IBM

MVS/Extended Architecture VSAM Catalog Administration: Access Method Services Reference

Licensed Program

Order Number
GC26-4136-3

Data Facility Product
5665-XA2

Version 2
Release 3.0

IBM

MVS/Extended Architecture VSAM Catalog Administration: Access Method Services Reference

Licensed Program

Order Number
GC26-4136-3

Data Facility Product
5665-XA2

Version 2
Release 3.0

Fourth Edition (December 1987)

This is a major revision of, and makes obsolete, GC26-4136-2.

This edition applies to Version 2 Release 3.0 of MVS/Extended Architecture Data Facility Product, Licensed Program 5665-XA2, and to any subsequent releases until otherwise indicated in new editions or technical newsletters.

The changes for this edition are summarized under "Summary of Changes" following the preface. Specific changes are indicated by a vertical bar to the left of the change. These bars will be deleted at any subsequent republication of the page affected. Editorial changes that have no technical significance are not noted.

Changes are made periodically to this publication; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/370, 30xx, 4300, and 9370 Processors Bibliography, GC20-0001, for the editions that are applicable and current.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's program may be used. Any functionally equivalent program may be used instead.

Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality. If you request publications from the address given below, your order will be delayed because publications are not stocked there.

A Reader's Comment Form is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, P.O. Box 50020, Programming Publishing, P.O. Box 49023, San Jose, California, U.S.A. 95161-9023. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

PREFACE

This publication is intended for catalog administrators and VSAM system programmers. It contains reference information about the access method services commands used to manipulate VSAM catalogs and VSAM data sets.

This book gives the format, a brief description, and examples of each access method services command used with VSAM catalogs and the objects cataloged in them. This book is meant to be a companion to MVS/Extended Architecture VSAM Administration Guide, GC26-4151, and MVS/Extended Architecture Catalog Administration Guide, GC26-4138.

For information on the use of commands related to VSAM catalog format and structure, see Catalog Administration Guide. For information on the use of commands related to VSAM data set format and structure, see VSAM Administration Guide.

ORGANIZATION

- Chapter 1, "Guide to Access Method Services" on page 1, summarizes the access method services functions and the commands used to perform them.
- Chapter 2, "Introduction" on page 3, provides an overview of access method services, including command syntax considerations, dynamic allocation, and JCL for data set and volume identification, and the use of the time sharing option (TSO) and the IBM 3850 Mass Storage System with VSAM and access method services. Everyone who intends to use access method services should first read this chapter.
- Chapter 3, "Controlling Command Execution" on page 21, describes the use of the IF-THEN-ELSE, SET, and PARM commands to control command execution and to specify diagnostic aids and printed-output options.
- Chapter 4, "Functional Command Format" on page 29, lists the access method services commands and their format, describes the required and optional parameters, and lists their abbreviations. Examples at the end of each command section show how the commands and their parameters are used.
- Appendix A, "Interpreting LISTCAT Output Listings," shows the output keywords and the field names associated with each type of catalog entry, and supplies listings for LISTCAT and their interpretation.
- Appendix B, "Interpreting LISTCRA Output Listings," shows the information necessary to interpret the listings for a recoverable catalog.
- Appendix C, "Sample Output from CHKLIST," shows the format and gives an explanation of the output for CHKLIST.
- Appendix D, "Caution for JCL DD Parameters," gives a list of some JCL DD parameters to be used with caution.
- Appendix E, "Invoking Access Method Services from a Problem Program," gives an example of how to use the access method services commands in a user program.
- Appendix F, "Command Reference Summary," shows the format of each access method services command, and its parameters, abbreviations, and defaults. The commands are grouped together to allow you to remove these pages and use them as a quick-reference guide.

- "Glossary" defines terms relevant to access method services, VSAM, VSAM data sets, and VSAM catalogs.

PREREQUISITE KNOWLEDGE

Readers of this book are presumed to have a background in programming, including:

- Job control language
- VSAM data administration
- Catalog administration

REQUIRED PUBLICATIONS

You should be familiar with the information presented in the following publications:

- MVS/Extended Architecture Catalog Administration Guide, GC26-4138, describes the administration of tasks for catalogs and how to use the access method services commands to manipulate catalogs, and the objects cataloged in them.
- MVS/Extended Architecture JCL User's Guide, GC28-1351,
- MVS/Extended Architecture JCL Reference, GC28-1352, describes the JCL parameters referred to in this publication.
- MVS/Extended Architecture Message Library: System Messages (Volumes 1 and 2), GC28-1376 and GC28-1377, provides a complete listing of the messages issued by access method services.
- MVS/Extended Architecture Data Administration: Utilities, GC26-4150, describes the utility programs available for use with non-VSAM data sets.
- MVS/Extended Architecture VSAM Administration: Macro Instruction Reference, GC26-4152, provides information on macro instructions used to process VSAM data sets.
- MVS/Extended Architecture VSAM Administration Guide, GC26-4151, provides information on creating and processing VSAM data sets, backup and recovery, and various options affecting performance.

RELATED PUBLICATIONS

Within the text, references are made to the publications listed below:

Short Title	Publication Title	Order Number
Access Method Services Logic	MVS/Extended Architecture Access Method Services Logic	LY26-3953 and LY26-3997
Cache Device Administration	MVS/Extended Architecture Cache Device Administration	GC26-4137
Catalog Administration Guide	MVS/Extended Architecture Catalog Administration Guide	GC26-4138

Short Title	Publication Title	Order Number
Diagnosis Guide	MVS/Extended Architecture Data Facility Product Version 3: Diagnosis Guide	LY27-9550
Checkpoint/ Restart User's Guide	MVS/Extended Architecture Checkpoint/Restart User's Guide	GC26-4139
Data Administration Guide	MVS/Extended Architecture Data Administration Guide	GC26-4140
Introduction to the IBM 3850 Mass Storage System	Introduction to the IBM 3850 Mass Storage System (MSS)	GA32-0028
JCL User's Guide	MVS/Extended Architecture JCL User's Guide	GC28-1351
JCL Reference	MVS/Extended Architecture JCL Reference	GC28-1352
OS/VS Mass Storage System Services: Reference Information	OS/VS Mass Storage System (MSS) Services: Reference Information	GC35-0017
RACF Command Language Reference	Resource Access Control Facility (RACF) Command Language Reference	SC28-0733
RACF General Information Manual	Resource Access Control Facility (RACF): General Information Manual	GC28-0722
System-Data Administration	MVS/Extended Architecture System-Data Administration	GC26-4149
System Generation	MVS/Extended Architecture Installation: System Generation	GC26-4148
System Macros and Facilities	MVS/Extended Architecture System Programming Library: System Macros and Facilities, Volumes 1 and 2	GC28-1150 and GC28-1151
System Messages	MVS/Extended Architecture Message Library: System Messages, Volumes 1 and 2	GC28-1376 and GC28-1377
TSO Command Language Reference	MVS/Extended Architecture TSO Command Language Reference (OS/VS2 TSO Command Language Reference, as updated by Supplement SD23-0259 for MVS/XA)	GC28-0646
TSO Terminal User's Guide	MVS/Extended Architecture TSO Terminal User's Guide (as updated by Supplement SD23-0263 for TSO/E)	GC28-1274

Short Title	Publication Title	Order Number
Utilities	MVS/Extended Architecture Data Administration: Utilities	GC26-4150
VSAM Administration Guide	MVS/Extended Architecture VSAM Administration Guide	GC26-4151
VSAM Administration: Macro Instruction Reference	MVS/Extended Architecture VSAM Administration: Macro Instruction Reference	GC26-4152

NOTATIONAL CONVENTIONS

A uniform system of notation describes the format of access method services commands. This notation is not part of the language; it merely provides a basis for describing the structure of the commands.

The command format illustrations in this book use the following conventions:

- Brackets [ ] indicate optional parameters.
- Braces { } indicate a choice of entry; unless a default is indicated, you must choose one of the entries.
- Items separated by a vertical bar (|) represent alternative items. No more than one of the items may be selected.
- An ellipsis (...) indicates that multiple entries of the type immediately preceding the ellipsis are allowed.
- Other punctuation (parentheses, commas, etc.) must be entered as shown.
- **BOLDFACE** type indicates the exact characters to be entered. Such items must be entered exactly as illustrated (in uppercase, except in TSO).
- Lowercase underscored type specifies fields to be supplied by the user.
- **BOLDFACE UNDERSCORED** type indicates a default option. If the parameter is omitted, the underscored boldface value is assumed.
- A ' ' in the command format indicates a blank (an empty space) must be present before the next parameter.

SUMMARY OF CHANGES

RELEASE 3.0, JUNE 1987

SERVICE CHANGES

Minor technical changes have been made.

RELEASE 2.0, JUNE 1986

NEW PROGRAMMING SUPPORT

The EXAMINE command allows the user to analyze and report on the structural consistency of the index component and/or data component of a key-sequenced data set (KSDS) cluster. It is described in "Chapter 4. Functional Command Format" and in "Appendix F. Command Reference Summary."

Information has been added to the DEFINE and DELETE commands for non-VSAM data sets to reflect the RACF ERASE attribute available in RACF Version 1, Release 7.

SERVICE CHANGES

The first four sections of this manual have been reorganized.

Minor technical changes have been made.

RELEASE 1.0, APRIL 1985

NEW DEVICE SUPPORT

Selected examples in DEFINE NON VSAM and IMPORT CONNECT, a section of the required parameters for DEFINE PAGESPACE, and the Device Type Translate Table shown in Appendix A have been updated to reflect 3380 support.

VERSION 2 PUBLICATIONS

The Preface includes new order numbers for Version 2.

CONTENTS

FIGURES

CHAPTER 1. GUIDE TO ACCESS METHOD SERVICES

Figure 1 is a list of tasks that access method services commands can be used to perform. The left column shows tasks that you might want to perform. The middle column further describes the tasks. The right column shows the commands that can be used to perform each task.

Task	Description	Command
Alter	attributes in catalog entries	ALTER
Attach	user catalog to the master catalog	IMPORT CONNECT
Backup	VSAM catalog	REPRO
	VSAM data set	EXPORT or REPRO
	non-VSAM data set	REPRO
Build	alternate index	DEFINE ALTERNATEINDEX and BLDINDEX
Catalog	VSAM data set	DEFINE CLUSTER
	VSAM data set with alternate index	DEFINE CLUSTER and ALTERNATEINDEX
	non-VSAM data set	DEFINE NONVSAM
	generation data group	DEFINE GENERATIONDATAGROUP
Connect	user catalog to a master catalog or another user catalog	IMPORT CONNECT
	OS CVOL to a master catalog	DEFINE NONVSAM
Convert	non-VSAM data set to VSAM format	REPRO
	VSAM data set to SAM format	REPRO
	OS CVOL entries to VSAM catalog entries	CNVTCAT
Copy	data set or catalog	REPRO
Define	VSAM catalog	DEFINE USERCATALOG
	alias for a non-VSAM data set	DEFINE ALIAS
	alias for a user catalog	DEFINE ALIAS
	alias for a VSAM data set	DEFINE PATH
	alternate index	DEFINE ALTERNATEINDEX and BLDINDEX
	relationship between an alternate index and its base cluster	DEFINE PATH
	generation data group	DEFINE GENERATIONDATAGROUP
	page space	DEFINE PAGESPACE
	swap data set	DEFINE PAGESPACE
	VSAM data set	DEFINE CLUSTER
	VSAM data space	DEFINE SPACE
	catalog entry for a non-VSAM data set	DEFINE NONVSAM

Figure 1 (Part 1 of 2). Tasks and Commands

Task	Description	Command
Delete	user catalog	DELETE USERCATALOG
	alias	DELETE ALIAS
	VSAM data spaces and data sets	ALTER REMOVEVOLUMES
	alternate index and its paths	DELETE ALTERNATEINDEX
	path	DELETE PATH
	any type of catalog entry	DELETE
	VSAM data set	DELETE CLUSTER
	VSAM data space	DELETE SPACE
	generation data group	DELETE GENERATIONDATAGROUP
	non-VSAM data set	DELETE NONVSAM
	page space	DELETE PAGESPACE
Disconnect	user catalog	EXPORT DISCONNECT
Export	user catalog	EXPORT DISCONNECT
	VSAM data set	EXPORT or EXPORTRA
Import	user catalog	IMPORT CONNECT
	VSAM data set	IMPORT or IMPORTRA
Inspect	a key-sequenced data set (KSDS) cluster	EXAMINE
List	password	LISTCAT
	data set's contents	PRINT
	catalog's contents	PRINT
	contents of a catalog or of a catalog entry (formatted)	LISTCAT
	contents of a catalog recovery area	LISTCRA
	tapes mounted at a checkpoint	CHKLIST
Load	records into a data set	REPRO
	alternate index	BLDINDEX
Move	catalog to another system	EXPORT DISCONNECT and IMPORT CONNECT
	VSAM data set to another system	EXPORT/IMPORT
	non-VSAM data set to another system	REPRO
Print	data set	PRINT
Recover	from a processing-program failure	VERIFY
	from a catalog failure	LISTCRA EXPORTRA IMPORTRA and RESETCAT
Rename	data set	ALTER
Restore	data set's end-of-file information	VERIFY
	catalog entry and/or the contents of its object	EXPORTRA/IMPORTRA or RESETCAT
	data set from a portable copy	IMPORT
Uncatalog	data set	DELETE
Unload	data set or catalog	REPRO
Verify	VSAM data set's end-of-file indicators	VERIFY

Figure 1 (Part 2 of 2). Tasks and Commands

CHAPTER 2. INTRODUCTION

Access method services is a service program used with the virtual storage access method (VSAM) to establish and maintain catalogs and data sets. If you plan to use VSAM, or if you are responsible for maintaining the system catalog, you must use access method services commands.

There are two types of access method services commands: functional commands that are used to request the actual work (for example, defining a data set or listing a catalog) and modal commands that allow the conditional execution of functional commands. The access method services modal commands are presented under Chapter 3, "Controlling Command Execution" on page 21. The access method services functional commands and their parameters are listed in alphabetic order under Chapter 4, "Functional Command Format" on page 29. Examples of the use of each command are included. TSO users are allowed to use only the functional commands.

This is a reference book only. See Catalog Administration Guide and VSAM Administration Guide for information on the tasks performed using access method services commands.

HOW TO CODE ACCESS METHOD SERVICES COMMANDS

All access method services commands have this general structure:

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

Commands

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

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 specified as separators. Comments are strings of characters surrounded by a /* and an */. Comments may contain any characters except an '*/'.

Many of the commands and keyword parameters may be abbreviated. The abbreviations allowed are listed in the description of each command and in Appendix F, "Command Reference Summary" on page 326. Keyword parameters in plural form may also be coded in the singular form. Restrictions that apply to abbreviations under TSO are described under "From a TSO Terminal" on page 13.

Positional and Keyword Parameters

A parameter may 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 entryname to be deleted.

A keyword parameter is a specific character string, which may 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 may 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 consist of lists of items. Unless the list contains only one item, it must be enclosed in parentheses, which may be preceded and followed by blanks, commas, or comments. For example:

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

indicates that, if more than one entry is to be deleted, the list of entrynames must be enclosed in parentheses. However, if only one entryname is specified, 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))

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

In the first case, only one entry is to be renamed. The entryname and its new name are enclosed in parentheses.

In the second case, each entryname 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 that immediately follow a subparameter set that is enclosed in parentheses do not need to be separated from the closing parenthesis.

A value cannot contain 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.

In some parameters it is necessary to specify a password following the name of a catalog entry or the name of a JCL DD statement. You do this by coding the name, a slash, and the password. (Separators are allowed on either side of the slash.) For example:

```
DELETE PAYROLL/CTLGPAY
```

specifies the password CTLGPAY for the data set PAYROLL.

When you specify a password following a name, you must remember the convention for entering commands. If you code:

```
MYNAME/*WORD*/
```

MYNAME is treated as a name followed by the comment WORD. If you want *WORD*/ to be the password, you code either:

```
MYNAME/'*WORD*/'
```

or

```
MYNAME/ *WORD*/
```

Notice that the second example contains a blank after the first slash that separates the name from the password.

How to Code Subparameters

The following coding conventions apply to the subparameters listed in this section:

- When the subparameter contains a special character, enclose it in single quotation marks; for example, CODE('*SECRET*').
- 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 alphameric or special character. Because the character string will be right justified, you should use an even number of hexadecimal characters; for example, MASTERPW(X'E3D6D7') is the same as MASTERPW(TOP).
- When the subparameter contains a lowercase alphabetic character, it will be folded to an uppercase alphabetic character.

The subparameters most often referred to in this book are:

aliasname
: can contain 1 to 44 alphameric or national characters, and two 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 may 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 alphameric or special characters.

entryname
: can contain 1 to 44 alphameric or national characters, and two 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 may be specified between periods. A name that is 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.

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

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

entrypoint
: can contain 1 to 8 alphameric or national characters, and two 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 alphameric or national characters, and two 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 may be specified between periods.

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

ownerid
: can contain 1 to 8 EBCDIC characters.

password
: can contain 1 to 8 alphameric or special characters.

pdsname(membername)
: specifies the entryname for a partitioned data set. Blank characters are not allowed between the left and right parentheses enclosing the membername, or between pdsname and the left parenthesis.

string
: can contain 1 to 255 EBCDIC characters.

volser
: a volume serial number, can contain 1 to 6 alphameric, national, or special characters.

Alphameric, National, and Special Characters

The following is a list of alphameric, national, and special characters that are referred to throughout this book:

- Alphameric characters:

  alphabetic characters A-Z

  numeric characters 0-9

- National characters

  at sign [@]

  dollar sign [$]

  pound sign [#]

- Special characters

  ampersand [&]

  asterisk [*]

  blank [ ]

  braces [{ or }]

  brackets [[or]]

  comma [,]

  equal sign [=]

  hyphen [-]

  parenthesis [(or)]

  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 indicates continuation of the command. A plus sign indicates continuation of the command and continuation of a value within the command.

These two types of continuation are shown in the example below:

```
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 up with the THEN keyword.
  ```

Records ending with partial comments must always end with a continuation mark. Also, remember that only blank characters may appear between a continuation mark and the end of the record.

Note: The DO-END sequence does not require continuation characters. If continuation characters are used, they may be taken as a null command or cause unpredictable results.

The Terminator

The terminator indicates the end of the command. The terminator can be either a semicolon or the absence of a continuation mark.

If you use the semicolon as the terminator, it cannot be enclosed in quotation marks or embedded in a comment. Everything to the right of the semicolon is ignored.

If there is information to the right of the semicolon that is continued to another record, all the information, including the continued information, is ignored.

For example, if you code:

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

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

DATA SET AND VOLUME IDENTIFICATION

Access method services commands require that both data sets and volumes be identified when they are used. A data set must be identified when it is accessed. A volume or 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 catalog recovery area. (See Catalog Administration Guide for information on this option.)

A VSAM data set or volume can be identified through 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 JCL is not used, an attempt is made to dynamically allocate the data set or volume, as required.

DYNAMIC ALLOCATION

To dynamically allocate a VSAM data set, the data set name must exist and be cataloged. The catalog that contains the entry must be either a user catalog identified with a JOBCAT or STEPCAT DD statement, a user catalog whose name or alias is the first qualifier of the qualified data set name, or the master catalog.

To dynamically allocate a non-VSAM data set, the data set name must exist and must be cataloged. The catalog that contains the entry must be either a user catalog identified with a JOBCAT or STEPCAT DD statement, a user catalog or OS CVOL whose name or alias is the first qualifier of the qualified data set name, or the master catalog.

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

When a user catalog is identified by name in an access method services command (using the CATALOG parameter) or when VSAM determines that a specific user catalog must be accessed, an attempt is made to dynamically allocate it, if it has not been identified with a JOBCAT or STEPCAT DD statement.

In order to dynamically allocate a volume, it must already be mounted as permanently resident or reserved. The PRIVATE and PUBLIC use attributes should be carefully considered when you mount a volume.

JCL should be used for allocation with data sets that reside on Mass Storage System (MSS) volumes, because MSS volumes are rarely mounted as permanently resident or reserved. A request to dynamically allocate a data set on MSS, or an MSS volume, may result in a nonzero return code from allocation.

JCL DD STATEMENTS

When a JCL DD statement is used to identify a data set, the following information must be included on the DD statement.

- The data set name.
- The unit and volume serial number(s), if the data set is not cataloged.
- The disposition.
- For VSAM data sets, the AMP='AMORG' parameter is required if the JCL also supplies unit and volume serial number information.

