170
NOERASE parameter
ALTER command 60
DEFINE command
ALTERNATEINDEX 127
CLUSTER 148
DELETE command 212
EXPORT command 241
IMPORT command 252
NOFORCE parameter
DELETE command 214
NOINDEXTEST parameter
EXAMINE command 235
NOINHIBITSOURCE parameter
EXPORT command 242
NOINHIBITTARGET parameter
EXPORT command 242
NOLIST parameter
DIAGNOSE command 230
NOMERGECAT parameter
REPRO command 293
non-VSAM
data set
allocating 3
defining 174, 178
deleting 219
identifying 2
JCL statement for 3
RACF security authorization 323
entry, LISTCAT output
keywords 333
special field 340
non-VSAM tailored compressed dataset
LISTCAT output examples 367
NONINDEXED parameter
DEFINE command
CLUSTER 151
NONSPANNED parameter
DEFINE command
CLUSTER 158
nonspanned records
record size 154
NONUNIQUEKEY parameter
ALTER command 71
DEFINE command
ALTERNATEINDEX 134
NONVSAM parameter
DEFINE command 174
DELETE command 209
LISTCAT command 269
NOPURGE parameter
DELETE command 215
EXPORT command 242
IMPORT command 256
NORECATALOG parameter
DEFINE command
ALTERNATEINDEX 129
CLUSTER 154
NONVSAM 177
PAGESPACE 184
PATH 188
NORECOVERY parameter
DELETE command 215
NOREPLACE parameter
REPRO command 294
NOREUSE parameter
DEFINE command
ALTERNATEINDEX 130
CLUSTER 156
REPRO command 295
NOSAVRAC parameter
IMPORT command 256
NOSCRATCH parameter
ALTER command 68
DEFINE command
GENERATIONDATAGROUP 170
DELETE command 217
NOSORTCALL parameter
BLDINDEX command 89
NOSTOREDATAKEY parameter
REPRO command 298
NOSWAP parameter
DEFINE command
PAGESPACE 185
NOUPDATE parameter
ALTER command 72
DEFINE command
PATH 189
RLS (record-level sharing) 71
NOUPGRADE parameter
ALTER command 72
DEFINE command
ALTERNATEINDEX 134
NOVOLUMEINFO parameter
DCOLLECT command 110
NOWRITECHECK parameter
ALTER command 72
DEFINE command
ALTERNATEINDEX 134
CLUSTER 160
USERCATALOG 200

NOWRITEPROTECT parameter
ALTER VOLUMEENTRY command 85
CREATE VOLUMEENTRY command 105
null command 17
NULLIFY parameter
ALTER command 66
ALTER LIBRARYENTRY command 78
ALTER VOLUMEENTRY command 83
NUMBERED parameter
DEFINE command
CLUSTER 151
NUMBEREMPTYSLOTS parameter
ALTER LIBRARYENTRY command 78
CREATE LIBRARYENTRY command 96
NUMBERSCRATCHVOLUMES parameter
ALTER LIBRARYENTRY command 79
CREATE LIBRARYENTRY command 96
NUMBERSLOTS parameter
ALTER LIBRARYENTRY command 79
CREATE LIBRARYENTRY command 97
NVR parameter
DELETE command 210
NVS (non-VSAM entry special field)
special field, LISTCAT output 340

## O

OAM (Object Access Method)
collection
defining 176
specifying 174
DEFINE NONVSAM command
COLLECTION parameter 176
OBJECTS parameter
IMPORT command 253
IMPORT CONNECT command 263
objects, collection
defining catalog entry 173
OFF parameter
PARM command 20
offset, alternate key 128
OPTCD parameter
ALLOCATE command 41
ORDERED
UNORDERED xxvii
OUTDATASET parameter
BLDINDEX command 88
EXPORT command 240
IMPORT command 251
REPRO command 289
OUTFILE parameter
BLDINDEX command 88
DCOLLECT command 109
DIAGNOSE command 230
EXPORT command 240
IMPORT command 251
LISTCAT command 273
PRINT command 281
REPRO command 289
SHCDS command 317
output keywords
LISTCAT command 327
output listings 413
LISTCAT command 327
output messages
LISTCAT command 348
OWNER parameter
ALTER command 66, 67
DEFINE command
ALTERNATEINDEX 129
CLUSTER 153
GENERATIONDATAGROUP 170
NONVSAM 176
PAGESPACE 184
PATH 188
USERCATALOG 198
ownerid subparameter, coding xxiv
OWNERINFORMATION parameter
ALTER VOLUMEENTRY command 83
CREATE VOLUMEENTRY command 103