For a VSAM Data Set

The two DD statements shown below demonstrate two different methods by which you can describe and allocate a VSAM data set:

```
//VSAMDS1  DD   DSN=VSAM.DATA.SET1,DISP=OLD
//VSAMDS2  DD   DSN=VSAM.DATA.SET2,VOL=SER=VSER01,
//              UNIT=DISK,DISP=OLD,AMP='AMORG'
```

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

VSAM uses default buffering for VSAM data sets. The default is the buffer space value contained in the data set's catalog entry. You may want to override this value using the BUFSP subparameter of the AMP parameter (see VSAM Administration Guide for information on this parameter).

For a Volume

To identify and allocate a volume, the following information must be included:

- The volume serial number
- The disposition
- The unit

For example, the DD statement shown below identifies and allocates volume VSER01:

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

Concatenated DD statements are supported under circumstances explicitly specified in the remainder of this publication.

"Direct Allocation Using JOBCAT and STEPCAT DD Statements" on page 11 contains a description of the DD statements required for user catalogs.

For a Non-VSAM Data Set

Methods of describing and allocating non-VSAM data sets are shown in the DD statements of the examples provided with the following commands: BLDINDEX, EXPORT, IMPORT, REPRO, PRINT,EXPORTRA, and IMPORTRA.

For a Snap Dump

If access method services encounters a condition that requires it to abort a job, it takes a snap dump of virtual storage. An AMSDUMP DD statement is required to obtain the snap dump; for example,

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

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

For a Target Data Set

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

- Record format: VBA
- Logical record length: 125, that is, 121+4
- Block size: 629, that is, (5x(121+4))+4

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

It is possible to alter the above defaults through specification of the desired values in the DCB parameter of the SYSPRINT statement. The record format, however, cannot be specified as F or FB. If you do specify either one, it is changed to VBA.

For an Alternate Target Data Set

In several commands you have the option of specifying an alternate target data set for listing. If you do specify an alternate, you must specify DCB parameters in the referenced DD statement by using the attributes described above.

When specifying an alternate target data set, you should not specify F or FB record formats.

Note: JCL statements, system messages, and job statistics are written to the SYSPRINT output device, not to the alternate target data set. Appendix B, "Interpreting LISTCRA Output Listings" on page 301 shows an example of the use of an alternate target data set.

DIRECT ALLOCATION USING JOBCAT AND STEPCAT DD STATEMENTS

User catalogs can be allocated directly by using a JOBCAT or STEPCAT DD statement.

You can specify a catalog that is to be available for use only with a job step, or that is to be available for the entire job. A STEPCAT DD statement identifies a catalog that is available for a single job step; a JOBCAT DD statement identifies a catalog that is available during the entire job. When both JOBCAT and STEPCAT catalogs are specified, the user catalog specified by STEPCAT is used for the job step. The JOBCAT and STEPCAT statements you code should be in the form:

```
//JOBCAT   DD  DISP={OLD|SHR},
//             DSNAME=usercatname
```

and

```
//STEPCAT  DD  DISP={OLD|SHR},
//             DSNAME=usercatname
```

Because the master catalog contains information about each user catalog, you do not need to specify the catalog's volume and device information.

You can specify more than one user catalog for a job or job step. To indicate another user catalog, follow the JOBCAT or STEPCAT DD statement with a concatenated DD statement—an unlabeled DD statement that specifies the name of another user catalog:

```
//STEPCAT DD DISP=SHR,DSN=MGMTCAT1
//        DD DISP=SHR,DSN=MGMTCAT2
```

Supplying a JOBCAT or STEPCAT DD statement to allocate the catalog is optional when:

- Your job processes a user catalog (that is, when your sequence of access method services commands adds, modifies, lists, or deletes a catalog entry)
- Your job includes a space allocation request directed toward a volume that contains VSAM data spaces (that is, a DEFINE SPACE or DELETE SPACE command)
- Your job causes a VSAM data set to be opened and that data set is defined in a user catalog (that is, a REPRO or PRINT of a VSAM data set, a DELETE with the ERASE option, EXPORT, IMPORT, IMPORTRA, BLDINDEX, or VERIFY)

  For information on REPRO and JOBCAT/STEPCAT DD statements, see Catalog Administration Guide.

Supplying a JOBCAT or STEPCAT DD statement to allocate the catalog is required when:

- Your job causes the catalog to be opened as a data set (for example, using the PRINT command to print a catalog, using the LISTCRA command with the COMPARE option, or using the VERIFY command to restore end-of-file indicators of a user catalog).
- Your job causes the scheduler allocation of the user catalog to be bypassed because unit and volume information was specified on the DD statement describing the VSAM data set defined in that user catalog.

INVOKING ACCESS METHOD SERVICES

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

There are three ways you can invoke the access method services program:

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

You can execute the IDCAMS program 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 execute access method services functional commands from the TSO terminal as though they were TSO commands. (See Appendix E, "Invoking Access Method Services from a Problem Program" on page 319, for information on invoking access method services.)

For more details on the structure and operation of access method services, see Access Method Services Logic.

AS A JOB OR JOBSTEP

You can use job control language (JCL) statements to invoke access method services. The examples that illustrate each command's use invoke access method services with JCL statements. You identify the access method services program with PGM=IDCAMS.

```
//YOURJOB  JOB  YOUR INSTALLATION'S JOB=ACCOUNTING DATA
//JOBCAT   DD   DSNAME=YOUR.CATALOG,DISP=SHR
//STEP1    EXEC PGM=IDCAMS
//STEPCAT  DD   DSNAME=ANOTHER.CATALOG,DISP=SHR
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *

 (access method services commands and their parameters)

/*
```

- //YOURJOB is required. The JOB statement describes your job to the system. Your installation might require you to supply user identification, accounting, and authorization information with the JOB statement's parameters.
- //JOBCAT is optional. The JOBCAT DD statement identifies a user catalog that can be used by each of the job's steps. Because the master catalog is always open and available to all jobs on the system, you should not identify it with a JOBCAT DD statement.
- //STEP1 is required (that is, an EXEC statement is required). The EXEC statement invokes access method services to decode and process the input statements (that is, 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 access method services. See the description of the PARM command in Chapter 3, "Controlling Command Execution" on page 21, for an explanation of the options you can specify.
- //STEPCAT is optional. The STEPCAT DD statement identifies a user catalog that can be used when processing the jobstep. If user catalogs are identified with JOBCAT and STEPCAT DD

statements, only the catalog(s) identified with the STEPCAT DD statement and the master catalog are used with the jobstep. Because the master catalog is always open and available to all jobs on the system, you should not identify it with a STEPCAT DD statement.

- //SYSPRINT is required. The SYSPRINT DD statement identifies the output device (usually a printer in a batch job) to which access method services sends messages and output information. (See "For a Target Data Set" on page 10 for more details on how to describe an output device other than SYSOUT=A.)

- //SYSIN is required. The SYSIN DD statement identifies 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 immediately following statements as input.

The last input statement may be followed by a delimiter statement that has "/*" in the first two columns. The restrictions that apply to all access method services commands are described in "How to Code Access Method Services Commands" on page 3.

FROM A TSO TERMINAL

TSO is a subsystem that provides conversational time sharing from remote terminals. You can use TSO with VSAM and access method services to:

- Execute access method services commands
- Execute a program to call access method services

When you use TSO to process your data, you can invoke access method services from your TSO terminal. Each time you issue an access method services command as a TSO command, TSO builds the appropriate interface information and invokes access method services.

You can issue one command at a time. Access method services processes the command completely before TSO allows you to continue processing. Except CHKLIST, all the access method services functional commands referred to in this book are supported in a TSO environment.

When TSO is used with access method services, the following differences must be observed:

- TSO allows the initial characters of a keyword to be supplied as an abbreviation of the keyword. The only restriction is that enough initial characters must be supplied to make the keyword unique. TRACKS, for example, could be abbreviated TR, TRA, or TRAC, because no other keyword within the same command can be abbreviated in the same way.

  Some abbreviations that are acceptable to access method services, such as CYL, CYLINDER, REC, RECORD, cannot be used under TSO because the abbreviations do not contain enough initial characters to make the keyword unique. TSO cannot tell whether you mean CYLINDERS or CYLINDERFAULT when CYL or CYLINDER is used. Similarly, abbreviations REC and RECORD are ambiguous; TSO does not know if you mean RECORDS or RECORDSIZE.

- In addition, the abbreviations described in the remainder of this publication (for example, TRK for TRACKS) are acceptable to TSO.

- When a parameter's value consists of a list of 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))
```

- Under TSO, a volume serial number can consist of alphabetic, national, numeric, or special (hyphen only) characters; if any other characters are used, the volume serial number cannot be used as an entryname under TSO.

- A name can be specified in quotation marks or not in quotation marks. Under TSO, however, a prefix (for example, the userid) is added to a name that is not in quotation marks. The prefix becomes the first qualifier in the name. If the name is a volume serial number (as it may be in the DELETE command, for example), enclose it in quotation marks; if you do not, a prefix is added to the volume serial number.

- Under TSO, a password can be supplied with any entryname; the password is ignored if not required.

  Note: At a TSO terminal, the logon password is checked first, before the user is prompted to supply a password for a cluster. Checking the logon password counts as one attempt to obtain a password. If the user has not specified ATTEMPTS in the DEFINE command, only one attempt to supply the catalog's password is allowed because the default is two.

- Under TSO, the user is prompted to complete a fully qualified name. The user is also prompted to supply required, but omitted, parameters.

- The modal commands, used to control execution (IF-THEN-ELSE command sequence, DO-END command sequence, SET, and PARM), are not allowed under TSO.

- Under TSO, the SYSPRINT data set is not used. The OUTFILE parameter may be used to specify a data set to receive the output procedure from access method services.

For details about the format of a displayed catalog entry (resulting from a LISTCAT command) for a TSO user, see Appendix A, "Interpreting LISTCAT Output Listings" on page 256.

Other TSO restrictions are noted with the descriptions of each appropriate parameter.

For details about writing and executing programs and allocating data sets with TSO, see TSO Terminal User's Guide and TSO Command Language Reference.

FROM A USER'S PROGRAM

A processing program can invoke access method services with the ATTACH, LINK, LOAD, and CALL macros. Before the program issues the invoking macro, however, it must initialize the appropriate register and argument list contents.

The contents of registers follow standard linkage conventions; that is, 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, and register 15 contains the address of the entry point IDCAMS in access method services.

The contents of the argument list are described in Appendix E, "Invoking Access Method Services from a Problem Program" on page 319. For information on manipulating sensitive

data in a secured environment, see "Authorized Program Facility" in Catalog Administration Guide.

ORDER OF CATALOG USE

You can specify the catalog to be searched or selected for an entry with the CATALOG parameters, or with a JOBCAT DD or STEPCAT DD statement.

Order of Catalog Search for ALTER

The order in which catalogs are searched when an entry is to be located for ALTER is:

1. If a catalog is specified in the CATALOG parameter, only that catalog is searched. If the entry is not found, a "no entry found" error is returned to the user.

2. Any user catalog(s) specified 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 below.

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

3. If the entry is identified with a qualified entryname, and it is not of the generic form, and its first qualifier is the same as:

   - The name of a user catalog, or
   - The alias of a user catalog, or
   - The alias of an OS CVOL,

   the user catalog or OS CVOL so identified is searched. If the entry is found, no other catalog is searched. Otherwise,

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 to the user.

Order of Catalog Selection for BLDINDEX

The order in which catalogs are selected to contain work file entries is:

1. If a catalog is specified with the CATALOG parameter, that catalog is selected to contain work file entries.

2. Otherwise, the user catalog specified in the current job step (STEPCAT) or, if none is specified for the job step, the user catalog specified for the current job (JOBCAT) is selected to contain the work file entries. If more than one catalog is specified for the job step or job, the first STEPCAT or JOBCAT catalog is selected to contain the work file entries.

3. Otherwise, if the entries (data set name on the DD statement) are identified with qualified entrynames and the first qualifier is the same as:

   - The name of a user catalog, or
   - The alias of a user catalog,

   the user catalog so identified is selected to contain the work file entries.

4. Otherwise, the master catalog is selected to contain the work file entries.

Catalog Selection for CNVTCAT

The target catalog selected to receive the converted entries is the catalog specified in the OUTFILE, OUTDATASET, or CATALOG parameter.

The source catalog, the catalog that contains the entries to be converted, is the catalog specified in the INFILE or INDATASET parameter.

If there are alias entries for OS CVOLs, the master catalog always receives alias entries that point to OS CVOLs.

Order of Catalog Selection for DEFINE

The order in which catalogs are selected when an entry is defined is:

1. If a catalog is specified in the CATALOG parameter, that catalog is selected to contain the entry to be defined.
2. When a non-VSAM generation data group (GDG) data set is defined, the catalog containing the GDG base is selected to contain the non-VSAM entry still to be defined. Otherwise,
3. The first user catalog specified 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 entry to be defined. ("Direct Allocation Using JOBCAT and STEPCAT DD Statements" on page 11 describes how you can specify a user catalog for the current job step or current job.) Otherwise,
4. If no user catalog is specified for the current job step or job, and the entry's name is a qualified name, and the first qualifier (that is, the first one to eight characters before the period) is the same as:

   - The name of a user catalog, or
   - The alias of a user catalog,

   the catalog so identified is selected to contain the entry to be defined.

5. If no catalog has been identified, either explicitly or implicitly, VSAM defines an object in the master catalog.

Order of Catalog Search 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 specified in the CATALOG parameter, only that catalog is searched. If the entry is not found, a "no entry found" error is returned to the user. Otherwise,

2. Any user catalog(s) specified 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 below.

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

3. If the entry is identified with a qualified entryname and its first qualifier is the same as:

   - The name of a user catalog, or
   - The alias of a user catalog, or
   - The alias of an OS CVOL,

   the user catalog or OS CVOL so identified is searched. If the entry is found, no other catalog is searched. Otherwise,

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 to the user.

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 specified 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 to the user. Otherwise,

2. Any user catalog(s) specified 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. The JOBCAT catalog is not searched. The catalog search continues with step 3 below.

   If no STEPCAT catalog is specified for the job step, and a user catalog is specified for the current job (with a JOBCAT DD statement), the JOBCAT catalog(s) is searched. If more than one catalog is specified for the job, the catalogs are searched in order of concatenation. The catalog search continues with step 3 below.

3. If the entry is identified with a qualified entryname and its first qualifier is the same as:

   - The name of a user catalog, or
   - The alias of a user catalog, or
   - The alias of an OS CVOL,

the user catalog or OS CVOL so identified is searched. The catalog search continues with step 4 below. Otherwise,

4. The master catalog is searched. If an entry has not been found in any of the catalogs searched that matches the supplied qualifiers, a "no entry found" error is returned to the user.

CAUTION: If the generic form of entryname is used to delete catalog entries, and the catalog is not specified via the CATALOG parameter, entries may be deleted from the master catalog if they meet the specified qualification.

Order of Catalog Search for LISTCAT

When the ENTRIES parameter is not specified, or when the command is not executed 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. Otherwise,

2. The first user catalog specified in the current job step (STEPCAT) or, if none is specified, the first user catalog specified in the current job (JOBCAT) is listed. Otherwise,

3. If no user catalog is specified in the current job step or job, the master catalog is listed.

If the command is not a generic LISTCAT and if the ENTRIES or LEVEL parameter is specified or when the command is executed 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 specified in the CATALOG parameter, only that catalog is searched. If the entry is not found, a "no entry found" error is returned to the user. Otherwise,

2. Any user catalog(s) specified in the current job step (STEPCAT) or, if none is specified for the job step, any user catalog(s) specified for the current job (JOBCAT). 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. If the entry is found, no other catalog is searched. Otherwise,

3. If the entry is not found, and the entry's name is a qualified name, and the first qualifier (that is, the first 1 to 8 characters before a period) is the same as:

   - The name of a user catalog, or
   - The alias of a user catalog, or
   - The alias of an OS CVOL,

   that user catalog or OS CVOL is searched. If the entry is found, no other catalog is searched. Otherwise,

4. The master catalog is searched. If the entry is not found, a "no entry found" error is returned to the user.

When the ENTRIES parameter is specified 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 specified 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 to the user. Otherwise,

2. Any user catalog(s) specified in the current job step (STEPCAT) or, if none is specified for the job step, any user catalog(s) specified for the current job (JOBCAT). 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 below. Otherwise,

3. If the entry's name is a qualified name, and the first qualifier (that is, the first 1 to 8 characters before a period) is the same as:

   - The name of a user catalog, or
   - The alias of a user catalog, or
   - The alias of an OS CVOL,

   that user catalog or OS CVOL is searched. The catalog search continues with step 4 below. Otherwise,

4. The master catalog is searched. If an entry has not been found in any of the catalogs search that matched the supplied qualifiers, a "no entry found" error is returned to the user.

MASS STORAGE SYSTEM (MSS)

The IBM 3850 Mass Storage System can be used to store a massive amount of data online to the operating system. It is described in Introduction to the IBM 3850 Mass Storage System (MSS).

When you have the Mass Storage System, you can define VSAM data spaces, user catalogs, and data sets and non-VSAM data sets on mass storage volumes. The master catalog cannot be stored on a mass storage volume.

A catalog may have defined in it both objects stored on direct access storage volumes and objects stored on mass storage volumes. In particular, the data component of a key-sequenced cluster may be stored on a mass storage volume and the index component on a direct access storage volume, or vice versa.

Space for an object larger than one cylinder that is stored on a mass storage volume should be allocated in cylinders to optimize data transfer (staging and destaging) between mass storage and direct access storage.

Data stored in the Mass Storage System is staged from mass storage to a direct access storage staging drive when the object to which the data belongs is opened or when the data is requested. It is destaged from the staging drive to mass storage when the object is closed. (Introduction to the IBM 3850 Mass Storage System (MSS) tells which direct access storage devices can be used for staging.)

Access method services for the Mass Storage System provides a set of commands for the management of mass storage volumes. These commands are described in OS/VS Mass Storage System (MSS) Services: Reference Information.

Access method services for managing catalogs provides parameters for options of the Mass Storage System in the DEFINE and ALTER commands, which enable you to specify how a VSAM data set that

is stored on a mass storage volume is to be staged and destaged and how a user catalog that is stored on a mass storage volume is to be destaged. These parameters are described in this book.

The staging attributes, BIND, CYLINDERFAULT, and STAGE, affect performance. BIND causes an object to be staged when it is opened and to be retained (bound) in direct access storage. CYLINDERFAULT causes portions of an object to be staged only as needed during processing. STAGE is a compromise: the object is staged when it is opened, but not retained in direct access storage. When the staging activity of other objects is light, STAGE can achieve results similar to BIND: the data might remain in direct access storage, available for requests for access without staging. When the staging activity of other objects is heavy, STAGE can achieve results similar to CYLINDERFAULT: the data might not remain in direct access storage and might have to be restaged when needed.

A user catalog that is stored on a mass storage volume is always staged and bound when it is opened—that is, it is retained in direct access storage until it is closed. Not binding a user catalog might degrade performance.

For recoverable catalogs, a catalog recovery area (CRA) is opened when it is first needed, and remains open for as long as the user catalog is open. Because of this, the CRA volume must remain mounted during that period of time.

The destaging attributes, DESTAGEWAIT and NODESTAGEWAIT, affect data integrity. DESTAGEWAIT causes VSAM to return control to the program that closes an object synchronously—only after destaging is complete. VSAM can notify the program whether destaging was successful. NODESTAGEWAIT causes VSAM to return control to the program asynchronously—as soon as the object is closed, but before it has been destaged.

A failure in destaging causes a message to be written to the operator. With DESTAGEWAIT, an error code is also returned from CLOSE to the processing program. But there are no recovery procedures for a program to undertake: One use of DESTAGEWAIT is for a processing program to periodically issue a temporary CLOSE of a bound object to cause it to be destaged at various checkpoints. The processing program can terminate processing if a failure occurs in destaging. The last copy destaged would be the copy to fall back to, pending correction of the error that caused the failure.

A data component that is defined with the ERASE parameter and stored on a mass-storage volume, is overwritten with binary zeros on the staging drive after it is destaged.

Note: On mass storage volumes, a GDG data set may not be scratched when it is automatically uncataloged, even if the SCRATCH option was specified in the DEFINE.

CHAPTER 3. CONTROLLING COMMAND EXECUTION

This chapter describes:

- IF-THEN-ELSE command sequence, which is used to control command execution on the basis of condition codes
- DO-END command sequence, which specifies more than one functional access method services command and its parameters
- NULL command, which specifies that no action is to be taken
- SET command, which is used to reset condition codes
- PARM command, which is used to specify diagnostic aids and printed output options

These commands cannot be used when access method services is being executed under TSO.

CONDITION CODES

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

Code Explanation

0 Indicates that the function was executed as directed and expected. Some informational messages might be issued.

4 Indicates that some problem was met in executing the complete function, but it was possible to continue. The continuation might not provide the user with exactly what is wanted, but no permanent harm will have been done by such continuation. A warning message was issued. An example of the kind of problem encountered is: The system was unable to locate an entry in a LISTCAT command.

8 Indicates that 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 was terminated.

12 Indicates that the requested function could not be performed. This condition code is set as a result of a logical error. A logical error condition exists when inconsistent parameters are specified, when required parameters are missing, or when a value specified 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 VSAM Administration: Macro Instruction Reference.

16 Indicates that a severe error occurred that caused the remainder of the command stream to be flushed. This condition code might result in the following examples: A system output data set cannot be opened (a SYSPRINT DD statement was missing, for example); an unrecoverable error occurred in a system data set; or 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.

IF-THEN-ELSE COMMAND SEQUENCE

The IF-THEN-ELSE command sequence is used to control command execution.

The format of the IF-THEN-ELSE command sequence is:

IF	{LASTCC|MAXCC} comparand number THEN[command] DO command set END] [ELSE[command] DO command set END]]

where:

IF
specifies that one or more functional commands are to be executed based on a test of a condition code. The condition code is set by a SET command or is set to reflect the completion status of previous functional commands.

LASTCC
specifies that the condition code value that resulted from the immediately preceding function command is to be compared, as indicated by the comparand, to the number following the comparand. For the meaning of condition codes, see "Condition Codes."

MAXCC
specifies that the maximum condition code value that has been established by any previous function command or by a SET command is to be compared, as indicated by the comparand, to the number following the comparand to determine whether the THEN action is to be performed. For the meaning of condition codes, see "Condition Codes."

comparand
specifies the comparison to be made between the variable and the following number. There are six possible comparisons:

Equal, specified as '=' or 'EQ' Not equal, specified as '¬=' or 'NE'

Greater than, specified as '>' or 'GT'

Less than, specified as '<' or 'LT'

Greater than or equal, specified as '>=' or 'GE'

Less than or equal, specified as '<=' or 'LE'

number
specifies the decimal integer that is to be compared with MAXCC or LASTCC. The number can be up to 10 digits long. Values greater than 16 are reduced to 16; both LASTCC and MAXCC are initialized to zero upon entry to access method services. For the meaning of condition codes, see "Condition Codes."

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

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

When an IF command appears in a THEN or ELSE clause, it is called a nested IF command. The maximum level of nesting allowed is 10, starting with the first IF specified.

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. (To say it another way, each ELSE is matched with the nearest preceding unmatched THEN.) Should there be 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.

DO-END COMMAND SEQUENCE

DO
specifies that the group of commands that follow is to be treated as a single unit, that is, to be executed as a result of a single IF command. The set of commands is terminated by END. A command following a DO must begin on a new line.

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

Note: Continuation characters should not be used in the DO-END sequence; they may be taken as a null command or cause unpredictable results.

NULL COMMANDS

If THEN or ELSE is not followed by 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. The null command is, in essence, a THEN or ELSE command that is not followed by a command continuation character.

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