## P

page space
data set entry 181
defining 181, 185
deleting 224
entry keywords, LISTCAT output 334
PAGESPACE
restriction 210
PAGESPACE parameter
DEFINE command 181
DELETE command 210
LISTCAT command 269
paging slot
for page space 182
PARALLEL parameter
ALLOCATE command 46
PARM command
continuation errors 24
examples 21, 23
specifying processing options 19
partitioned data set
copying 289
deleting 208
deleting member 224
printing 277
renaming 57
path
defining a 187
entry keywords, LISTCAT output 334
PATH parameter
DEFINE command 187
DELETE command 210
LISTCAT command 269
PATHENTRY parameter
DEFINE command
PATH 187
PDSE
Allocate Command 34
LISTCAT command 341

pdsname subparameter, coding xxiv
PERMANENT parameter
  EXPORT command 243
PERMITNONRLSUPDATE example 317
PERMITNONRLSUPDATE parameter
  SHCDS command 315
PL/I program
  invoking access method services from 396
portable data set
  importing 250
POSITION parameter
  ALLOCATE command 41
preformatting
  control area
    alternate index 133, 158
primary space allocation
  alternate index 124
  cluster 143
  page space 182
  user catalog 193
prime key field
  specifying length of 151
PRINT command
  dump 278
  examples 283, 287
  format 277
  HEX format 278
  key-sequenced cluster data records 283
  parameters
    optional 277, 283
    required 277
  printing a catalog 283
printing
  by relative byte address 279, 281
  data set 277
  linear data set 279, 281, 285
private key
  decipher 308
  encipher 307
PRIVATE parameter
  ALLOCATE command 41
PRIVATEKEY parameter
  REPRO command 297
processor, access method services
  argument list 398
  condition codes 399
  invocation 398
programmed cryptographic option
  parameters, REPRO command 297, 300
PROTECT parameter
  ALLOCATE command 42
protection attribute
  nullifying 66
protection, RACF
  authorization levels 321
PRT (protection group)
  LISTCAT output fields 342
PURGE parameter
  DELETE command 215
  EXPORT command 242
  IMPORT command 256

## Q

qualified names, coding xxiii

## R

RACF (Resource Access Control Facility)
  data set profiles 256
  security authorization levels 321
RECATALOG parameter
  DEFINE command
    ALTERNATEINDEX 129
    CLUSTER 153
    NONVSAM 176
    PAGESPACE 184
    PATH 188
recataloging path, example 190
RECFM parameter
  ALLOCATE command 42
record
  characteristics 42
  enciphering 297
  fixed-length 154
  format 42
  length
    altering 67
    alternate index 129
    cluster 154
RECORDING parameter
  ALTER VOLUMEENTRY command 83
  CREATE VOLUMEENTRY command 103
RECORDMODE parameter
  EXPORT command 241
RECORDS parameter
  DEFINE command
    ALTERNATEINDEX 123
    CLUSTER 141
    PAGESPACE 181
    USERCATALOG 192
RECORDSIZE parameter
  ALTER command 67
  DEFINE command
    ALTERNATEINDEX 129
    CLUSTER 154
    USERCATALOG 198
RECORG parameter
  ALLOCATE command 43
RECOVERY parameter
  DEFINE command
    ALTERNATEINDEX 133, 159
  DELETE command 215
REFDD parameter
  ALLOCATE command 43
RELATE
  restriction 118
RELATE parameter
  DEFINE command
    ALIAS 117
    ALTERNATEINDEX 123
RELEASE parameter
  ALLOCATE command 43