```
ELSE
```

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

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

The null command is used 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).

SET COMMAND

The SET command is used to change or reset a previously defined condition code. (For the meaning of condition codes, see "Condition Codes.") It is possible to terminate all processing merely by setting MAXCC or LASTCC to 16.

The format of the SET command is:

SET	{MAXCC|LASTCC}=number

where:

SET
specifies that a condition code value is to be set. A SET command that follows a THEN or ELSE that is not executed does not cause the value of LASTCC or MAXCC to be altered.

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

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

number
specifies the value to be assigned to MAXCC or LASTCC. The maximum value that can be assigned is 16; a greater value will be reduced to 16. If the value assigned to LASTCC is greater than the value of MAXCC, MAXCC is set equal to the higher value.

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 specify these options in the PARM field of an EXEC statement (in the JCL).

The format of the PARM command is:

PARM	[TEST({[TRACE] [AREAS(areaid[areaid...])] [FULL((dumpid[countl[count2]]) [(dumpid...)...])]| OFF})] [GRAPHICS(CHAIN(chain)|TABLE(mname))] [MARGINS(leftmargin rightmargin)]

where:

TEST(
{[TRACE]
[AREAS(areaid**[** areaid...**])]**
[FULL((dumpid**[** countl**[** count2**]])**
[(dumpid...**)...])]|**
OFF})
specifies the diagnostic aids to be used. After the TEST option has been established, it remains in effect until it is reset by another PARM command. The TRACE, AREAS, and FULL parameters may be used concurrently.

TRACE
specifies that trace tables are to be listed whenever the built-in dump points of the processor are encountered.

AREAS(areaid**[** areaid...**])**
identifies modules that are to have selected variables dumped at their dump points. Each item in the areaid is a 2-character area identifier defined within the implementation. For more information, see Access Method Services Logic.

FULL([dumpid[count1[count2]]][(dumpid...)...])
specifies that a region dump, as well as the trace tables and selected variables, is to be provided at the specified points. dumpid specifies the 4-character identifier of the dump point. For more information, see Access Method Services Logic.

count1
is a decimal integer that specifies the number of times the program is to go through the dump point before beginning the dump listing. (Default is 1.)

count2
is a decimal integer that specifies the number of times through the dump point that dumps are to be listed. (Default is 1.)

If the FULL keyword is used, an AMSDUMP DD statement must be provided. For example:

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

OFF
specifies that testing is to stop.

GRAPHICS[CHAIN(chain)|TABLE(mname)]
specifies the print chain graphic character set or a special graphics table to be used in producing the output.

CHAIN(AN|HN|PN|QN|RN|SN|TN)
specifies the graphic character set of the print chains you want to use. PN is used by the processor unless explicitly directed to use another set of graphics.

TABLE(mname)
specifies the name of a user-supplied table. This 256-byte table defines the graphics for each of the 256 possible bit patterns. Any character to be printed is translated to the bit pattern found in such a table at the position corresponding to its numeric value (0 through 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 (VS).

MARGINS(leftmargin rightmargin)
specifies that the margins of input records on which command statements are written are to be changed. The normal left and right margins are 2 and 72, respectively. If MARGINS is coded, all subsequent input records are scanned in accordance with the new margins. This feature may be used in conjunction with the comment feature: Respecification of margins could be used to cause the /* and */ characters to be omitted from the scan and thus cause comments to be treated as commands.

leftmargin
specifies the location of the left margin.

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

COMMON CONTINUATION ERRORS IN CODING CONTROL COMMANDS

The continuation rules must be used cautiously when modal commands appear in the input stream. (See "How to Continue Commands and Parameters" on page 7.) 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 executed.

- ```
  IF LASTCC = 0 -
  THEN -
  REPRO ...
  /*ALTERNATE PATH*/
  ELSE -
  PRINT ...
  ```

  Because no continuation mark (hyphen) follows the comment, a null command is assumed. The ELSE keyword will not match up with 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 executed.

- ```
  PARM TEST ( - /*COMMENT*/
  TRACE)
  ```

  Because characters other than blanks appear between the continuation mark (hyphen) and the end of the record, the PARM command will not be continued onto the second record.

- ```
  PARM TEST ( TRA+
  /*FIELD CONTINUATION*/
  CE)
  ```

  Because no continuation was indicated, the end of the PARM command is found after the second record. The command is rejected.
  ```

CONTROL COMMAND EXECUTION EXAMPLES

The examples that follow show the use of the IF-THEN-ELSE command sequence, the SET command, and the PARM command.

Control Command Execution with Nested IF Commands: Example 1

In this example, nested IF commands are used to determine whether a REPRO, DELETE, or PRINT command is to be executed.

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

This example specifies that, if the value of LASTCC is greater than 4, the value of MAXCC is to be tested. If the value of MAXCC is less than 12, the REPRO command is executed; if the value of MAXCC is 12 or greater, the DELETE command is executed instead. Again, if the value of LASTCC is 4 or less, LASTCC is tested for being exactly 4: No action is to be taken in this case. If, however, LASTCC is less than 4, the PRINT command is to be executed.

Control Command Execution with Nested IF Commands: Example 2

In this example, nested IF commands are used to determine whether a REPRO or PRINT command is to be executed.

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

Should the first IF command determine that LASTCC is greater than 4, and the second IF command determine that MAXCC is 12 or greater, no functional commands will be executed. The null ELSE command is employed here to specify that the next ELSE is to correspond to the first THEN.

Control Command Execution Using MAXCC Parameter: Example 3

In this example, if the maximum condition code is 0, an entry from a catalog is listed and a data set is printed.

```
IF MAXCC=0 THEN DO
  LISTCAT  CATALOG (AMASTCAT/MST27) ENT (MN01.0005)
  PRINT INFILE (AJK006)
  END
  ELSE ...
```

Control Command Execution Using LASTCC Parameter: Example 4

If you want to list a catalog and print a data set if the last condition code is 0, but want to list its catalog entry before and after a VERIFY command if the last condition code is greater than 0, specify:

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

Control Command Execution with LASTCC Value Established: Example 5

If you want to set the last condition code established to 12, specify:

```
SET LASTCC=12
```

Control Command Execution Replacing MAXCC Value: Example 6

If you want to replace the highest condition code established in processing so far with 8, specify:

```
SET MAXCC=8
```

CHAPTER 4. FUNCTIONAL COMMAND FORMAT

This chapter contains the access method services functional commands for VSAM catalogs, OS CVOLs, and for VSAM and non-VSAM data sets. The commands are arranged alphabetically, with a summary of the required and optional parameters following each command section. The required parameters are arranged in alphabetic order, with a discussion of the parameter and an abbreviation following. The optional parameters are in the same format and follow the required parameters.

Examples of how each command is used are provided at the end of each command.

For an explanation of the symbols used in the command formats, see "Notational Conventions" on page vi. See "How to Code Subparameters" on page 5 for coding conventions that apply to the subparameters used in access method services commands.

FUNCTIONAL COMMAND FORMAT SUMMARY

This section provides reference information about the following functional commands.

- ALTER, which is used to alter attributes of data sets and catalogs that have already been defined.
- BLDINDEX, which builds alternate indexes for existing data sets.
- CHKLIST, which lists tape data sets opened during a checkpoint.
- CNVTCAT, which converts OS CVOL entries into VSAM catalog entries.
- DEFINE, which defines the following objects:
  - DEFINE ALIAS, which defines an alternate name for a non-VSAM data set or a user catalog.
  - DEFINE ALTERNATEINDEX, which defines an alternate index.
  - DEFINE CLUSTER, which defines a cluster for a key-sequenced, entry-sequenced, or relative record data set.
  - DEFINE GENERATIONDATAGROUP, which creates a catalog entry for a generation data group.
  - DEFINE NONVSAM, which defines a catalog entry for a non-VSAM data set.
  - DEFINE PAGESPACE, which defines an entry for a page space data set.
  - DEFINE PATH, which defines a path directly over a base cluster or over an alternate index and its related base cluster.
  - DEFINE SPACE, which defines a VSAM data space.
  - DEFINE USERCATALOG|MASTERCATALOG, which defines a user catalog.

- DELETE, which deletes catalogs, VSAM data sets, and non-VSAM data sets.
- EXAMINE, which is used to analyze and report on the structural consistency of the index component and/or data component of a key-sequenced data set cluster.
- EXPORT, which disconnects user catalogs and exports VSAM data sets.
- EXPORTRA, which recovers VSAM and non-VSAM catalog entries from catalog recovery areas, and recovers data for VSAM objects (clusters and alternate indexes) by means of catalog recovery areas.
- IMPORT, which connects user catalogs and imports VSAM data sets.
- IMPORTRA, which reconstructs multiple VSAM data sets from a data set created by the EXPORTRA command.
- LISTCAT, which lists catalog entries.
- LISTCRA, which lists or compares the contents of a given catalog recovery area.
- PRINT, which is used to print VSAM data sets, non-VSAM data sets, and catalogs.
- REPRO, which is used to copy VSAM and non-VSAM data sets, to copy catalogs, and to unload and reload VSAM catalogs.
- RESETCAT, which compares catalog entries with catalog recovery area (CRA) entries in order to regain access to catalog data.
- VERIFY, which causes a catalog to correctly reflect the end of a data set after an error occurred while closing a VSAM data set. The error may have caused the catalog to be incorrect.

ALTER

The ALTER command modifies the attributes of previously defined catalog entries. The format of this command is:

```
ALTER   entryname[/password]
        [ADDVOLUMES(volser[ volser...])]
        [ATTEMPTS(number)]
        [AUTHORIZATION(entrypoint[ string])]
        [BUFFERSPACE(size)]
        [CODE(code)]
        [CONTROLPW(password)]
        [DESTAGEWAIT|NODESTAGEWAIT]
        [EMPTY|NOEMPTY]
        [ERASE|NOERASE]
        [EXCEPTIONEXIT(entrypoint)]
        [FILE(ddname)]
        [FREESPACE(CI-percent[ CA-percent])]
        [INHIBIT|UNINHIBIT]
        [KEYS(length offset)]
        [MASTERPW(password)]
        [NEWNAME(newname)]
        [NULLIFY(
          [AUTHORIZATION(MODULE|STRING)]
          [CODE]
          [CONTROLPW]
          [EXCEPTIONEXIT]
          [MASTERPW]
          [OWNER]
          [READPW]
          [RETENTION]
          [UPDATEPW])]
        [OWNER(ownerid)]
        [READPW(password)]
        [RECORDSIZE(average maximum)]
        [REMOVEVOLUMES(volser[ volser...])]
        [SCRATCH|NOSCRATCH]
        [SHAREOPTIONS(crossregion[ crosssystem])]
        [STAGE|BIND|CYLINDERFAULT]
        [TO(date)|FOR(days)]
        [UNIQUEKEY|NONUNIQUEKEY]
        [UPDATE|NOUPDATE]
        [UPDATEPW(password)]
        [UPGRADE|NOUPGRADE]
        [WRITECHECK|NOWRITECHECK]

        [CATALOG(catname[/password])]
```

Entry Types That Can Be Altered

An "X" in Figure 2 on page 32 indicates you can alter the value or attribute for the type of catalog entry specified. KEYS and RECORDSIZE, when specified for a cluster or alternate index, apply to its data component entry only, and not to the cluster or alternate index entry. Some attributes can be specified only for the cluster's or alternate index's data or index component entry—you must identify the component with its entryname. You can use the LISTCAT command to determine the names generated for the object's components.

When you identify a group of entries with a generic name, entries whose entrynames match the supplied qualifiers are altered if they contain the type of information specified with the ALTER command's other parameters.

You cannot alter the following types of entries: alias entries, a catalog's data or index component entries, or a master catalog's self-describing entries.

Attributes that can be altered	Type of Catalog Entry: ALTERNATE IX	AIX DATA	AIX INDX	CLUSTER	CL DATA	CL INDX	PGSPC	PATH	USERCATLG	NON VSAM	GDG
ADDVOLUME		X	X		X	X					
ATTEMPTS	X	X	X	X	X	X	X	X	X		
AUTHORIZATION	X	X	X	X	X	X	X	X	X		
BIND		X	X		X	X					
BUFFERSPACE		X			X				X		
CODE	X	X	X	X	X	X	X	X	X		
CONTROLPW	X	X	X	X	X	X	X	X	X		
CYLINDERFAULT		X	X		X	X					
DESTAGEWAIT		X	X		X	X					
EMPTY											X
ERASE		X			X						
EXCEPTIONEXIT		X	X		X	X					
FOR	X			X				X	X	X	X
FREESPACE		X			X						
INHIBIT		X	X		X	X					
KEYS	X	X		X	X						
MASTERPW	X	X	X	X	X	X	X	X	X		
NEWNAME	X	X	X	X	X	X	X	X		X	
NODESTAGEWAIT		X	X		X	X					
NOEMPTY											X
NOERASE		X			X						
NOUNIQUEKEY		X									
NOSCRATCH											X
NOUPDATE								X			
NOUPGRADE	X										
NOWRITECHECK		X	X		X	X					

Figure 2 (Part 1 of 2). ALTER Parameters and the Entry Types to Which Each Applies

Attributes that can be altered	ALTERNATE IX	AIX DATA	AIX INDX	CLUSTER	CL DATA	CL INDX	PGSPC	PATH	USERCATLG	NON VSAM	GDG
NULLIFY	X	X	X	X	X	X	X	X	X	X	X
AUTHORIZATION	X	X	X	X	X	X	X	X	X		
CODE	X	X	X	X	X	X	X	X	X		
CONTROLPW	X	X	X	X	X	X	X	X	X		
EXCEPTIONEXIT		X	X		X	X					
MASTERPW	X	X	X	X	X	X	X	X	X		
OWNER	X	X	X	X	X	X	X	X	X	X	X
READPW	X	X	X	X	X	X	X	X	X		
RETENTION	X			X			X	X	X	X	X
UPDATEPW	X	X	X	X	X	X	X	X	X		
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					
SCRATCH											X
SHAREOPTIONS		X	X		X	X					
STAGE		X	X		X	X					
TO	X			X			X	X	X	X	X
UNINHIBIT		X	X		X	X					
UNIQUEKEY		X									
UPDATE								X			
UPDATEPW	X	X	X	X	X	X	X	X	X		
UPGRADE	X										
WRITECHECK		X	X		X	X					

Figure 2 (Part 2 of 2). ALTER Parameters and the Entry Types to Which Each Applies

ALTER PARAMETERS

Required Parameters

entryname[/password]
names the entry to be altered and supplies a password.

When attributes of a catalog are to be altered, entryname must specify the catalog name.

For a description of how to specify a generic name in order to alter multiple entries with one ALTER command, see Catalog Administration Guide. See this guide also for a description of how to alter data or index entrynames or cluster entry subparameters.

If you are renaming a member of a non-VSAM partitioned data set, the entryname must be specified in the format: pdsname(membername).

password
specifies the password of the entry or catalog to be altered.

If you are altering a password-protected entry in a password-protected catalog, you must specify a password. The password can be specified with the entryname or in the CATALOG parameter.

- Use the master password for the entry or for the catalog that contains the entry for an alternate index, cluster, page space, path, user catalog, data, or index. RACF: The alter RACF authority for an entry or for the catalog is required.
- If a data or index component entry is to be altered, use the master password of the cluster component or catalog. RACF: The alter RACF authority for a cluster component or a catalog is required.
- Use the update- or higher-level password of the catalog that contains the entry for non-VSAM and generation data group (GDG). RACF: The update or higher RACF authority to the catalog is required.

Optional Parameters

ADDVOLUMES(volser[volser])
specifies volumes to be added to the list of candidate volumes. When you specify ADDVOLUMES, the FILE parameter must also be specified to identify the volumes being added. The volumes to be added as candidate volumes must already be owned by the catalog that contains the entry being altered. (Space must have been defined on a volume to be added or the volume must have been identified as a candidate volume.)

Abbreviation: AVOL

ATTEMPTS(number)
specifies the maximum number of times the operator can try to enter a correct password in response to a prompting message.

This parameter only has effect when the entry's master password is not null. A prompting message is issued only if the user has not already supplied the appropriate password.

number
is an integer from 0 to 7 and can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

When ATTEMPTS(0) is coded, the operator is not prompted and is not allowed to enter a password from the console.

Note to TSO users: At a TSO terminal, the logon password is checked first, before the user is prompted to supply a password. Checking the logon password counts as one attempt to obtain a password. If ATTEMPTS is not specified, the user has one attempt to supply the password, because the default is 2.

Abbreviation: ATT

AUTHORIZATION(entrypoint[string]**)**
specifies that a user-security-verification routine (USVR) is available for additional security verification.

entrypoint
specifies the name of the USVR.

string
specifies information to be passed on to the USVR when it receives control to verify authorization.

Abbreviation: AUTH

BUFFERSPACE(size**)**
specifies the minimum space to be provided for buffers.

It is recommended that the amount specified be not less 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. BUFFERSPACE can be specified only for a catalog, or the data component of a cluster or alternate index.

size
specifies the amount of space to be provided for buffers. If you are altering the bufferspace for a VSAM catalog on a 3380, the decimal values you can specify are 4096, 6144, 8192, 10240, 12288, and 14336. If you are altering the bufferspace for a VSAM catalog not residing on a 3380, the decimal values you can specify are limited to 3072, 4608, 6144, 7680, 9216, and 10752.

size can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form, but must not exceed 16776704. If the size specified is less than the amount VSAM requires, VSAM gets the amount it requires when the data set is opened.

Abbreviations: BUFSP or BUFSPC

CATALOG(catname[/password]**)**
specifies the catalog containing the entry to be altered. See "Order of Catalog Search for ALTER" on page 15 for information about the order in which catalogs are searched.

catname
specifies the name of the catalog that contains the entry.

password
specifies the master password of the catalog that contains the entry to be altered. If the entry to be altered is password-protected and the catalog is also password-protected, a password must be entered, either

through this parameter or through the entryname parameter. RACF: The alter RACF authority to the catalog is required.

Abbreviation: CAT

CODE(code**)**
specifies a code name for the entry being altered. 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 cluster or component that is password protected without supplying a password, the operator or TSO terminal user is prompted with the name of the entry.

CONTROLPW(password**)**
specifies a control password for the entry being altered.

Abbreviation: CTLPW

DESTAGEWAIT|NODESTAGEWAIT
specifies whether a data or index component that is stored on a mass storage volume is to be destaged synchronously or asynchronously, with respect to the program that closes it.

These parameters can be specified only for the data or index component of a cluster, alternate index, or integrated catalog facility catalog.

DESTAGEWAIT
indicates that destaging is to be completed before control is returned from VSAM to the program that issues the CLOSE macro. VSAM can thus notify the program whether destaging was successful.

Abbreviation: DSTGW

NODESTAGEWAIT
indicates that notification of unsuccessful destaging is to be made only by a message to the operator.

Abbreviation: NDSTGW

EMPTY|NOEMPTY
specifies what is to happen when the maximum number of generation data sets has been cataloged.

EMPTY
specifies that, when the maximum number of generation data sets is exceeded, all the generation data sets will be uncataloged.

Abbreviation: EMP

NOEMPTY
specifies that when the maximum number of generation data sets is exceeded, only the oldest generation data set will be uncataloged.

Abbreviation: NEMP

ERASE|NOERASE
specifies whether the data component is to be erased when its entry in the catalog is deleted.

ERASE
specifies the component is to be overwritten with binary zeros when its catalog entry is deleted.

Abbreviation: ERAS

NOERASE
specifies the component is not to be overwritten with binary zeros when its catalog entry is deleted.

Abbreviation: NERAS

EXCEPTIONEXIT(entrypoint**)**
specifies the name of the user-written routine that receives control when 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 to be taken. The object's exception exit routine is processed first, then the user's SYNAD-exit routine receives control.

Note: You may not specify this parameter to add an exception-exit routine to a VSAM cluster that is cataloged in a VSAM catalog created in an OS/VS2 Release 2 (or lower release level) system.

Abbreviation: EEXT

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

- The name of a DD statement that identifies the volume whose catalog recovery area contains a copy of the catalog entry. For a description of catalog recovery area contents for each volume, see Catalog Administration Guide.
- The name of a DD statement that identifies the volume of an entry which will be renamed. The entry must be a non-VSAM data set or the data or index component of a unique cluster, alternate index, or page space.
- The name of a DD statement that identifies a partitioned data set when a member is to be renamed.

If multiple volumes of different device types are to be identified via FILE, you must use concatenated DD statements. If you specify ADDVOLUMES or REMOVEVOLUMES, the volume(s) being added or removed must be identified. If you specify NEWNAME for a non-VSAM data set or the data or index component of a unique cluster, alternate index, or page space that does not reside on the same volume as the catalog recovery area entry, that volume(s) must be identified.

If FILE is not specified, an attempt is made to dynamically allocate the catalog recovery area volume(s) or the object's volume. Therefore, the volume must be mounted as permanently resident or reserved.

FREESPACE(CI-percent**[** CA-percent**])**
specifies the amount of space (percentage of space) that is to be left free after any allocation and after any split of control intervals (CI-percent) and control areas (CA-percent).

The amounts, as percentages, must be equal to or less than 100. They may be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. If you specify 100% of free

space, one record is placed in each control interval and one control interval is placed in each control area.

This parameter may be specified to alter the data component of a cluster or alternate index.

If the free space is altered after the data set has been loaded, and sequential insert strategy processing is being used, this allocation of free space may not be honored.

Abbreviation: FSPC

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

INHIBIT
specifies that the entry being altered is to be read-only.

Abbreviation: INH

UNINHIBIT
specifies 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 entry being altered defines an alternate index, offset applies to the alternate key in the data records in the base cluster.

Restrictions: Keys can be specified only 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 that defines the object.

  Note: If the values for KEYS in the object's catalog entry are not default values, and ALTER KEYS specifies the same nondefault values, processing continues for any other parameters specified on the command, and no error message is issued.
- 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
specifies the length of the key (between 1 and 255), in bytes, and its displacement from the beginning of the data record, in bytes. You can express length and offset in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

MASTERPW(password)
specifies a master password for the entry being altered. For more details about the object's master password, see the DEFINE command that defines the object.

- If MASTERPW is not specified, the highest level password specified becomes the password for all higher levels.
- The AUTHORIZATION, CODE, and ATTEMPTS parameters have no effect unless the entry has a master password associated with it.
- With the master password, all operations are allowed.

Note: The master password must exist to alter lower-level passwords. It must be the last level of authority to be altered when nullifying all levels of authority.

Abbreviation: MRPW

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

To give the altered entry a new name:

- Unless the object being renamed is a path, the object's volume must be mounted because the volume's VTOC is modified.

  You can use the FILE parameter to supply a JCL DD statement to cause the object's volume(s) to be allocated. If you do not supply a DD statement, an attempt is made to dynamically allocate the object's volume(s). The volume(s) must be mounted as either permanently resident or reserved.
- If you specify generic names, you must specify both the entryname and the newname as generic names. For information on specifying generic names, see Catalog Administration Guide.
- 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 a data set profile exists for the new data set name prior to the ALTER command, the command is terminated and the data set name and/or protection attributes remain unchanged.

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

  If the rename fails, it is possible that the object exists with both the original and the new name.

Abbreviation: NEWNM

NULLIFY([AUTHORIZATION(MODULE|STRING)]
[CODE][CONTROLPW][EXCEPTIONEXIT]
[MASTERPW][OWNER][READPW]
[RETENTION][UPDATEPW])
specifies that the protection attributes identified by subparameters of NULLIFY are to be nullified. Attributes are nullified before any respecification of attributes is performed.

If all levels of passwords are nullified and none are respecified, CODE, AUTHORIZATION, and ATTEMPTS have no effect.

Abbreviation: NULL

AUTHORIZATION(MODULE|STRING)
specifies that either the user-authorization routine or the user-authorization record is to be nullified.

Abbreviation: AUTH

MODULE
specifies that the module name is to be removed 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
specifies that the authorization record is to be nullified, but the corresponding module is not nullified.

Abbreviation: STRG

CODE
specifies that the code name used for prompting is to be nullified.

CONTROLPW
specifies that the control password is to be nullified.

Abbreviation: CTLPW

EXCEPTIONEXIT
specifies that the entry's exception exit is to be nullified. The module name is removed from the catalog record, but the exception-exit routine itself is not deleted.

Abbreviation: EEXT

MASTERPW
specifies that the master password is to be nullified. If a new master password is not specified and if other passwords exist, the highest level password that exists automatically becomes the password for all higher levels, including the master password.

Abbreviation: MRPW

OWNER
specifies that the owner identification is to be nullified.

READPW
specifies that the read password is to be nullified.

Abbreviation: RDPW

RETENTION
specifies that the retention period which was specified in a TO or FOR parameter is to be nullified.

Abbreviation: RETN

UPDATEPW
specifies that the update password is to be nullified.

Abbreviation: UPDPW

OWNER(ownerid**)**
specifies the owner identification of the entry being altered.

READPW(password**)**
specifies a read password for the entry being altered.

Abbreviation: RDPW

RECORDSIZE(average maximum**)**
specifies new average and maximum lengths for data records that will be 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 can be specified only if all of 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 value for maximum RECORDSIZE in the object's catalog entry is the default value. For default values, see the DEFINE command that defines the object.

  Note: If the value for RECORDSIZE in the object's catalog entry is not the default value, and ALTER RECORDSIZE specifies the same nondefault value, processing continues for any other parameters specified on the command, and no error message is issued.
- For an alternate index, if NONUNIQUEKEY is specified, the record length to be specified accounts for the increased record size resulting from the multiple prime-key pointers in the alternate index data record.
- The maximum record length to be specified is at least seven bytes less than control interval size, unless the record is a spanned record.
- The record length to be specified is large enough to contain all prime and alternate keys previously defined.

Abbreviation: RECSZ

REMOVEVOLUMES(volser[volser]**)**
has two uses:

1. REMOVEVOLUMES specifies volume(s) to be removed from the list of candidate volumes associated with the entry being altered. If you are also adding volumes, the volumes to be removed are removed after the new volumes are added to the candidate list. The name of the data or index component must be specified in the entryname parameter.

   Note: If a volume to be removed contains data that belongs to the entry being altered, the volume is not removed.
2. REMOVEVOLUMES specifies volume(s) from which all VSAM data spaces are to be removed and VSAM ownership is to

be taken away—without access to the user catalog that owns the volume(s). For this use of REMOVEVOLUMES, the name of the master catalog and its master password (if any) must be specified in the entryname parameter, and the FILE parameter is required. (You can have only one DD statement with this use of ALTER REMOVEVOLUMES—only volumes of the same device type can be processed.)

For information and cautions about this use of ALTER REMOVEVOLUMES, see the sections "VSAM Volume Cleanup" and "VSAM Volume Recovery Function" in the Catalog Administration Guide.

Abbreviation: RVOL

SCRATCH|NOSCRATCH
specifies whether generation data sets, when they are uncataloged, are to be removed from the VTOC of the volume on which they reside.

SCRATCH
indicates that the data set's format-1 DSCB is to be removed (scratched) from the VTOC so that the data set can no longer be accessed.

Abbreviation: SCR

NOSCRATCH
indicates that the data set's format-1 DSCB is not to be removed from the VTOC.

Abbreviation: NSCR

SHAREOPTIONS(crossregion[crosssystem]**)**
specifies how a data or index component of a cluster or alternate index can be shared among users. (For a full description of data set sharing, see VSAM Administration Guide.)

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 using GRS can access a VSAM data set concurrently.

To share a data set, each user must specify DISP=SHR in the data set's DD statement. The values that can be specified are:

1
specifies that 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. With this option, VSAM ensures complete data integrity for the data set.

2
specifies that 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. With this option, VSAM ensures write integrity by obtaining exclusive control for a control interval when it is to be updated.

If a user requires read integrity, it is that user's responsibility to use the ENQ and DEQ macros appropriately to provide read integrity for the data obtained by the program. (For information on using ENQ and DEQ, see System Macros and Facilities.)

3

specifies that the data set can be fully shared by any number of users. With this option, each user 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 data set failures, and other unpredictable results. This option places heavy responsibility on each user sharing the data set.

4

specifies that the data set can be fully shared by any number of users and buffers used for direct processing are refreshed for each request.

This option requires the program to use the ENQ and DEQ macros to maintain data integrity while sharing the data set. Improper use of the ENQ macro can cause problems similar to those described under SHAREOPTIONS 3. (For information on using ENQ and DEQ, see System Macros and Facilities.)

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 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 values that can be specified 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 failures, and other unpredictable results. This option places heavy responsibility on each user sharing the data set.

4

specifies that the data set can be fully shared. Buffers used for direct processing are refreshed for each request.

This option requires that you use the RESERVE and DEQ macros to maintain data integrity while sharing the data set. (For information on using RESERVE and DEQ, see System Macros and Facilities.) Writing is limited to PUT-update and PUT-insert processing that does not change the high used RBA if your program opens the data set with DISP=SHR.

Data set integrity cannot be maintained unless all jobs accessing the data set in a cross-system

environment specify DISP=SHR. Improper use of the RESERVE macro can cause problems similar to those described under SHAREOPTIONS 3.

Abbreviation: SHR

STAGE|BIND|CYLINDERFAULT
specifies how a data or index component that is stored on a mass storage volume is to be staged.

These parameters can be specified only for the data or index component of a cluster or alternate index.

STAGE
indicates that the component is to be staged from mass storage to a direct access storage staging drive when the component is opened.

BIND
indicates that the component is not only to be staged, but also bound (or retained) on the direct access storage staging drive until it is closed.

CYLINDERFAULT
indicates that the component is not to be staged when it is opened, but that data from it is to be staged as the processing program needs it.

Abbreviation: CYLF

TO(*date*)|FOR(*days*)
specifies the retention period for the entry being altered.

These parameters cannot be specified for the data or index components of clusters, alternate indexes, or catalogs. In the case of a non-VSAM data set the expiration date in the catalog will be updated, but the expiration date in the format-1 DSCB will not be changed.

TO(*date*)
specifies the date, in the form yyddd, where yy is the year and ddd is the number (001 through 365) of the day through which the entry is to be kept.

FOR(*days*)
specifies the number of days for which the entry is to be kept. The maximum number that can be specified is 9999.

If the number specified is 0 through 1830, the cluster is retained for the number of days specified; if the number is between 1831 and 9999, the cluster is retained through the year 1999.

days
can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

UNIQUEKEY|NONUNIQUEKEY
specifies whether the alternate-key value can be found in more than one of the base cluster's data records.

UNIQUEKEY
specifies that each alternate-key value is unique. If the same alternate-key value is found in more than one of the base cluster's data records, an error results.

UNIQUEKEY can be specified only for an empty alternate index (that is, an alternate index that is defined but not yet built).

Abbreviation: UNQK

NONUNIQUEKEY
specifies that a given alternate-key value might 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 specifying 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.

UPDATE
specifies that the cluster's alternate index upgrade set is to be allocated 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.

Note: If the path is opened specifying NOUPDATE and sharing a control block structure which specifies UPDATE, the upgrade set has been allocated and the upgrade set may be updated.

Abbreviation: NUPD

UPDATEPW(password)
specifies an update password for the entry being altered.

Abbreviation: UPDPW

UPGRADE|NOUPGRADE
specifies whether an alternate index is to be upgraded (that is, kept up to date) when its base cluster is modified.

UPGRADE
specifies 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 specified 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).

UPGRADE can be specified only 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 that the alternate index is not modified when its base cluster is modified. NOUPGRADE can be specified for 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 may be specified to alter the data or index components of a cluster or an alternate index.

WRITECHECK
specifies that a record is to be written and then read, without data transfer, to test for the data check condition.

Abbreviation: WCK

NOWRITECHECK
specifies that a record is to be written only.

Abbreviation: NWCK

ALTER EXAMPLES

Alter a Cluster's Attributes: Example 1

In this example, an ALTER command is used to specify passwords for an entry-sequenced (nonindexed) cluster, D50.EXAMPLE.ESDS1. No password for the cluster is required, because the cluster was defined without passwords.

```
//ALTER1   JOB    ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALTER -
          D50.EXAMPLE.ESDS1 -
          MASTERPW(DEPT26M) -
          CONTROLPW(DEPT26C) -
          UPDATEPW(DEPT26U) -
          READPW(DEPT26R) -
          AUTHORIZATION(D26AUTH)
/*
```

The ALTER command adds passwords to the entry-sequenced cluster's cluster catalog entry. However, the passwords are not added to the cluster's data entry. If a user's program supplies the cluster's data entry entryname and opens the data component, the unauthorized user can access the cluster's data records even though the cluster itself is password-protected.

The ALTER command's parameters are:

- D50.EXAMPLE.ESDS1, the name of the entry-sequenced cluster. It is assumed that an alias entrynamed D50 exists for the user catalog D27UCAT2. The data set name, D50.EXAMPLE.ESDS1, causes the ALTER request to be directed to D27UCAT2.
- MASTERPW, CONTROLPW, UPDATEPW, READPW, and AUTHORIZATION, specify passwords and the entryname of the user-security-verification routine.

Alter the Entrynames of Generically Named Clusters: Example 2

In this example, several clusters with similar names, GENERIC.*.BAKER (where "*" is any 1- to 8-character simple name), are renamed so that their entrynames are GENERIC.*.ABLE. The name "GENERIC.*.BAKER" is called a generic name.

```
//ALTER2   JOB   ...
//JOBCAT   DD    DSNAME=USERCAT4,DISP=SHR
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     ALTER -
          GENERIC.*.BAKER -
          NEWNAME(GENERIC.*.ABLE)
/*
```

Job control language statement:

- JOBCAT DD, which makes a catalog available for this job: USERCAT4.

The ALTER command changes each generic entryname, GENERIC.*.BAKER, to GENERIC.*.ABLE. Its parameters are:

- GENERIC.*.BAKER, which identifies the objects to be modified.
- NEWNAME, which specifies that each generic entryname GENERIC.*.BAKER is changed to GENERIC.*.ABLE.

Alter the Attributes of a Generation Data Group: Example 3

In this example, the attributes of a generation data group are modified. Because the attributes of the group 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. Its parameters are:

- GDG01, which identifies the object to be modified.
- NOEMPTY, which specifies that only the oldest generation data set is to be uncataloged when the maximum number of cataloged generation data sets is exceeded.
- SCRATCH, which specifies that the generation data set's DSCB is to be removed from the volume's VTOC when the data set is uncataloged.

The attributes specified for the generation data group with the ALTER command override any attributes previously specified for the GDG.

BLDINDEX

The BLDINDEX command builds alternate indexes for existing data sets. The format of this command is:

```
BLDINDEX  {INFILE(ddname[/password])|
            INDATASET(entryname[/password])}
          {OUTFILE(ddname[/password]
            [  ddname[/password]...])|
            OUTDATASET(entryname[/password]
              [  entryname[/password]...])}
          [EXTERNALSORT|INTERNALSORT]
          [WORKFILES(ddname   ddname)]
          [CATALOG(catname[/password])]
```

BLDINDEX can be abbreviated: BIX

BLDINDEX PARAMETERS

Required Parameters

INFILE(ddname[/password]**)|INDATASET(**entryname[/password]**)**
names the DD statement or data set that identifies the base cluster or a path that points to the base cluster.

INFILE(ddname[/password]**)**
specifies the DD statement that identifies the base cluster or a path that points to the base cluster. The base cluster must be defined in the same catalog as the alternate index, and must contain at least one data record.

password
If the base cluster or path is password protected, supply the read (or higher-level) password of the object named in the DD statement. If you don't supply the password, the operator or TSO terminal user will be prompted to supply the correct password. RACF: The read or higher RACF authority of the object named in the DD statement is required.

Abbreviation: IFILE

INDATASET(entryname[/password]**)**
names the data set that identifies the base cluster or a path that points to the base cluster. The base cluster must be defined in the same catalog as the alternate index, and must contain at least one data record.

password
If the base cluster or path is password protected, you supply the read (or higher-level) password of the named object. Otherwise, the operator or TSO terminal user will be prompted to supply the correct password. RACF: The read or higher RACF authority of the named objects is required.

When you specify INDATASET, an attempt is made to dynamically allocate the base cluster's volume. Therefore, the base cluster's volume must be mounted as permanently resident or reserved.

Abbreviation: IDS

OUTFILE(ddname[/password])|OUTDATASET(entryname[/password])
names the DD statement or data set that identifies the alternate index or a path that points to the alternate index. You can build more than one alternate index for the same base cluster by specifying more than one ddname or data set name with the OUTFILE or OUTDATASET parameter.

OUTFILE(ddname[/password][ddname[/password]...])
specifies the DD statement that identifies the alternate index or a path that points to the alternate index. The alternate index must be defined in the same catalog as the base cluster, and must be empty (that is, its high-used RBA equals zero) or defined with the REUSE attribute.

The alternate index must be related to the base cluster identified with INDATASET or INFILE.

password
If the alternate index or path is password protected, supply the update- or higher-level password of the object named in the DD statement. If you don't supply the password, the operator or TSO terminal user will be prompted to supply the correct password. RACF: The update or higher RACF authority of the named object is required.

Abbreviation: OFILE

OUTDATASET(entryname[/password] [entryname[/password]...])
specifies the data set that identifies the alternate index or a path that points to the alternate index. The alternate index must be defined in the same catalog as the base cluster, and must be empty (that is, its high-used RBA equals zero) or must have been defined with the REUSE attribute.