REMOVESUBSYS parameter
SHCDS command 315
REMOVEVOLUMES parameter
ALTER command 67
REPLACE parameter
DCOLLECT command 110
REPRO command 294
REPLICATE
NOREPLICATE xxvii
REPRO
restriction 130, 155
REPRO command
cryptographic option parameters 297, 300
examples 300, 309
format 287
merging integrated catalog facility catalogs 302
parameters
optional 289, 297
required 288, 289
requirement
ALIAS 257
BLDINDEX 87
IGG.CATLOCK 258
requirements
SHCDS 312
resetting condition codes 15
restriction
EXPORT 130, 155
FILE 127, 149
GDG 172
NEWNAME 254
PAGESPACE 210
RELATE 118
REPRO 130, 155
REUSE 156
SHIPKEYNAMES 298, 299
SYMBOLICRELATE 118
RETENTION parameter
ALTER command 66
retention period
alternate index 133
cluster 159
deleting entry 214
generation data set 170
non-VSAM data set 177
page space 185
path 188
user catalog 199
RETPD parameter
ALLOCATE command 37
return codes
ALLOCATE command 30
REUSE
restriction 156
REUSE parameter
ALLOCATE command 43
DEFINE command
ALTERNATEINDEX 130
CLUSTER 155
REPRO command 295
REUSE|NOREUSE parameter
ALTER command 68
RLS (record-level sharing)
Alter command
NOUPDATE parameter 71
ALTER command 58
BUFND parameter 58
DEFINE ALTERNATEINDEX command 128
DEFINE CLUSTER command 151
DELETE Command 209
ROLLIN parameter
ALTER command 68
ROUND parameter
ALLOCATE command 44
RRDS (relative record data set)
data organization 151
defining, example 163
variable-length record (VRRDS)
examining structural integrity 235

# S

SAVRAC parameter
IMPORT command 256
scanning a data set 227
SCHDS example using PERMITNONRLSUPDATE 317
SCHDS example with SYSPLEX 317
SCRATCH parameter
ALTER command 68
DEFINE command
GENERATIONDATAGROUP 170
DELETE command 216
SCRATCHTHRESHOLD parameter
ALTER LIBRARYENTRY command 79
CREATE LIBRARYENTRY command 97
SECMODEL parameter
ALLOCATE command 44
secondary space allocation
alternate index 124
cluster 143
user catalog 193
security authorization
description 2
levels
catalogs 321
data set operations 324
LIB/VOL operations 324
LISTCAT operations 324
non-VSAM data sets 323
required RACF 321
VOLCAT operations 324
VSAM data sets 322
RACF authorization
IDCAMS commands 325
RACF authorization 325
segmenting subparameter names xxiii
selection order, specifying attributes 14
SET command
description 18
resetting condition codes 15

SETCACHE command
  UNITNUMBER subparameter 403
SHAREOPTIONS parameter
  ALTER command 68
  DEFINE command
    ALTERNATEINDEX 130
    CLUSTER 156
    USERCATALOG 198
sharing
  cross-region
    ALTER command 68
    DEFINE ALTERNATEINDEX command 130
    DEFINE CLUSTER command 156
    DEFINE USERCATALOG command 198
  cross-system
    ALTER command 69
    DEFINE ALTERNATEINDEX command 131
    DEFINE CLUSTER command 157
    DEFINE USERCATALOG command 198
SHCDS
  requirements 312
SHCDS command
  format 311
  FRDELETEUNBOUNDLOCKS 314
  interpreting SHCDS commands
    LISTRECOVERY example 390
    LISTSUBSYS example 386
    LISTSUBSYSDS examples 388
  parameters 312
  record-level sharing (RLS) 311
SHELFLOCATION parameter
  ALTER VOLUMEENTRY command 84
  CREATE VOLUMEENTRY command 104
SHIPKEYNAMES
  restriction 298, 299
SHIPKEYNAMES parameter
  REPRO command 298
shortcut keys 483
SKIP parameter
  PRINT command 280
  REPRO command 292
SKIPDBCSCHECK parameter
  PRINT command 281
  REPRO command 295
SMS (Storage Management Subsystem)
  ALLOCATE command 27
  catalog considerations 2
  data set
    classes, description 28
    listing, example 273
  defining alternate index 135
  JOBCAT/STEPCAT restriction 2
  volume identification 1
SMSDATA parameter
  DCOLLECT command 110
SORTCALL parameter
  BLDINDEX command 89
SORTDEVICETYPE parameter
  BLDINDEX command 89
SORTFILENUMBER parameter
  BLDINDEX command 89
SORTMESSAGEDD parameter
  BLDINDEX command 89
SORTMESSAGELEVEL parameter
  BLDINDEX command 90
space allocation
  alternate index
    defining 123
    free space 127
  cluster
    defining 142
    free space 149
  importing, output object 255
  page space 181
  user catalog
    defining 192
    free space 196
SPACE parameter
  ALLOCATE command 44
SPANNED parameter
  DEFINE command
    CLUSTER 158
spanned records
  record size 129, 154
special characters, coding xxiii, xxv
SPECIALATTRIBUTE parameter
  ALTER VOLUMEENTRY command 84
  CREATE VOLUMEENTRY command 104
SPEED parameter
  DEFINE command
    ALTERNATEINDEX 133, 158
STA (statistics group)
  LISTCAT output keyword fields 342
STEPCAT DD statement
  SMS restriction 2
storage class
  description 28
storage control unit
  caching models 403
STORAGECLASS parameter
  ALTER command 70
  DEFINE command
    CLUSTER 159
    PAGESPACE 184
    USERCATALOG 199
  IMPORT command 254
STORAGEGROUP parameter
  ALTER VOLUMEENTRY command 84
  CREATE VOLUMEENTRY command 104
  DCOLLECT command 111
STORCLAS parameter
  ALLOCATE command 45
STOREDATAKEY parameter
  REPRO command 298
STOREKEYNAME parameter
  REPRO command 299
STRING parameter
  ALTER command 66
string subparameter, coding xxv
STRNO parameter
  ALTER command 70