The alternate index must be related to the base cluster identified with INDATASET or INFILE.

password
If the alternate index or path is password protected, you supply the update- or higher-level password of the named object. Otherwise, the operator or TSO terminal user may be prompted to supply the correct password. RACF: The update or higher RACF authority for the named object is required.

When you specify OUTDATASET, an attempt is made to dynamically allocate the alternate index's volume. Therefore, the volume must be mounted as permanently resident or reserved.

Abbreviation: ODS

Optional Parameters

CATALOG(catname[/password])
names the catalog that the work files are to be defined in. The work files are defined and used by the BLDINDEX routine.

When all alternate indexes are built and the work files are no longer needed by the BLDINDEX routine, they are deleted.

password
: If the catalog is password protected, supply its update- or higher-level password. If you don't supply the password, the operator will be prompted to supply the correct password. RACF: The update or higher RACF authority to the catalog is required.

For information about the order in which a catalog is selected when the CATALOG parameter is not specified, see "Order of Catalog Selection for BLDINDEX" on page 15.

Abbreviation: CAT

EXTERNALSORT|INTERNALSORT
: specifies whether the key-pointer pairs are to be sorted entirely within virtual storage.

EXTERNALSORT
: specifies that two external sort work files will be 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.

INTERNALSORT
: specifies that access method services will sort the key-pointer pairs entirely within the user-provided virtual storage if possible.

  If not enough virtual storage is provided when you specify INTERNALSORT, two external sort work files are built and the key-pointer pairs are sorted externally. In this case, you must provide DD statements as for EXTERNALSORT.

  If the minimum amount of virtual storage is not provided (see VSAM Administration Guide), the BLDINDEX processing terminates with an error message.

Abbreviations: ESORT and ISORT

WORKFILES(ddname ddname**)**
: names the DD statements that identify the work files you want BLDINDEX to define if an external sort of the key-pointer pairs is required.

  You can use DD statements to describe two work files that will be defined and opened before the BLDINDEX routine begins processing the base cluster's data records.

  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 specify the WORKFILES parameter.

  For information on how to code the DD statements that describe the work files, see "DD Statements that Describe the SORT Workfiles" in VSAM Administration Guide.

Abbreviation: WFILE

Build an Alternate Index Over a Key-Sequenced Data Set: Example 1

This example builds an alternate index over a previously defined base cluster, EXAMPLE.KSDS2. Data records have already been loaded into the base cluster, so that it is not empty. The alternate index, its path, and its base cluster are all defined in the same catalog, USERRCAT.

```
//BUILDAIX JOB  ...
//STEPCAT  DD   DSNAME=USERRCAT,DISP=SHR
//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/AIXUPPW) -
              CATALOG(USERRCAT/USER)
/*
```

Job control language statements:

- STEPCAT DD, which describes the catalog.
- BASEDD DD, which describes the base cluster.
- AIXDD DD, which describes the alternate index.
- IDCUT1 and IDCUT2 DD, which describe a volume containing VSAM data space to be made available to BLDINDEX for defining and using two sort work data sets in the event an external sort is performed. This data space will not be used by BLDINDEX if enough virtual storage is available to perform an internal sort. If multiple volumes are involved, a maximum of five volumes for each work file can be specified.

The BLDINDEX command builds an alternate index. The assumption is made that enough virtual storage will be available to perform an internal sort. However, note that DD statements with the default ddnames of IDCUT1 and IDCUT2 have been provided for two external sort work data sets in the event that the assumption is incorrect and an external sort must be performed.

The BLDINDEX command's parameters are:

- INFILE, which names the base cluster. The ddname of the DD statement for this object must be identical to this name. Note that a password is not required, because the base cluster is not protected.
- OUTFILE, which names the alternate index. The ddname of the DD statement for this object must be identical to this name. The update password of the alternate index is also required.
- CATALOG, which identifies the user catalog. If it is necessary for BLDINDEX to use external sort work data sets, they will be defined in and deleted from the catalog. The update password will permit these actions.

CHKLIST

The CHKLIST command lists tape data sets that were opened during a checkpoint. The format of this command is:

CHKLIST	**INFILE(**ddname**)** **[CHECKID(**checkid...**)]** **[OUTFILE(**ddname**)]**

CHKLIST can be abbreviated: CKLST

CHKLIST PARAMETERS

Required Parameter

INFILE(ddname**)**
specifies the name of the DD statement that identifies the checkpoint data set that contains the checkpoint entries to be processed.

Abbreviation: IFILE

Optional Parameters

CHECKID(checkid...**)**
specifies one or more checkpoint identifiers of entries in the checkpoint data set. These entries contain names of tape data sets that were open at the time of the checkpoint.

checkid
checkid must be 1 to 16 characters in length.

The checkid must be the same as was specified in the CHKPT macro. A maximum of 255 checkids can be specified. Multiple checkids may be specified in any sequence.

If the checkpoint data set contains duplicates of a checkid, you can cause all the checkpoint entries with that checkid to be listed:

- By specifying the checkid at least twice
- By specifying, along with the checkid, a checkid for which there is no entry in the checkpoint data set

If you do neither, only the first checkpoint entry found with the checkid is listed.

If CHECKID is omitted, the identifications of tape data sets that were open at the time of a checkpoint are listed for all entries in the checkpoint data set.

CHECKID must be omitted when a member of a partitioned checkpoint data set has been specified after ddname in the DD statement identified by INFILE.

Abbreviation: CHKID

OUTFILE(ddname)
specifies the ddname of the DD statement that identifies a data set other than the SYSPRINT data set to be used as a target data set.

A target data set must meet the requirements shown under "For a Target Data Set" on page 10. If OUTFILE is not specified, the tape data set information is listed in the SYSPRINT data set.

Abbreviation: OFILE

CHKLIST EXAMPLES

Selecting Specific Checkpoint Entries: Example 1

In this example, the tape data sets that were open at checkpoint time are identified and listed on SYSPRINT for checkpoint entries C0000001 and C0000002.

```
//CHKLIST1  JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//CHKPT     DD    DSN=CHKPT.DATASET,DISP=OLD
//SYSIN     DD    *
    CHKLIST -
        INFILE(CHKPT) -
        CHECKID(C0000001 C0000002)
/*
```

Job control statement:

- CHKPT DD, which identifies the checkpoint data set that contains the entries for which the tape data set information is to be listed.

The CHKLIST command prints the tape data set information as specified by the command's parameters:

- INFILE, which points to the CHKPT DD statement. The CHKPT DD statement identifies the checkpoint data set.
- CHECKID, which specifies that the checkpoint entries with checkids C0000001 and C0000002 are the only ones on the checkpoint data set for which the tape data set information is to be listed.

The output shown in Appendix C, "Sample Output from CHKLIST" on page 315 was obtained with this JCL.

Processing a Member of a Partitioned Checkpoint Data Set: Example 2

In this example, the checkpoint data set is a partitioned data set, and member C0000001 is selected for processing by CHKLIST.

```
//CHKLIST2  JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//CHKPTDD   DD    DSN=EXAMPLE.CHKPTDS2(C0000001),
//      DISP=SHR
//SYSIN     DD    *
    CHKLIST -
        INFILE(CHKPTDD)
/*
```

Job control language statement:

- CHKPTDD DD, which identifies the partitioned checkpoint data set and the specific member for which the tape data set information is to be listed.

The CHKLIST command prints the tape data set information as specified by its parameter, INFILE, which points to the CHKPTDD DD statement. The CHKPTDD DD statement identifies the member of the partitioned checkpoint data set.

The CHECKID parameter is omitted, so the checkpoint entry C0000001 will automatically be processed by CHKLIST.

CNVTCAT

The CNVTCAT command converts OS CVOL entries into VSAM catalog entries. The format of this command is:

```
CNVTCAT  {INFILE(ddname)|
           INDATASET(entryname)}
         {OUTFILE(ddname[/password])|
           OUTDATASET(entryname[/password])|
           CATALOG(catname[/password])}
         [CVOLEQUATES((catname   (volser
           [  volser...]))[  (catname...)...])]
         [LIST|NOLIST]
         [MASTERCATALOG(catname[/password])]
```

CNVTCAT can be abbreviated: CNVTC

CNVTCAT PARAMETERS

Required Parameters

INFILE(ddname)|INDATASET(entryname)
specifies the name of a DD statement or an alias entryname for an OS CVOL to be converted to a VSAM catalog.

INFILE(ddname)
specifies the name of a DD statement for an OS CVOL that is to be converted.

ddname
specifies the name of the DD statement that identifies the OS CVOL that is to be converted.

Abbreviation: IFILE

INDATASET(entryname/password)
specifies the alias entryname (in the master catalog) of the OS CVOL that is to be converted.

If INDATASET is specified, an attempt is made to dynamically allocate the OS CVOL. The OS system catalog entryname, SYSCTLG, cannot be specified.

Abbreviation: IDS

entryname
names the entry to be converted.

OUTFILE(ddname[/password])|OUTDATASET(entryname[/password])|CATALOG(catname[/password])
identifies the VSAM catalog to receive the converted OS CVOL entries.

OUTFILE(ddname[/password])
specifies the name of a DD statement that identifies the target VSAM catalog.

ddname
identifies the VSAM target catalog.

password
specifies the update- or higher-level password for a password-protected catalog. RACF: The update or higher RACF authority to the catalog is required.

Abbreviation: OFILE

OUTDATASET(entryname**[/**password**])**
specifies the name of the VSAM target catalog. If OUTDATASET is specified, an attempt is made to dynamically allocate the catalog.

entryname
names the target VSAM catalog. The alias name of the catalog will not be accepted.

password
specifies the update- or higher-level password for a password protected catalog. RACF: The update or higher RACF authority to the catalog is required.

Abbreviation: ODS

CATALOG(catname**[/**password**])**
specifies the name of a catalog that is to receive the converted entries. The alias name of the catalog will not be accepted.

password
specifies the update- or higher-level password for a password-protected catalog. RACF: The update or higher RACF authority for the catalog is required.

Abbreviation: CAT

Optional Parameters

CVOLEQUATES((catname **(**volser**[** volser...**]))[** **(**catname...**)]**
identifies the user catalog that converted control volume pointer entries (CVPEs) point to.

CVPEs point to OS CVOLs that contain entries. The entries in an OS CVOL might have been (or might soon be) converted to VSAM catalog entries.

When a CVPE is converted to an alias entry, the alias entry points to the user catalog that contains (or is to contain) the OS CVOLs converted entries.

When you specify CVOLEQUATES, you must also specify MASTERCATALOG.

catname
specifies the name of an existing VSAM catalog. The catalog named contains or is to contain converted OS CVOL entries that were cataloged in the OS CVOL pointed to by the CVPE being converted.

volser
specifies the volume serial number(s) of one or more OS CVOLs for which entries have been or are to be converted.

Abbreviation: CVEQU

LIST|NOLIST
specifies whether entries are to be listed.

LIST
specifies that entries are to be listed after they are converted.

NOLIST
specifies that entries are not to be listed after they are converted.

Abbreviation: NLIST

MASTERCATALOG(catname[/password])
specifies the name of the master catalog into which any aliases for user catalogs are to be placed. MASTERCATALOG is required when CVOLEQUATES is specified.

password
specifies the master catalog's update- or higher-level password.

Abbreviation: MCAT

CNVTCAT EXAMPLES

The two CNVTCAT examples are related and show how the entries of two OS CVOLs in a system can be converted to entries in a catalog. The OS CVOL on volume VSER09 contains control volume pointer entries (CVPEs) that point to the OS CVOL on volume VSER08. Therefore, the two OS CVOLs are chained together with VSER08's OS CVOL at the end of the chain. When two or more OS CVOLs are chained together, the OS CVOL at the end of the chain should be converted first.

Convert OS CVOL Entries to Entries in a VSAM Catalog: Example 1

In this example, the entries in the OS CVOL on volume VSER08 are converted to catalog entries and written into the USER11 catalog.

```
//CNVTCAT1  JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//OSCAT1    DD    VOL=SER=VSER08,DISP=OLD,DSN=SYSCTLG,
//   UNIT=DISK
//SYSIN     DD    *
      CNVTCAT -
          INFILE(OSCAT1) -
          CATALOG(USER11/MR)
/*
```

Job control language statement:

- OSCAT1 DD, which describes and allocates the OS CVOL that is to be converted.

The CNVTCAT command converts the entries in the OS CVOL on volume VSER08 to entries in a user catalog, USER11. The command's parameters are:

- INFILE, which points to the OSCAT1 DD statement.
- CATALOG, which specifies the name of an existing catalog, the user catalog USER11. USER11 is to receive the converted entries. The user catalog's update- or higher-level password, MR, is required if passwords are used.

Convert an OS CVOL Having Control Volume Pointer Entries: Example 2

In this example, the entries in the OS CVOL on volume VSER09 are converted to catalog entries and written into the USER12 catalog. Some of the entries in the OS CVOL are control volume pointer entries, and point to the OS CVOL on volume VSER08 (which was converted in the previous example).

```
//CNVTCAT2  JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//OSCAT2    DD    VOL=SER=VSER09,DISP=OLD,DSN=SYSCTLG,
//   UNIT=DISK
//SYSIN     DD    *
     CNVTCAT -
        INFILE(OSCAT2) -
        CATALOG(USER12/MR) -
        CVOLEQUATES( -
            (USER11 (VSER08))   ) -
        NOLIST -
        MASTERCATALOG(AMASTCAT)
/*
```

Job control language statement:

- OSCAT2 DD, which describes and allocates the OS CVOL that is to be converted.

The CNVTCAT command converts the entries in the OS CVOL on volume VSER09 to entries in a user catalog, USER12. Because CVOLEQUATES is specified, an alias entry is built and put in the master catalog to relate the name VSER08 to the user catalog USER11. The command's parameters are:

- INFILE, which points to the OSCAT2 DD statement.
- CATALOG, which specifies the name of an existing catalog, the user catalog USER12. USER12 is to receive the converted entries. The user catalog's update- or higher-level password, MR, is required if passwords are used.
- CVOLEQUATES, which specifies that control volume pointer entries (CVPEs) are to be converted to alias entries. Each CVPE in VSER09's OS CVOL points to an entry in VSER08's OS CVOL. Each new alias entry points to the USER11 user catalog connector entry in the master catalog. The alias entries are written into the master catalog.
- NOLIST, which specifies that the entries are not to be listed after they are converted.
- MASTERCATALOG, which identifies the master catalog. This parameter must be specified because CVOLEQUATES is also specified. If the master catalog is password-protected, the catalog's update- or higher-level password must be specified.

DEFINE ALIAS

The DEFINE ALIAS command defines an alternate name for a non-VSAM data set or a user catalog. The format of this command is:

```
DEFINE | ALIAS
       |   (NAME(aliasname)
       |   RELATE(entryname))
       | [CATALOG(catname[/password])]
```

DEFINE can be abbreviated: DEF

DEFINE ALIAS PARAMETERS

Required Parameters

ALIAS
specifies that an alias for a user catalog or non-VSAM data set is to be defined.

NAME(aliasname)
specifies the alias (the alternate entryname) for a user catalog or non-VSAM data set. An alias must be unique within a catalog.

RELATE(entryname)
specifies the name of the entry (the user catalog entryname or the non-VSAM data set name) for which the alias is being defined.

An alias entry must reside in the same catalog as the entry to which it is related. When an alias is defined, catalogs are searched for the related entry. When the related entry is found, the alias entry is added to the catalog.

Abbreviation: REL

Optional Parameter

CATALOG(catname[/password])
identifies the catalog in which the alias 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. For information about the order in which a catalog is selected when the catalog's name is not specified, see "Order of Catalog Selection for DEFINE" on page 16.

catname
specifies the name of the catalog. When the alias is for a user catalog connector, catname is the name of the master catalog.

password
specifies a password. If the catalog is password protected, supply the catalog's update- or higher-level password. If you don't supply the password, the operator or TSO terminal user will be prompted to supply the correct password. RACF: The update or higher RACF authority to the catalog is required.

Abbreviation: CAT

DEFINE ALIAS EXAMPLES

Define an Alias for a Non-VSAM Data Set: Example 1

In this example, an alias is defined for a non-VSAM data set.

```
//DEFALS   JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//STEPCAT  DD     DSNAME=USERCAT4,DISP=OLD
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE ALIAS -
          (NAME(EXAMPLE.NONVSAM1) -
          RELATE(EXAMPLE.NONVSAM) ) -
          CATALOG(USERCAT4/USERMRPW)
/*
```

Job control language statement:

- STEPCAT DD, which makes a catalog available for this job step: USERCAT4.

The DEFINE ALIAS command defines an alias, EXAMPLE.NONVSAM1, for the non-VSAM data set EXAMPLE.NONVSAM. Its parameters are:

- NAME, which specifies the alias (alternate entryname), EXAMPLE.NONVSAM1.
- RELATE, which specifies the name that the alias is an alternate entryname for, EXAMPLE.NONVSAM.
- CATALOG, which supplies the update password 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(D40) -
          RELATE(D27UCAT1)) -
          CATALOG(AMAST1/MASTUPW1)
/*
```

The DEFINE ALIAS command defines an alias, D40, for the user catalog, D27UCAT1. Any data set defined in D27UCAT1 having 'D40' as its first qualifier will be located by VSAM in D27UCAT1 when it is referred to in an access method services command or user program without requiring a JOBCAT or STEPCAT DD statement for D27UCAT1. D27UCAT1 can also be referred to using the alias D40. The parameters are:

- NAME, which specifies the alias, D40.
- RELATE, which specifies the name of the user catalog for which D40 is an alternate entryname (alias).
- CATALOG, which supplies the update password of the master catalog.

DEFINE ALTERNATEINDEX

The DEFINE ALTERNATEINDEX command defines an alternate index. When you use this command, you can specify attributes for the alternate index as a whole and for the components of the alternate index. The format of the DEFINE ALTERNATEINDEX command is:

```
DEFINE ALTERNATEINDEX (parameters) -

  [DATA(parameters)] -

  [INDEX(parameters)] -

  [CATALOG(subparameters)]
```

```
DEFINE    ALTERNATEINDEX
            (NAME(entryname)
            RELATE(entryname[/password])
            {CYLINDERS(primary[ secondary])|
              RECORDS(primary[ secondary])|
              TRACKS(primary[ secondary])}
            VOLUMES(volser[ volser...])
            [ATTEMPTS(number|2)]
            [AUTHORIZATION(entrypoint[ string])]
            [BUFFERSPACE(size)]
            [CODE(code)]
            [CONTROLINTERVALSIZE(size)]
            [CONTROLPW(password)]
            [DESTAGEWAIT|NODESTAGEWAIT]
            [ERASE|NOERASE]
            [EXCEPTIONEXIT(entrypoint)]
            [FILE(ddname)]
            [FREESPACE(CI-percent[ CA-percent]|0 0)]
            [IMBED|NOIMBED]
            [KEYRANGES((lowkey  highkey)
              [(lowkey  highkey)...])]
            [KEYS(length  offset|64 0)]
            [MASTERPW(password)]
            [MODEL(entryname[/password]
              [ catname[/password]])]
            [ORDERED|UNORDERED]
            [OWNER(ownerid)]
            [READPW(password)]
            [RECORDSIZE(average  maximum|
              4086 32600)]
            [REPLICATE|NOREPLICATE]
            [REUSE|NOREUSE]
            [SHAREOPTIONS(crossregion
              [ crosssystem]|1 3)]
            [SPEED|RECOVERY]
            [STAGE|BIND|CYLINDERFAULT]
            [TO(date)|FOR(days)]
            [UNIQUE|SUBALLOCATION]
            [UNIQUEKEY|NONUNIQUEKEY]
            [UPDATEPW(password)]
            [UPGRADE|NOUPGRADE]
            [WRITECHECK|NOWRITECHECK])
```

(Continued on next page.)

```
[DATA
   ([ATTEMPTS(number)]
   [AUTHORIZATION(entrypoint[  string])]
   [BUFFERSPACE(size)]
   [CODE(code)]
   [CONTROLINTERVALSIZE(size)]
   [CONTROLPW(password)]
   [CYLINDERS(primary[  secondary])|
     RECORDS(primary[  secondary])|
     TRACKS(primary[  secondary])]
   [DESTAGEWAIT|NODESTAGEWAIT]
   [ERASE|NOERASE]
   [EXCEPTIONEXIT(entrypoint)]
   [FILE(ddname)]
   [FREESPACE(CI-percent[  CA-percent])]
   [KEYRANGES((lowkey  highkey)
     [(lowkey  highkey)...])]
   [KEYS(length  offset)]
   [MASTERPW(password)]
   [MODEL(entryname[/password]
     [  catname[/password]])]
   [NAME(entryname)]
   [ORDERED|UNORDERED]
   [OWNER(ownerid)]
   [READPW(password)]
   [RECORDSIZE(average  maximum)]
   [REUSE|NOREUSE]
   [SHAREOPTIONS(crossregion[  crosssystem])]
   [SPEED|RECOVERY]
   [STAGE|BIND|CYLINDERFAULT]
   [UNIQUE|SUBALLOCATION]
   [UNIQUEKEY|NONUNIQUEKEY]
   [UPDATEPW(password)]
   [VOLUMES(volser[  volser...])]
   [WRITECHECK|NOWRITECHECK])]
```

```
[INDEX
   ([ATTEMPTS(number)]
   [AUTHORIZATION(entrypoint[  string])]
   [CODE(code)]
   [CONTROLINTERVALSIZE(size)]
   [CONTROLPW(password)]
   [CYLINDERS(primary[  secondary])|
     RECORDS(primary[  secondary])|
     TRACKS(primary[  secondary])]
   [DESTAGEWAIT|NODESTAGEWAIT]
   [EXCEPTIONEXIT(entrypoint)]
   [FILE(ddname)]
   [IMBED|NOIMBED]
   [MASTERPW(password)]
   [MODEL(entryname[/password]
     [  catname[/password]])]
   [NAME(entryname)]
   [ORDERED|UNORDERED]
   [OWNER(ownerid)]
   [READPW(password)]
   [REPLICATE|NOREPLICATE]
   [REUSE|NOREUSE]
   [SHAREOPTIONS(crossregion[  crosssystem])]
   [STAGE|BIND|CYLINDERFAULT]
   [UNIQUE|SUBALLOCATION]
   [UPDATEPW(password)]
   [VOLUMES(volser[  volser...])]
   [WRITECHECK|NOWRITECHECK])

[CATALOG(catname[/password])]
```

DEFINE can be abbreviated: DEF

Required Parameters

ALTERNATEINDEX
specifies that an alternate index is to be defined.

The ALTERNATEINDEX keyword is followed by the parameters specified for the alternate index as a whole. These parameters are enclosed in parentheses and, optionally, are followed by parameters specified separately for the DATA and INDEX components.

Abbreviation: AIX

NAME(entryname**)**
specifies the alternate index's entryname or the name of each of its components. The entryname specified for the alternate index as a whole is not propagated to the alternate index's components.

You can define a separate entryname for the alternate index, its data component, and its index component. If no name is specified for the data or index component, a name is generated.

When the alternate index, data component, and index component are individually named, each can be addressed.

RELATE(entryname[/password]**)**
names the alternate index's base cluster. The base cluster is an entry-sequenced cluster or a key-sequenced cluster that the alternate index is to be related to. You cannot relate an alternate index to a reusable cluster or a relative record cluster.

password
specifies either the base cluster's master password or the catalog's master password (that is, the master password of the catalog that contains the base cluster's entry).

If no password is specified with either the RELATE or the CATALOG parameter, VSAM will ask the operator for the base cluster's master password. RACF: The alter RACF authority to the base cluster or catalog is required.

Abbreviation: REL

CYLINDERS(primary[secondary]**)**|
RECORDS(primary[secondary]**)**|
TRACKS(primary[secondary]**)**
specifies the amount of space to be allocated to the alternate index from the volume's available space when UNIQUE is specified, or from the available space of one of the volume's VSAM data spaces.

Unless you specify the MODEL parameter, you must specify one, and only one, of the following parameters: CYLINDERS, RECORDS, or TRACKS.

When you specify UNIQUE, space is always allocated in cylinders. If the amount of space you specify for the data component of the alternate index is not an even multiple of cylinder size, VSAM rounds the allocation upward to a multiple of a cylinder.

If you specify RECORDS, the amount of space allocated is the minimum number of tracks that are required to contain the specified number of records. However, if you specify

TRACKS or RECORDS and the minimum number of tracks exceeds a cylinder, space is allocated in terms of cylinders.

When UNIQUE or REUSE is specified, each volume can contain a maximum of 16 extents.

When the data component is not divided into key ranges and more than one volume is specified, the primary amount of space is allocated only on the first volume when the component is defined. When the component increases in size so as to extend on to additional volumes, the first allocation on each overflow volume is the primary amount.

When KEYRANGES is specified and the data component is to be contained on more than one volume, the primary amount of space is immediately allocated on each volume required for the key ranges. If more than one key range is allocated on a single volume, the primary amount is allocated for each key range.

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 specify 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 may result in a larger total allocation for your alternate index than that which you specified.

- 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 alternate index's catalog entry, using the LISTCAT command.

When you specify UNIQUE and the alternate index's data space is the first data space on a volume that belongs to a recoverable catalog, one additional cylinder is allocated for the catalog recovery area.

The primary and each secondary allocation must be able to be satisfied within five extents; otherwise, your DEFINE or data set extension will fail. Primary and secondary can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

primary
: specifies the initial amount of space that is to be allocated to the alternate index.

secondary
: specifies the amount of space that is to be allocated each time the alternate index extends, as a secondary extent. When secondary is specified and the alternate index is suballocated, space for the alternate index's data and index components can be expanded to include a maximum of 123 extents.

If the data space that contains all or part of the cluster cannot be extended because the cluster's secondary allocation amount is greater than the data space's secondary allocation amount, VSAM builds a new data space. The new data space's primary and secondary allocation amounts are equal to the cluster's secondary allocation amount.

Abbreviations: CYL, REC, and TRK

VOLUMES(volser[volser...]**)**
specifies the volume(s) on which an alternate index's components are to have space.

If you do not specify the MODEL parameter, VOLUMES must be specified either 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 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.

A volume serial number may be repeated in the list only if the KEYRANGE parameter is specified. You may want to do this in order to have more than one key range on the same volume. Even in this case, repetition is only valid if all duplicate occurrences are used for the primary allocation of some key range.

In a system with the Mass Storage System, an alternate index or component can be defined on a mass storage volume.

The VOLUMES parameter interacts with other DEFINE ALTERNATEINDEX parameters. You should ensure that the volume(s) you specify for the alternate index are consistent with the alternate index's other attributes:

- SUBALLOCATION: If UNIQUE is not specified, the volume must already contain VSAM data space.
- KEYRANGES: If KEYRANGES and UNIQUE are specified for a VSAM catalog, a VSAM data space is built and allocated on a separate volume for each key range.
- ORDERED: If ORDERED is specified, the volumes are used in the order listed. If ORDERED and KEYRANGES are specified, there is a one-to-one correspondence between the key ranges in the key range list and the volumes in the volser list.
- CYLINDERS, RECORDS, TRACKS: The volume(s) contains enough available 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 alternate index and its components.
- CATALOG: The data space on the volume must be defined before the alternate index, and the catalog specified must own the volume.

Abbreviation: VOL

Optional Parameters

ATTEMPTS(number|**2)**
specifies the maximum number of times the operator can try to enter a correct password in response to a prompting message.

This parameter only has effect when the alternate index's master password is not null. A prompting message is issued only when the user has not already supplied the appropriate password.

number
is an integer from 0 to 7 and can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

When ATTEMPTS(0) is coded, the operator is not prompted and is not allowed to enter a password from the console.

Note to TSO users: At a TSO terminal, the logon password is checked first, before the user is prompted to supply a password for the alternate index. Checking the logon password counts as one attempt to obtain a password. If ATTEMPTS is not specified, the user has one attempt to supply the alternate index's password, because the default is 2.

Abbreviation: ATT

AUTHORIZATION(entrypoint**[** string**])**
specifies that a user-security-verification routine (USVR) is available for additional security verification. When a protected alternate index is opened and the user supplies a correct password other than the alternate index's master password, the USVR receives control. For details on coding a USVR, see VSAM Administration Guide.

If a USVR is loaded from an unauthorized library during access method services processing, an abnormal termination will occur. See "Authorized Program Facility" in VSAM Administration Guide.

This parameter only has effect when the master password is not null.

entrypoint
specifies the name of the USVR.

string
specifies information to be passed to the USVR when it receives control to verify authorization.

Abbreviation: AUTH

BUFFERSPACE(size**)**
specifies the minimum space that your program's address space is to provide for buffers.

When you specify BUFFERSPACE, you must specify at least enough space to contain two data control intervals and one index control interval. The buffer space size you specify helps VSAM determine the data component's and index component's control interval size. If the specified size is less than VSAM requires for the buffers needed to run your job, VSAM terminates your job and provides an appropriate error message.

When you do not specify BUFFERSPACE, VSAM determines the buffer space size. VSAM determines the control interval size first, then sets the buffer space amount equal to two data control intervals and one index control interval.

size
is the amount of space to be provided for buffers. The value can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form, but must not exceed 16776704.

Abbreviations: BUFSPC or BUFSP

CATALOG(catname[/password]**)**
identifies the catalog in which the alternate index is to be defined. The catalog also contains the base cluster's entry (see the RELATE parameter above). For information about the order in which a catalog is selected if the catalog's name is not specified, see "Order of Catalog Selection for DEFINE" on page 16.

catname
specifies the catalog's name.

password
specifies the catalog's update- or higher-level password. If the catalog is password-protected, you must supply the catalog's update- or higher-level password. If no password is specified with the CATALOG parameter, VSAM will prompt the operator for the catalog's update password. RACF: The update or higher RACF authority to the catalog is required.

If the base cluster's master password is not specified with the RELATE parameter, the catalog's master password must be specified. RACF: The alter RACF authority to the catalog is required.

If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved.

Abbreviation: CAT

CODE(code**)**
specifies a code name for the alternate index. If an attempt is made to access a password-protected alternate index without first supplying an appropriate password, a prompting message will be issued to the operator's console (see ATTEMPTS above). The prompting message includes the code name, which identifies the alternate index without revealing its entryname.

This parameter only has effect when the alternate index's master password is not null.

If you do not specify a code name for the alternate index, the prompting message identifies the alternate index with its entryname.

CONTROLINTERVALSIZE(size**)**
specifies the size of the alternate index's control intervals. The size of the control interval depends on the maximum size of data records, and on the amount of buffer space specified.

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
for the alternate index's data component.

Because an alternate index always has the spanned attribute, the control interval size can be less than the maximum record length. The sizes you can specify

(between 512 and 32 768 bytes) are integer multiples of 512 or 2048:

CISZ = (n x 512) or (n x 2048)

where n is a positive integer from 1 to 16.

When you specify a size that is not a multiple of 512 or 2048, VSAM chooses the next higher multiple.

size
for the alternate index's index component.

You can specify the following values for the index component's control interval size:

CISZ = [512 | 1024 | 2048 | 4096]

For a discussion of the relationship between control interval size and physical block size, refer to "Optimizing VSAM Performance" in VSAM Administration Guide. The discussion also includes restrictions that apply to control interval size and physical block size.

Abbreviations: CISZ or CNVSZ

CONTROLPW(password)
specifies a control password for the entry being defined.

When specified as a parameter of DATA or INDEX, the control password allows the user's program to open the data component or index component for read and write processing of the component's control intervals (that is, the entire control interval, including the data portion of stored records and the control fields VSAM inserts into stored records and control intervals).

If a read or update password is the only password specified for the object, it (the highest-level password) propagates upward and becomes the password for all higher unspecified levels.

Abbreviation: CTLPW

DESTAGEWAIT|NODESTAGEWAIT
specifies whether an alternate index or its component that is stored on a mass storage volume is to be destaged synchronously or asynchronously with respect to the program that closes it.

Note: When the alternate index or component is not stored on a mass storage volume, the attribute is ineffective until the direct access storage volume on which the alternate index or component is stored is converted to a mass storage volume (using the CONVERTV command, described in Mass Storage System (MSS) Services: Reference Information).

DESTAGEWAIT
indicates that destaging is to be completed before VSAM returns control to the program that issued the CLOSE macro. VSAM can thus notify the program whether destaging was successful.

Abbreviation: DSTGW

NODESTAGEWAIT
indicates that notification of unsuccessful destaging is to be made only by a message to the operator.

Abbreviation: NDSTGW

When one of these parameters is specified for the data component and the other parameter is specified for the

index component, the components are destaged separately as specified, unless the sequence set of the index component is embedded in the data. In that case, the parameter specified for the index component applies to both components.

ERASE|NOERASE
specifies whether the alternate index data component's records are to be erased when the alternate index is deleted.

This attribute applies only to the alternate index's data component.

ERASE
specifies that the records of the alternate index data component are to be overwritten with binary zeros when the alternate index is deleted.

Abbreviation: ERAS

NOERASE
specifies that the records of the alternate index data component are not to be overwritten with binary zeros.

Abbreviation: NERAS

EXCEPTIONEXIT(entrypoint)
specifies the name of the user-written routine, called the 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 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 will occur.

Abbreviation: EEXT

FILE(ddname)
names the DD statement that identifies the direct access device(s) and volume(s) on which space is to be allocated to the alternate index.

If more than one volume is specified in a volume list, all volumes must be of the same device type. When the data component and index component 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.

When the alternate index is defined in a recoverable catalog, and FILE is specified as a parameter of ALTERNATEINDEX, the FILE parameter can identify the DD statement for all volumes on which space is to be allocated.

When defining into a recoverable catalog, FILE also identifies the volume containing the base cluster's index component (when the base cluster is key sequenced) or data component (when the base cluster is entry sequenced).

When the alternate index is cataloged in a recoverable catalog, the DD statement can be concatenated if the volumes are a different device type. Part of the concatenated DD statement describes the alternate index's or component's volumes (of one device type); the other part

of the DD statement describes the volume that contains the catalog recovery area (of another device type).

If the FILE parameter is not specified, an attempt is made to dynamically allocate the required volumes. The volume(s) must be mounted as permanently resident or reserved.

The DD statement you specify must be in the form:

```
//ddname DD UNIT=(devtype[,unitcount]),
//  VOL=SER=(volser1,volser2,volser3,...),...
```

Note: 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 amount of space that is to be left empty 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 "BLDINDEX" on page 49).

The amount of 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 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.

  The percentages, which must be equal to or less than 100, can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

When you specify 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
specifies whether the sequence set (the lowest level of the index) is to be placed with the alternate index's data component.

IMBED
specifies that the sequence-set record for each control area is to be written as many times as it will fit on the first track adjacent to the control area. If the allocation is less than a cylinder, one track will be added to the primary and secondary allocation quantities.

Abbreviation: IMBD

NOIMBED
specifies that the sequence-set record for each control area is to be written with the other index records.

Abbreviation: NIMBD

The IMBED|NOIMBED parameter interacts with the REPLICATE|NOREPLICATE parameter in determining the physical attributes of the index of the alternate index. These index attributes can be used in the following combinations:

- The sequence-set records are adjacent to the data control areas, and only the sequence-set records are replicated (IMBED and NOREPLICATE).
- The sequence-set records are adjacent to the data control intervals, and all levels of index records are replicated (IMBED and REPLICATE).
- All index records are together, and all index records are replicated (REPLICATE and NOIMBED).
- All index records are together, and no index records are replicated (NOREPLICATE and NOIMBED).

For some applications, specifying index options can improve the application's performance. For information on how the index's optional attributes affect performance, see VSAM Administration Guide.

KEYRANGES((lowkey highkey**)**
[(lowkey highkey**)...])**
specifies that portions of the alternate index's data component are to be put on different volumes. Each portion of the alternate index is called a key range.

The maximum number of key ranges for an alternate index is 123. Key ranges must be in ascending order, and are not allowed to overlap. However, a gap can exist between two key ranges. You cannot insert records within the gap.

Keys can contain 1 to 64 characters; 1 to 128 hexadecimal characters if coded as X'lowkey' X'highkey'.

lowkey
specifies the low key of the key range. If lowkey is shorter than the actual keys, it will be padded on the right with binary zeros.

highkey
specifies the high key of the key range. If highkey is shorter than the actual keys, it will be padded on the right with binary ones.

The KEYRANGES parameter interacts with other DEFINE ALTERNATEINDEX parameters. You should ensure that the KEYRANGES you specify for the alternate index are consistent with the alternate index's other attributes.

- VOLUMES: There should be as many volume serial numbers in the volser list as there are key ranges.

  When a volume serial number is duplicated in the volser list, more than one key range is allocated space on the volume. When more than one key range is to be contained on a volume, UNIQUE cannot be coded for the alternate index's data component.

  When there are more volumes in the volser list than there are key ranges, the excess volumes are used for overflow records from any key range without consideration for key range boundaries.

  When there are fewer volumes in the volser list than there are key ranges, the excess key ranges are allocated on the last volume specified. UNIQUE cannot also be specified.

- UNIQUE: When UNIQUE is specified, each key range resides in its own VSAM data space on a separate volume. Other key ranges for the alternate index cannot also reside on the volume.

- ORDERED: There is a one-to-one correspondence between the volumes in the volser list and the key ranges: The

first volume on the volume list contains the first key range, the second volume contains the second key range, and so on.

If a volume cannot be allocated in the order specified by the volser list, your alternate index definition job terminates with an error message. (For example, the job would terminate if there were no space on one of the volumes specified.)

- KEYS: The low key and high key values must not exceed the key length specified in the KEYS parameter. The KEYS parameter must be specified in the same component as the KEYRANGE parameter.

Abbreviation: KRNG

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 sum of length plus offset cannot be greater than the length of the base cluster's data record.

When the base cluster's data record is allowed to span control intervals, the record's alternate-key field is within the record's first segment (that is, in the first control interval).

length offset
specifies 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. You can express length and offset in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

MASTERPW(password**)**
specifies a master password for the entry being defined. The master password allows all access method services operations against the alternate index entry and its data and index entries, and allows the user's program to access the alternate index's contents without restriction. For more details on how passwords can be used, see Catalog Administration Guide or VSAM Administration Guide.

The AUTHORIZATION, CODE, and ATTEMPTS parameters have no effect unless the entry has a master password associated with it. If MASTERPW is not specified, the highest-level password specified becomes the password for all higher levels.

Abbreviation: MRPW

MODEL(entryname[/password]
[catname[/password]**])**
specifies an existing entry to be used as a model for the entry being defined.

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 Catalog Administration Guide.

You may 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 (alternate index, data, or index) of the model to be used and the name of the entry to be defined.

Unless another entry is specified with the MODEL parameter as a subparameter of DATA or INDEX, when you use an alternate index entry as a model for an alternate index, the data and index components of the model entry are used as models for data and index components of the entry still to be defined.

entryname
names the entry to be used as a model.

password
specifies a password. If the model entry is password-protected and it is cataloged in a password-protected catalog, you supply the read- or higher-level password of either the model entry or its catalog. If you supply both passwords, the catalog's password is used. RACF: The read or higher RACF authority to the model or catalog is required.

If you are not specifying new protection attributes for the alternate index and you wish to copy the model's passwords and protection attributes, you must supply the master password of either the model entry or its catalog. RACF: The alter RACF authority to the model or catalog is required.

catname
names the model entry's catalog. You must identify the catalog that contains the model entry when:

- You want to specify the catalog's password instead of the model entry's password.
- 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. For information about the order in which a catalog is selected when the catalog's name is not specified, see "Order of Catalog Selection for DEFINE" on page 16.

ORDERED|UNORDERED
specifies whether volumes are to be used in the order in which they are listed in the VOLUMES parameter.

ORDERED
specifies that the volumes are to be used in the order in which they are listed for the VOLUMES parameter.

When you want each key range to reside on a separate volume, you can use ORDERED so that the first key range goes on the first volume, the second key range goes on the second volume, and so on.

If ORDERED is specified and the volumes cannot be allocated in the order specified, the command is terminated. (For example, the command would terminate if there were no space on one of the volumes specified.)

Abbreviation: ORD

UNORDERED
specifies no order for the use of the volumes specified in the VOLUMES parameters.

Abbreviation: UNORD

OWNER(ownerid)
specifies the identification of the alternate index's owner.

Note to TSO users:

If the owner is not identified with the OWNER parameter, the TSO user's userid becomes the ownerid.

READPW(password)
specifies a read password for the entry being defined. The read password permits read operations against the entry's records.

When specified as a parameter of DATA or INDEX, the read password allows the user's program to open the data component or the index component for read-only processing of the component's data records (that is, the data portion of the sorted record, not the control fields VSAM inserts into stored records). For more details on how passwords can be used, see Catalog Administration Guide.

Abbreviation: RDPW

RECORDSIZE(average maximum|4086 32600)
specifies 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. average and maximum are any integer values that do not exceed the capacity of a control area. It can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. You can identify the records as fixed length by specifying the same value for average and maximum.

You can use the following 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 issue the access method services LISTCAT command to determine the base cluster's prime-key length).
- n = 1 when UNIQUEKEY is specified (RECSZ is also the maximum record size).
- n = the number of data records in the base cluster that contain the same alternate-key value, when NONUNIQUEKEY is specified.

When you also specify NONUNIQUEKEY, the record size you specify should be large enough to allow for as many key pointers or RBA pointers as you anticipate. The record length values apply only to the alternate index's data component.

Note: REPRO and EXPORT will not support data sets with record sizes greater than 32760.

Abbreviation: RECSZ

REPLICATE|NOREPLICATE
specifies how many times each record in the alternate index's index component is to be written on a direct access device track.

For a discussion of the relationship between IMBED|NOIMBED and REPLICATE|NOREPLICATE, see the IMBED|NOIMBED parameter above.

REPLICATE
specifies that each index record is to be written on a DASD track as many times as it will fit.

When you specify REPLICATE, rotational delay is reduced and performance is improved. However, the alternate index's index usually takes more direct access device space.

Abbreviation: REPL

NOREPLICATE
specifies that the index record is to be written on a DASD track only one time.

Abbreviation: NREPL

REUSE|NOREUSE
specifies whether the alternate index can be used over and over as a new alternate index.

REUSE
specifies 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 if you open it with an access control block that specifies 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.

You cannot specify REUSE when you also specify KEYRANGES or UNIQUE for the alternate index or its components. Reusable alternate indexes may be multivolumed and are restricted to 16 physical extents per volume.

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.

For data integrity purposes, you should make sure that share options specified for data and index components are the same. For a full description of data set sharing, see VSAM Administration Guide.

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 using GRS can access a VSAM data set concurrently. To share a data set, each user must specify DISP=SHR in the data set's DD statement. The values that can be specified are:

1
specifies that 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. With this option, VSAM ensures complete data integrity for the data set.

2
specifies that 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. With this option, VSAM ensures write integrity by obtaining exclusive control for a control interval when it is to be updated.

If a user requires read integrity, it is that user's responsibility to use the ENQ and DEQ macros appropriately to provide read integrity for the data the program obtains. (For information on using ENQ and DEQ, see System Macros and Facilities.)

3
specifies that the data set can be fully shared by any number of users. With this option, each user 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 data set failures, and other unpredictable results. This option places heavy responsibility on each user sharing the data set.

4
specifies that the data set can be fully shared by any number of users, and buffers used for direct processing are refreshed for each request. This option requires your program to use the ENQ and DEQ macros to maintain data integrity while sharing the data set. Improper use of the ENQ macro can cause problems similar to those described under SHAREOPTIONS 3. (For information on using ENQ and DEQ, see System Macros and Facilities.)

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 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 values that can be specified 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

failures, and other unpredictable results. This option places heavy responsibility on each user sharing the data set.

4

specifies that the data set can be fully shared. Buffers used for direct processing are refreshed for each request.

This option requires that you use the RESERVE and DEQ macros to maintain data integrity while sharing the data set. (For information on using RESERVE and DEQ, see System Macros and Facilities.) Writing is limited to PUT-update and PUT-insert processing that does not change the high-used RBA if your program opens the data set with DISP=SHR.

Data set integrity cannot be maintained unless all jobs accessing the data set in a cross-system environment specify DISP=SHR. Improper use of the RESERVE macro can cause problems similar to those described under SHAREOPTIONS 3.

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 results in the appropriate buffer being written immediately into the VSAM object's direct access device space. VSAM writes out the buffer in the user's address space that contains the new or updated data record.
- Each GET request results in all the user's input buffers being refreshed. The contents of each data and index buffer being used by the user's program are retrieved from the VSAM object's direct access device.

Abbreviation: SHR

SPEED|RECOVERY
specifies whether the data component's control areas are to be preformatted before alternate index records are loaded into them.

When you specify RECOVERY, your initial load takes longer because the control areas are written initially with end-of-file indicators and again with your alternate index records. When you specify SPEED, your initial load is quicker.

SPEED
specifies that the data component's space is not to be preformatted. Its space might contain data records from a previous use of the space, or it might contain binary zeros; its contents are unpredictable.

If the initial load fails, you must load the alternate index records again from the beginning because VSAM is unable to determine where your last correctly written record is. 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 that has just been written as the last record in the alternate index.

If the initial load fails, you can resume loading alternate index records after the last correctly written record, because an end-of-file indicator identifies it as the last record.

Abbreviation: RCVY

STAGE|BIND|CYLINDERFAULT
specifies how an alternate index or component that is stored on a mass storage volume is to be staged.

When the alternate index or component is not stored on a mass storage volume, the attribute is ineffective until the direct access storage volume on which the alternate index or component is stored is converted to a mass storage volume. (The CONVERTV command used for this conversion is described in Mass Storage System (MSS) Services: Reference Information.)

Unless the sequence set of the index component is embedded in the data, when one of these parameters is specified for the data component and another parameter is specified for the index component, the components are staged separately as specified. In that case, the parameter specified for the index component applies to both components.

STAGE
indicates that the alternate index or component is to be staged from mass storage to a direct access storage staging drive when the alternate index or component is opened.

If the alternate index or component cannot be staged at open time because of heavy staging activity of other objects, data is staged as the processing program needs it.

BIND
indicates that the alternate index or component is not only to be staged, but also to be bound—that is, retained on the direct access storage staging drive until it is closed. If the alternate index or component cannot be staged at open time because of heavy staging activity of other objects, data is staged as the processing program needs it.

CYLINDERFAULT
indicates that the alternate index or component is not to be staged when it is opened, but that data from it is to be staged as the processing program needs it.

Abbreviation: CYLF

TO(date)|FOR(days)
specifies the retention period for the alternate index. The alternate index is not automatically deleted when the expiration date is reached. When you do not specify a retention period, the alternate index can be deleted at any time.

TO(date)
specifies the date, in the form yyddd, where yy is the year and ddd is the Julian date (001, for January 1, through 365, for December 31), through which the alternate index is to be kept before it is allowed to be deleted.

FOR(days)
specifies the number of days for which the alternate index is to be kept before it is allowed to be deleted. days can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

The maximum number that can be specified for days is 9999. If the number specified is 0 through 1830, the retention period is the number of days specified. If the number specified is between 1831 and 9999, the retention period is through the year 1999.

UNIQUE|SUBALLOCATION
specifies whether the alternate index's components are allocated an amount of space from the volume's available space (UNIQUE) or from a VSAM data space's available space (SUBALLOCATION).

UNIQUE
specifies that a VSAM data space is to be built and assigned exclusively to each component of the alternate index. The data space is created when the alternate index is defined. The alternate index's volume(s) must be mounted.

VSAM builds a DSCB in the volume's table of contents (VTOC) to describe the data space. The name of the data space, which is the same as the component's name, is put in the DSCB. A subentry is added to the volume entry (in the VSAM catalog) to describe the VSAM data space.

Abbreviation: UNQ

SUBALLOCATION
specifies that space from one of the VSAM data spaces on the volume is to be assigned to the alternate index's component.

Abbreviation: SUBAL

The space allocation attribute interacts with other DEFINE ALTERNATEINDEX parameters. You should ensure that the space allocation attribute you specify for the alternate index is consistent with other attributes:

- REUSE: You cannot specify REUSE when you specify UNIQUE for an alternate index or its components.
- KEYRANGES: When UNIQUE is specified, a data space is built and allocated on a separate volume for each key range.
- VOLUMES: When UNIQUE is not specified, VSAM data space must exist on the volume that is to contain the alternate index's component.

  When UNIQUE is specified, and more than one volume is specified, VSAM must already own all the volumes except the first. If there is no VSAM space on a volume, you must execute a DEFINE SPACE CANDIDATE before your DEFINE UNIQUE.

UNIQUEKEY|NONUNIQUEKEY
specifies whether more than one data record (in the base cluster) can contain the same key value for the alternate index.

UNIQUEKEY
specifies that each alternate index key points to only one data record. When the alternate index is built (see "BLDINDEX" on page 49) and more than one data record contains the same key value for the alternate index, the BLDINDEX processing terminates with an error message.

Abbreviation: UNQK

NONUNIQUEKEY
specifies that a key value for the alternate index can point to more than one data record in the base cluster. The alternate index's key record can point to a maximum of 32768 records with nonunique keys.

When you specify NONUNIQUEKEY, the value you specify as the maximum record size should be large enough to allow for alternate index records that point to more than one data record.

Abbreviations: NUNQK

UPDATEPW(password)
specifies the update password for the entry being defined. The update password permits read and write operations against the entry's records.

When specified as a parameter of DATA or INDEX, the update password allows the user's program to open the data component or index component for read and write processing of the component's VSAM records (that is, the data portion of the stored record, not the control fields that VSAM inserts into stored records).

If a read password is the only password specified for the object (that is, it is the highest-level password), it propagates upward and becomes the password for all higher levels. If you specify a higher-level password and do not specify an update password, the update or read password is null.

Abbreviation: UPDPW

UPGRADE|NOUPGRADE
specifies whether the alternate index is to be upgraded (that is, kept up to date) when its base cluster is modified.

UPGRADE
specifies that the cluster's alternate index is upgraded 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 "BLDINDEX" on page 49). 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
specifies that the alternate index is not to be upgraded when its base cluster is modified.

Abbreviation: NUPG

WRITECHECK|NOWRITECHECK
specifies whether an alternate index or component is to be checked by a machine action called write check when a record is written into it.

WRITECHECK
specifies that a record is written and then read, without data transfer, to test for the data check condition.

Abbreviation: WCK

NOWRITECHECK
specifies that the alternate index or component is not to be 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. 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 alternate index as a whole. Restrictions are noted with each parameter's description.

DEFINE ALTERNATEINDEX EXAMPLE

Define an Alternate Index: Example 1

In this example, an alternate index is defined. An example for DEFINE CLUSTER illustrates the definition of the alternate index's base cluster, EXAMPLE.KSDS2. A subsequent example illustrates the definition of a path, EXAMPLE.PATH, that allows you to 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, USERRCAT.

```
//DEFAIX1  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
      DEFINE ALTERNATEINDEX -
             (NAME(EXAMPLE.AIX) -
             RELATE(EXAMPLE.KSDS2) -
             MASTERPW(AIXMRPW) -
             UPDATEPW(AIXUPPW) -
             KEYS(3 0) -
             RECORDSIZE(40 50) -
             VOLUMES(VSER01) -
             CYLINDERS(3 1) -
             NONUNIQUEKEY -
             UPGRADE) -
          CATALOG(USERRCAT/USERUPPW)
/*
```

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 VSER04, and allocates 3 cylinders for the alternate index's use. Because the alternate index is being defined into a recoverable catalog, the catalog recovery volume will be dynamically allocated. The command's parameters are:

- NAME, which specifies that the alternate index's name is EXAMPLE.AIX.
- RELATE, which identifies the alternate index's base cluster, EXAMPLE.KSDS2.
- MASTERPW and UPDATEPW, which specifies the alternate index's master password, AIXMRPW, and update password, AIXUPPW.
- KEYS, which 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, which 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, which specifies that the alternate index is to reside on volume VSER01. This example assumes that the volume is already cataloged in the user catalog, USERRCAT.
- CYLINDERS, which specifies that 3 cylinders are allocated for the alternate index's space. When the alternate index is extended, it is to be extended in increments of 1 cylinder.
- NONUNIQUEKEY, which specifies that the alternate key value might be the same for two or more data records in the base cluster.

- UPGRADE, which specifies that the alternate index is to be opened by VSAM and upgraded each time the base cluster is opened for processing.
- CATALOG, which specifies that the alternate index is to be defined in the user catalog, USERRCAT. The example also supplies the master catalog's update password, USERUPPW.

DEFINE CLUSTER

The DEFINE CLUSTER command defines a cluster for a key-sequenced, entry-sequenced, or relative record data set. When you use this command, you can specify attributes for the cluster as a whole and for the components of the cluster. The general format of the DEFINE CLUSTER command is:

```
DEFINE CLUSTER (parameters) -

  [DATA(parameters)] -

  [INDEX(parameters)] -

  [CATALOG(subparameters)]
```

DEFINE	CLUSTER (NAME(entryname) {CYLINDERS(primary[secondary])| RECORDS(primary[secondary])| TRACKS(primary[secondary])} VOLUMES(volser[volser...]) [ATTEMPTS(number|2)] [AUTHORIZATION(entrypoint[string])] [BUFFERSPACE(size)] [CODE(code)] [CONTROLINTERVALSIZE(size)] [CONTROLPW(password)] [DESTAGEWAIT|NODESTAGEWAIT] [ERASE|NOERASE] [EXCEPTIONEXIT(entrypoint)] [FILE(dname)] [FREESPACE(CI-percent [CA-percent]|0 0)] [IMBED|NOIMBED] [INDEXED|NONINDEXED|NUMBERED] [KEYRANGES((lowkey highkey) [(lowkey highkey)...])] [KEYS(length offset|64 0)] [MASTERPW(password)] [MODEL(entryname[/password] [catname[/password]])] [ORDERED|UNORDERED] [OWNER(ownerid)] [READPW(password)] [RECORDSIZE(average maximum)] [REPLICATE|NOREPLICATE] [REUSE|NOREUSE] [SHAREOPTIONS(crossregion [crosssystem]|1 3)] [SPANNED|NONSPANNED] [SPEED|RECOVERY] [STAGE|BIND|CYLINDERFAULT] [TO(date)|FOR(days)] [UNIQUE|SUBALLOCATION] [UPDATEPW(password)] [WRITECHECK|NOWRITECHECK])

(Continued on next page.)

```
[DATA
  ([ATTEMPTS(number)]
  [AUTHORIZATION(entrypoint[ string])]
  [BUFFERSPACE(size)]
  [CODE(code)]
  [CONTROLINTERVALSIZE(size)]
  [CONTROLPW(password)]
  [CYLINDERS(primary[ secondary])|
    RECORDS(primary[ secondary])|
    TRACKS(primary[ secondary])]
  [DESTAGEWAIT|NODESTAGEWAIT]
  [ERASE|NOERASE]
  [EXCEPTIONEXIT(entrypoint)]
  [FILE(dname)]
  [FREESPACE(CI-percent[ CA-percent])]
  [KEYRANGES((lowkey highkey)
    [(lowkey highkey)...])]
  [KEYS(length offset)]
  [MASTERPW(password)]
  [MODEL(entryname[/password]
    [ catname[/password]])]
  [NAME(entryname)]
  [ORDERED|UNORDERED]
  [OWNER(ownerid)]
  [READPW(password)]
  [RECORDSIZE(average maximum)
  [REUSE|NOREUSE]
  [SHAREOPTIONS(crossregion[ crosssystem])]
  [SPANNED|NONSPANNED]
  [SPEED|RECOVERY]
  [STAGE|BIND|CYLINDERFAULT]
  [UNIQUE|SUBALLOCATION]
  [UPDATEPW(password)]
  [VOLUMES(volser[ volser...])]
  [WRITECHECK|NOWRITECHECK])]
```

```
[INDEX
  ([ATTEMPTS(number)]
  [AUTHORIZATION(entrypoint[ string])]
  [CODE(code)]
  [CONTROLINTERVALSIZE(size)]
  [CONTROLPW(password)]
  [CYLINDERS(primary[ secondary])|
    RECORDS(primary[ secondary])|
    TRACKS(primary[ secondary])]
  [DESTAGEWAIT|NODESTAGEWAIT]
  [EXCEPTIONEXIT(entrypoint)]
  [FILE(dname)]
  [IMBED|NOIMBED]
  [MASTERPW(password)]
  [MODEL(entryname[/password]
    [ catname[/password]])]
  [NAME(entryname)]
  [ORDERED|UNORDERED]
  [OWNER(ownerid)]
  [READPW(password)]
  [REPLICATE|NOREPLICATE]
  [REUSE|NOREUSE]
  [SHAREOPTIONS(crossregion[ crosssystem])]
  [STAGE|BIND|CYLINDERFAULT]
  [UNIQUE|SUBALLOCATION]
  [UPDATEPW(password)]
  [VOLUMES(volser[ volser...])]
  [WRITECHECK|NOWRITECHECK])