STRNO parameter *(continued)*
DEFINE command
USERCATALOG 199
subparameters, coding xxii, xxv
SWAP parameter
DEFINE command
PAGESPACE 185
SYMBOLICRELATE
restriction 118
SYMBOLICRELATE parameter
DEFINE command
ALIAS 118
SYNAD exit routine 60
SYSIN DD statement 5
SYSPLEX qualifier with SCHDS 317
SYSPRINT data set 4
SYSPRINT DD statement 5
SYSTEMDATAKEY parameter
REPRO command 300
SYSTEMKEY parameter
REPRO command 300
SYSTEMKEYNAME parameter
REPRO command 300

# T

TABLE parameter
PARM command 21
tape library
date format 9
IDCAMS commands 7
naming conventions
library 9
tape volume 9
summary of IDCAMS support 7
target data set
alternate 4
JCL DD statement 4
TSO restrictions 6
TEMPORARY parameter
EXPORT command 243
terminal monitor program 27
terminator, coding xxvi
TEST parameter
PARM command 20
THEN parameter 15
IF command 17
time sharing option 5
TO parameter
ALTER command 71
DEFINE command
ALTERNATEINDEX 133
CLUSTER 159
GENERATIONDATAGROUP 170
NONVSAM 177
PAGESPACE 185
PATH 189
USERCATALOG 199
TOADDRESS parameter
PRINT command 282
REPRO command 296
TOKEY parameter
PRINT command 281
REPRO command 296
TONUMBER parameter
PRINT command 282
REPRO command 296
TRACE parameter
PARM command 20
trace tables 20
TRACKS parameter
ALLOCATE command 45
DEFINE command
ALTERNATEINDEX 123
CLUSTER 141
PAGESPACE 181
USERCATALOG 192
translate table, device type 346
TRTCH parameter
ALLOCATE command 46
TRUENAME parameter
DELETE command 210
TSO (time sharing option)
invoking access method services 5
LISTCAT output examples 381
modal command 6
TYPE parameter
ALTER command 71

# U

UCOUNT parameter
ALLOCATE command 46
UNINHIBIT parameter
ALTER command 62
UNIQUEKEY parameter
ALTER command 71
DEFINE command
ALTERNATEINDEX 134
UNIT parameter
ALLOCATE command 46
UNLOCK parameter
ALTER command 64
DEFINE command
USERCATALOG 197
IMPORT command 253
UPDATE parameter
ALTER command 72
DEFINE command
PATH 189
UPGRADE parameter
ALTER command 72
DEFINE command
ALTERNATEINDEX 134
USEATTRIBUTE parameter
ALTER VOLUMEENTRY command 84
CREATE VOLUMEENTRY command 104
user catalog
components 200
connecting to master catalog 264
defining 191
deleting 221

user catalog *(continued)*
disconnecting 247
entry keywords 335
merging, example 302, 303
user program
invoking access method services 393
macro instructions 393
user, I/O routines 399
USERCATALOG parameter
DEFINE command 192
DELETE command 211
LISTCAT command 269
USERDATA parameter
REPRO command 299

# V

VERIFY command
example 319
format 319
required parameters 319
virtual
storage
alternate index 90
external record sort 91
internal record sort 91
VLS (volumes group)
LISTCAT output keyword fields 344
VOLCATALOG parameter
DEFINE command
USERCATALOG 196
IMPORT CONNECT command 264
VOLCAT (tape volume catalog)
general 196
specific 196
volser subparameter, coding xxv
volume
adding to candidate list 57
cleanup 67
identification
description 1
dynamic allocation 2
JCL DD statement 3
SMS restriction 1
removing from candidate list 67
serial number (volser), naming
DASD xxv
tape volume 9
TSO restrictions 6
VOLUME parameter
ALLOCATE command 46
DEFINE command
PAGESPACE 182
VOLUMEENTRIES parameter
LISTCAT command 269
REPRO command 296
VOLUMEENTRY parameter
ALTER VOLUMEENTRY command 81
CREATE VOLUMEENTRY command 101
DELETE command 211
VOLUMES parameter
DCOLLECT command 111
DEFINE command
ALTERNATEINDEX 124
CLUSTER 144
NONVSAM 175
USERCATALOG 193
IMPORT command 254
IMPORT CONNECT command 263
LISTCAT command
description 272
output listings 351
VSAM (virtual storage access method)
data set
allocation 3
copying records 301
dynamic allocation 2
RACF security authorization 322
VSAM compressed data set
Altering data set characteristics 54
VSAM data set
NOREUSE parameter
ALTER command 68
REUSE parameter
ALTER command 68
VSEQ parameter
ALLOCATE command 47
VVDS (VSAM volume data set)
DEFINE CLUSTER example 166
entryname 141
parameter, DIAGNOSE command
description 227
example 230, 231
VVR parameter
DELETE command 211

# W

WORKFILE parameter
BLDINDEX command 90
WRITECHECK parameter
ALTER command 72
DEFINE command
ALTERNATEINDEX 134
CLUSTER 160
USERCATALOG 200
WRITEDATE parameter
ALTER VOLUMEENTRY command 85
CREATE VOLUMEENTRY command 105
WRITEPROTECT parameter
ALTER VOLUMEENTRY command 85
CREATE VOLUMEENTRY command 105

# Readers' Comments — We'd Like to Hear from You

**z/OS**
**DFSMS Access Method Services for Catalogs**

**Publication No. SC26-7394-01**

**Overall, how satisfied are you with the information in this book?**

| | Very Satisfied | Satisfied | Neutral | Dissatisfied | Very Dissatisfied |
|---|---|---|---|---|---|
| Overall satisfaction | ☐ | ☐ | ☐ | ☐ | ☐ |

**How satisfied are you that the information in this book is:**

| | Very Satisfied | Satisfied | Neutral | Dissatisfied | Very Dissatisfied |
|---|---|---|---|---|---|
| Accurate | ☐ | ☐ | ☐ | ☐ | ☐ |
| Complete | ☐ | ☐ | ☐ | ☐ | ☐ |
| Easy to find | ☐ | ☐ | ☐ | ☐ | ☐ |
| Easy to understand | ☐ | ☐ | ☐ | ☐ | ☐ |
| Well organized | ☐ | ☐ | ☐ | ☐ | ☐ |
| Applicable to your tasks | ☐ | ☐ | ☐ | ☐ | ☐ |

**Please tell us how we can improve this book:**

Thank you for your responses. May we contact you? ☐ Yes ☐ No

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

Name _____ Address _____

Company or Organization _____

Phone No. _____

**Readers' Comments — We'd Like to Hear from You**
SC26-7394-01

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

# BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
RCF Processing Department
M86/050
5600 Cottle Road
San Jose, CA 95193-0001

Fold and Tape | **Please do not staple** | Fold and Tape

SC26-7394-01

Cut or Fold
Along Line

Program Number: 5694-A01

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC26-7394-01