[CATALOG(catname[/password])]
```

DEFINE can be abbreviated: DEF

DEFINE CLUSTER PARAMETERS

Required Parameters

CLUSTER
specifies that a cluster is to be defined.

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 specified separately for the DATA and INDEX components.

Abbreviation: CL

NAME(entryname**)**
specifies the cluster's entryname or the name of each of its components. The entryname specified for the cluster as a whole is not propagated to the cluster's components.

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.

CYLINDERS(primary**[** secondary**])|**
RECORDS(primary**[** secondary**])|**
TRACKS(primary**[** secondary**])**
specifies the amount of space to be allocated to the cluster from the volume's available space, when UNIQUE is specified, or from the available space of one of the volume's VSAM data spaces.

If you do not specify the MODEL parameters, you must specify one, and only one, of the following parameters: CYLINDERS, RECORDS, or TRACKS.

When you specify UNIQUE, space is always allocated in cylinders. If the amount of space you specify for the data component of the cluster is not an even multiple of cylinder size, VSAM rounds the allocation upward to a multiple of a cylinder.

If you specify RECORDS, the amount of space allocated is the minimum number of tracks that are required to contain the specified number of records.

Regardless of the allocation type, the control area (CA) size for the data set is calculated based on the smaller of the two allocation quantities (primary or secondary) specified in the define command. A CA is never greater than a single cylinder, but may be less (that is, some number of tracks) depending upon the allocation amount and type used. When tracks or records are specified, the space allocation unit (the CA size) may be adjusted to one cylinder. This occurs if the calculated CA size contains more tracks than exist in a single cylinder of the device being used.

DEFINE RECORDS will allocate sufficient space to contain the specified number of records, but variable factors unknown at define time (such as key compression or method of loading records) may result in inefficient use of the space allocated. This may prevent every data CA from being completely used, and you may be unable to load the specified number of records without requiring secondary allocation.

When UNIQUE or REUSE is specified, each volume can contain a maximum of 16 extents.

When the data component is not divided into key ranges and more than one volume is specified, the primary amount of

space is allocated only on the first volume when the component is defined. The secondary amount must be coded to extend the component to additional volumes. When the component increases in size so as to extend to additional volumes, the first allocation on each overflow volume is the primary amount.

When KEYRANGES is specified and the data component is to be contained on more than one volume, the primary amount of space is immediately allocated on each volume required for the key ranges. If more than one key range is allocated on a single volume, the primary amount is allocated for each key range.

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 DATA and INDEX. When a key-sequenced cluster is being defined and the space is specified as a parameter of:

- CLUSTER, 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 may result in a larger total allocation for your cluster than you specified.

- 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.

When you specify UNIQUE, and the cluster's data space is the first data space on a volume that belongs to a recoverable catalog, the equivalent in tracks of one additional cylinder is allocated for the recovery area data space.

The primary and each secondary allocation must be able to be satisfied in 5 extents; otherwise, your DEFINE or data set extension will fail.

primary
: specifies the initial amount of space that is to be allocated to the cluster.

secondary
: specifies the amount of space that is to be allocated each time the cluster extends, as a secondary extent. When secondary is specified and the cluster is suballocated, space for the cluster's data and index components can be expanded to include a maximum of 123 extents.

  If the data space that contains all or part of the cluster cannot be extended because the cluster's secondary allocation amount is greater than the data space's secondary allocation amount, VSAM builds a new data space. The new data space's primary and secondary allocation amounts are equal to the cluster's secondary allocation amount.

primary and secondary can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

Abbreviations: CYL, REC, and TRK

VOLUMES(volser[volser...]**)**
specifies the volume(s) on which a cluster's components are to have space.

If you do not specify the MODEL parameter, VOLUMES must be specified either as a parameter of CLUSTER, or as a parameter of both DATA and INDEX.

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.

The volume serial number may be repeated in the list only if the KEYRANGE parameter is specified. You may want to do this in order to have more than one key range on the same volume. Even in this case, repetition is only valid if all duplicate occurrences are used for the primary allocation of some key range.

In a system with the Mass Storage System, a cluster or component can be defined on a mass storage volume.

The VOLUMES parameter interacts with other DEFINE CLUSTER parameters. You should ensure that the volume(s) you specify for the cluster is consistent with the cluster's other attributes:

- SUBALLOCATION: If UNIQUE is not specified, the volume must already contain VSAM data space.
- KEYRANGES: If KEYRANGES and UNIQUE are specified, a VSAM data space is built and allocated on a separate volume for each key range.
- ORDERED: If ORDERED is specified, the volumes are used in the order listed. If ORDERED and KEYRANGES are specified, there is a one-to-one correspondence between the key ranges in the key range list and the volumes in the volser list.
- CYLINDERS, RECORDS, TRACKS: The volume(s) must contain enough unallocated space to satisfy the component's primary space requirement.
- FILE: The volume information supplied with the DD statement(s) pointed to by FILE must be consistent with the information specified for the cluster and its components.
- CATALOG: The data space on the volume must be defined before the cluster, and the catalog specified must own the volume.

Abbreviation: VOL

Optional Parameters

ATTEMPTS(number|**2)**
specifies the maximum number of times the operator can try to enter a correct password in response to a prompting message. This parameter only has effect when the entry's master password is not null. A prompting message is issued only when the user has not already supplied the appropriate password.

number
can be any number, 0 through 7. If 0 is specified, the operator is not prompted and is not allowed to enter a password from the console. number can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

Note to TSO Users: At a TSO terminal, the logon password is checked first, before the user is prompted to supply a password for the cluster. Checking the logon password counts as one attempt to obtain a password. If ATTEMPTS is not specified, the user has one attempt to supply the cluster's password, because the default is 2.

Abbreviation: ATT

AUTHORIZATION(entrypoint**[** string**])**
specifies that a user-security-verification routine (USVR) is available for additional security verification. When a protected cluster is accessed and the user supplies a correct password other than the cluster's master password, the USVR receives control. For details on the USVR, see VSAM Administration Guide.

If a USVR is loaded from an unauthorized library during access method services processing, an abnormal termination will occur.

This parameter only has effect when the entry's master password is not null.

entrypoint
specifies the name of the user's-security-verification routine.

string
specifies information to be passed on to the USVR when it receives control to verify authorization.

Abbreviation: AUTH

BUFFERSPACE(size**)**
specifies the minimum space to be provided for buffers. The buffer space size you specify 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.

size
is the amount of space to be provided for buffers. Size can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form, but must not exceed 16776704.

The size specified 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 specified size is less than VSAM requires for the buffers needed to run your job, VSAM terminates your DEFINE and provides an appropriate error message.

Abbreviations: BUFSP or BUFSPC

CATALOG(catname[/password]**)**
identifies the catalog in which the cluster is to be defined. For information about the order in which catalogs are selected, see "Order of Catalog Selection for DEFINE" on page 16.

catname
specifies the name of the catalog in which the entry is to be defined.

password
specifies the catalog's password. If the catalog is password protected, you must supply the update- or higher-level password. If no password is specified, VSAM will prompt the operator for the correct password. RACF: The update or higher RACF authority to the catalog is required.

If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved.

Abbreviation: CAT

CODE(code**)**
specifies a code name for the entry being defined. 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 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 cluster or component that is password protected without supplying a password, the operator will be prompted with the name of the entry.

This parameter only has effect when the cluster's or component's master password is not null.

CONTROLINTERVALSIZE(size**)**
specifies the size of the control interval for the cluster or component.

If CONTROLINTERVALSIZE is specified 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. If the size you specify is not an integer multiple of 512 or 2048, VSAM selects the next higher multiple for your cluster's control interval size.

If CONTROLINTERVALSIZE is not coded, VSAM determines the size of control intervals. If you have not specified BUFFERSPACE and the size of your records permits, VSAM uses the optimum size for the data component and 512 for the index component.

size
for a cluster's data component.

If SPANNED is not specified, 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, then VSAM will increase 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. The sizes you can specify (between 512 and 32768 bytes) are an integer multiple of 512 or 2048:

CISZ = (n x 512) or (n x 2048)

where n is a positive integer from 1 to 16.

size
for a key-sequenced cluster's index component.

You can specify the following values:

CISZ = [512 | 1024 | 2048 | 4096]

When you specify a size that is not a multiple of 512 or 2048, VSAM chooses the next higher multiple.

For a discussion of the relationship between control interval size and physical block size, refer to "Optimizing VSAM Performance" in VSAM Administration Guide. The discussion also includes restrictions that apply to control interval size and physical block size.

Abbreviations: CISZ or CNVSZ

CONTROLPW(password**)**
specifies a control password for the entry being defined.

When specified as a parameter of DATA or INDEX, the control password allows the user's program to open the data component or index component for read and write processing of the component's control intervals (that is, the entire control interval, including the data portion of stored records and the control fields VSAM inserts into stored records and control intervals).

If a read or update password is the only password specified for the object, it (the highest-level password) propagates upward and becomes the password for all higher unspecified levels.

Abbreviation: CTLPW

DESTAGEWAIT|NODESTAGEWAIT
specifies whether a cluster or its component that is stored on a mass storage volume is to be destaged synchronously or asynchronously with respect to the program that closes it.

Note: When the cluster or component is not stored on a mass storage volume, the attribute is ineffective until the direct access storage volume the cluster or component is stored on is converted to a mass storage volume by way of the CONVERTV command. (CONVERTV is described in OS/VS Mass Storage System (MSS) Services: Reference Information.)

DESTAGEWAIT
indicates that destaging is to be completed before VSAM returns control to the program that issued the CLOSE macro. VSAM can thus notify the program whether destaging was successful.

Abbreviation: DSTGW

NODESTAGEWAIT
indicates that notification of unsuccessful destaging is to be made only by a message to the operator.

Abbreviation: NDSTGW

When one of these parameters is specified for the data component and the other parameter is specified for the index component, the components are destaged separately as specified, unless the sequence set of the index component is embedded in the data. In that case, the parameter specified for the index component applies to both components.

ERASE|NOERASE
specifies whether the cluster's data component is to be erased when its entry in the catalog is deleted.

ERASE
specifies that the data component is to be overwritten with binary zeros when its catalog entry is deleted.

Abbreviation: ERAS

NOERASE
specifies that the data component is not to be overwritten with binary zeros.

Abbreviation: NERAS

EXCEPTIONEXIT(entrypoint)
specifies the name of a user-written routine, called the 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 direct access storage space.

An exception is any I/O error 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 will occur. See VSAM Administration Guide.

Abbreviation: EEXT

FILE(dname)
names the DD statement that identifies the direct access device(s) and volume(s) on which space is to be allocated to the cluster. 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.

When the cluster is defined in a recoverable catalog, and FILE is specified as a parameter of CLUSTER, the FILE parameter can identify the DD statement for all volumes on which space is to be allocated. The DD statement can be concatenated if the volumes are of different device types: Part of the concatenated DD statement describes the cluster's or component's volume(s) of one device type; the other part of the DD statement describes the volume that contains the catalog recovery area (of another device type).

If the FILE parameter is not specified, an attempt is made to dynamically allocate the required volumes. The

volume(s) must be mounted as permanently resident or reserved.

The DD statement you specify must be in the form:

```
//ddname DD UNIT=(devtype[,unitcount]),
//    VOL=SER=(volser1,volser2,volser,...),...
```

Note: 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 amount of space that is to be left free when the cluster is loaded and after any split of control intervals (CI-percent) and control areas (CA-percent).

The amount of 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.

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, which must be equal to or less than 100, can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

When you specify 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 be reserved when the data set is loaded.

When you define the cluster using the RECORDS parameter, the amount of free space specified is not taken into consideration in the calculations to determine primary allocation.

Abbreviation: FSPC

IMBED|NOIMBED
specifies whether the sequence set (the lowest level of the index) is to be placed with the data component.

IMBED
specifies that the sequence-set record for each control area is written as many times as it will fit on the first track adjacent to the control area. If the allocation is less than a cylinder, one track will be added to the primary and secondary allocation quantities.

Abbreviation: IMBD

NOIMBED
specifies that the sequence-set record for each control area is written with the other index records.

Abbreviation: NIMBD

The IMBED|NOIMBED parameter interacts with the REPLICATE|NOREPLICATE parameter in determining the physical

attributes of the index of the cluster. These index attributes can be used in the following combinations:

- The sequence set records are adjacent to the data control areas, and only the sequence set records are replicated (IMBED and NOREPLICATE).
- The sequence set records are adjacent to the data control intervals, and all levels of index records are replicated (IMBED and REPLICATE).
- All index records are together, and all index records are replicated (REPLICATE and NOIMBED).
- All index records are together, and no index records are replicated (NOREPLICATE and NOIMBED).

For some applications, specifying index options can improve the application's performance. For information on how the index's optional attributes affect performance, see VSAM Administration Guide.

INDEXED|NONINDEXED|NUMBERED
specifies the type of data organization that the cluster is to have.

When a cluster is defined, you specify whether the data is to be indexed (key-sequenced), nonindexed (entry sequenced), or numbered (relative record).

Certain parameters apply only to key-sequenced clusters, as noted in the description of each of these parameters.

If you do not specify the data organization and you also do not specify the MODEL parameter, your cluster will default to key-sequenced (indexed).

If you wish to define either an entry-sequenced or relative record cluster, you must specify the NONINDEXED or NUMBERED parameter unless you specify the MODEL parameter.

The data organization you select must be consistent with other parameters you specify.

INDEXED
specifies 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

NONINDEXED
specifies 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 is similar to an entry-sequenced cluster, and has fixed-length records that are stored in slots. Empty slots hold space for records to be added later. The data records are accessed by relative record number (slot number).

Abbreviation: NUMD

KEYRANGES((lowkey highkey)
[(lowkey highkey)...])

specifies that portions of key-sequenced data are to be placed on different volumes. Each portion of the data is called a key range. This parameter applies only to key-sequenced clusters.

The maximum number of key ranges is 123. Keyranges must be in ascending order, and are not allowed to overlap. A gap can exist between two key ranges, but you cannot insert records whose keys are within the gap. The space to be allocated for each key range must be contiguous.

Keys can contain 1 to 64 characters; 1 to 128 hexadecimal characters if coded as X'lowkey' X'highkey'.

lowkey
: specifies the low key of the key range. If lowkey is shorter than the actual keys, it will be padded on the right with binary zeros.

highkey
: specifies the high key of the key range. If highkey is shorter than the actual keys, it will be padded on the right with binary ones.

The KEYRANGES parameter interacts with other DEFINE CLUSTER parameters. You should ensure that your specification of KEYRANGES is consistent with the cluster's other attributes.

- VOLUMES: There should be as many volume serial numbers in the volser list as there are key ranges. When a volume serial number is duplicated in the volser list, more than one key range is allocated space on the volume.

  When more than one key range is to be contained on a volume, UNIQUE cannot be coded for the cluster's data component.

  When there are more volumes in the volser list than there are key ranges, the excess volumes are used for overflow records from any key range without consideration for key range boundaries.

  When there are fewer volumes in the volser list than there are key ranges, the excess key ranges are allocated on the last volume specified, and UNIQUE cannot also be specified.

- UNIQUE: When UNIQUE is specified, each key range resides on its own volume in its own VSAM data space. Other key ranges for the cluster cannot also reside on the volume.

- ORDERED: There is a one-to-one correspondence between the volumes in the volser list and the key ranges: The first volume on the volser list contains the first key range, the second volume contains the second key range, and so on.

  If a volume cannot be allocated in the order specified by the volser list, your cluster definition job terminates with an error message. (For example, the job would terminate if there were no space on one of the specified volumes.)

- KEYS: The low key and high key values must not exceed the key length specified in the KEY parameter. The KEY parameter must be specified in the same component as the KEYRANGE parameter.

Abbreviation: KRNG

KEYS(length offset|64 0**)**
specifies information about the prime key field of a key-sequenced data set's data records. This parameter applies only to key-sequenced clusters. The default specifies a key field of 64 bytes in length, 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. For more details on how to specify alternate indexes for a cluster, see "DEFINE ALTERNATEINDEX" on page 62.

When the data record is allowed to span 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. Length and offset can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

MASTERPW(password**)**
specifies a master password for the entry being defined. The master password allows all access method services operations against the cluster entry and its data and index entries, and allows the user's program to access the cluster's contents without restriction. For more details on how passwords can be used, see "Security Protection" in Catalog Administration Guide.

The AUTHORIZATION, CODE, and ATTEMPTS parameters have no effect unless the entry has a master password associated with it. If MASTERPW is not specified, the highest-level password specified becomes the password for all higher levels.

Abbreviation: MRPW

MODEL(entryname[/password]
[catname[/password]]**)**
specifies an existing entry to be used as a model for the entry being defined.

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 Catalog Administration Guide.

You may 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, specify only the entry type (cluster, data, or index) of the model to be used and the name of the entry to be defined.

Unless another entry is specified with the MODEL parameter as a subparameter of DATA or INDEX, 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.

entryname
specifies the name of the cluster or component entry to be used as a model.

password
specifies a password. If the model entry is password-protected and it is cataloged in a password-protected catalog, you must supply the read-

or higher-level password of either the model entry or its catalog. If both passwords are supplied, the catalog's password is used. RACF: The read or higher RACF authority to the model or catalog is required.

If you are not specifying new protection attributes for the cluster (that is, the model's passwords and protection attributes are being copied), you must supply the master password of either the model entry or its catalog. RACF: The alter RACF authority to the model or catalog is required.

catname
names the model entry's catalog. You identify the catalog that contains the model entry for either of these cases:

- You specify the catalog's password instead of the model entry's password.
- 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. For information about the order in which a catalog is selected when the catalog's name is not specified, see "Order of Catalog Selection for DEFINE" on page 16.

ORDERED|UNORDERED
specifies whether volumes are to be used in the order in which they are listed in the VOLUMES parameter.

ORDERED
specifies that the volumes are to be used in the order in which they are listed for the VOLUMES parameter.

When you want each key range to reside on a separate volume, you can use ORDERED so that the first key range goes on the first volume, the second key range goes on the second volume, and so on.

If ORDERED is specified and the volumes cannot be allocated in the order specified, the command is terminated. (For example, the command would terminate if there were no space on one of the volumes specified.)

Abbreviation: ORD

UNORDERED
specifies no order for the use of the volumes specified in the VOLUMES parameters.

Abbreviation: UNORD

OWNER(ownerid)
specifies the identification of the cluster's owner.

Note to TSO Users: If the owner is not identified with the OWNER parameter, the TSO user's userid becomes the ownerid.

READPW(password)
specifies a read password for the entry being defined. The read password permits read operations against the entry's records.

Abbreviation: RDPW

RECORDSIZE(average maximum|default)
specifies the average and maximum lengths, in bytes, of the records in the data component.

RECORDSIZE can be specified as a parameter of either CLUSTER or DATA.

average and maximum are integer values and can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. The minimum record size that you can specify is one byte.

For nonspanned records, the maximum record size + 7 cannot exceed the data component's control interval size (that is, the maximum nonspanned record size, 32 761 + 7 equals the maximum data component control interval size, 32 768).

For spanned records, the maximum record size cannot exceed the number of control intervals per control area times the quantity control interval size less 10, as calculated by VSAM. When you specify a record size that is larger than one control interval, you must also specify spanned records (SPANNED).

You can identify the records as fixed length by specifying the same value for average and maximum. If NUMBERED is also specified, the records must be fixed length (that is, average equals maximum).

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:

$$CISZ = (n \times RECSZ) + 10$$

or

$$n = \frac{(CISZ - 10)}{RECSZ}$$

When SPANNED or NUMBERED is specified:

$$CISZ = (n \times (RECSZ + 3)) + 4$$

or

$$n = \frac{(CISZ - 4)}{(RECSZ + 3)}$$

where:

- n is the number of fixed-length records in a control interval, and is a positive integer.
- CISZ is the control interval size (see CONTROLINTERVALSIZE parameter).
- RECSZ is the average record size.

default
: when SPANNED is specified, the default is RECORDSIZE(4086 32600). Otherwise, the default is RECORDSIZE(4089 4089).

Caution: When you specify RECORDS, you should ensure that:

$$REC(sec) \times RECSZ(avg) > RECSZ(max)$$

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.

Note: REPRO and EXPORT will not support data sets with record sizes greater than 32760.

Abbreviation: RECSZ

REPLICATE|NOREPLICATE
specifies how many times each index record is to be written on a track.

REPLICATE
specifies that each index record is to be written on a track as many times as it will fit. With REPLICATE, rotational delay is reduced and performance is improved. However, the cluster's index usually requires more direct access device space.

This parameter applies only to key-sequenced clusters.

Abbreviation: REPL

NOREPLICATE
specifies that the index records are to be written on a track only one time.

Abbreviation: NREPL

For a discussion of the relationship between IMBED|NOIMBED and REPLICATE|NOREPLICATE, see the description of the IMBED|NOIMBED parameter.

REUSE|NOREUSE
specifies whether the cluster can be opened again and again as a reusable cluster.

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 allows you to 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) or with its own data space (see the UNIQUE parameter). Reusable data sets may be multivolume and are restricted to 16 physical extents per volume.

Abbreviation: RUS

NOREUSE
specifies that the cluster cannot be opened again as a new cluster.

Abbreviation: NRUS

SHAREOPTIONS(crossregion[crosssystem]|1 3)
specifies how a component or cluster can be shared among users. For a description of data set sharing, see VSAM Administration Guide. 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 using GRS can access a VSAM data set concurrently. The values that can be specified are:

1
specifies that 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. With this option, VSAM ensures complete data integrity for the data set.

2
specifies that 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.

With this option, VSAM ensures write integrity by obtaining exclusive control for a control interval when it is to be updated. If a user desires read integrity, it is that user's responsibility to use the ENQ and DEQ macros appropriately to provide read integrity for the data the program obtains. (For information on using ENQ and DEQ, see System Macros and Facilities.)

3
specifies that the data set can be fully shared by any number of users. With this option, each user 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 data set failures, and other unpredictable results. This option places heavy responsibility on each user sharing the data set.

4
specifies that the data set can be fully shared by any number of users and buffers used for direct processing are refreshed for each request.

This option requires your program to use the ENQ and DEQ macros to maintain data integrity while sharing the data set. Improper use of the ENQ macro can cause problems similar to those described under SHAREOPTIONS 3. (For information on using ENQ and DEQ, see System Macros and Facilities.)

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 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 values that can be specified 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 failures, and other unpredictable results. This option places heavy responsibility on each user sharing the data set.

4

specifies that the data set can be fully shared. Buffers used for direct processing are refreshed for each request.

This option requires that you use the RESERVE and DEQ macros to maintain data integrity while sharing the data set. (For information on using RESERVE and DEQ, see System Macros and Facilities.) Writing is limited to PUT-update and PUT-insert processing that does not change the high-used RBA if your program opens the data set with DISP=SHR.

Data set integrity cannot be maintained unless all jobs accessing the data set in a cross-system environment specify DISP=SHR. Improper use of the RESERVE macro can cause problems similar to those described under SHAREOPTIONS 3.

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 results in the appropriate buffer(s) being written immediately to the VSAM cluster's direct access device 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 results in all the user's input buffers being refreshed. The contents of each data and index buffer being used by the user's program are retrieved from the VSAM cluster's direct access device.

Abbreviation: SHR

SPANNED|NONSPANNED

specifies whether a data record is allowed to cross control interval boundaries.

SPANNED

specifies that, if the maximum length of a data record (as specified with RECORDSIZE) is larger than a control interval, the record will be contained on more than one control interval. This allows VSAM to select a control interval size that is optimum for the direct access device.

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.

Abbreviation: SPND

NONSPANNED
specifies that the record must be contained in one control interval. VSAM will select a control interval size that accommodates your largest record.

Abbreviation: NSPND

SPEED|RECOVERY
specifies whether storage allocated to the data component is to be preformatted before records are inserted. SPEED|RECOVERY applies only to initial loading.

When you specify RECOVERY, your initial load takes longer because the control areas are written initially with end-of-file indicators and again with your data records. When you specify SPEED, your initial load is quicker.

SPEED
specifies that the data component's space is not preformatted. Its space might contain data records from a previous use of the space, or it might contain binary zeros (its contents are unpredictable).

If the initial load fails, you must load the data records again from the beginning, because VSAM is unable to determine where your last correctly written record is. (VSAM cannot find a valid end-of-file indicator when it searches your data records.)

RECOVERY
specifies that the data component's control areas are written with records that indicate end of file. When a data record is written (during the initial load) into a control interval, it is always followed by a record that identifies the record that has just been written as the last record in the cluster.

If the initial load fails, you can resume loading data records after the last correctly written data record, because an end-of-file indicator identifies it as the last record.

Abbreviation: RCVY

STAGE|BIND|CYLINDERFAULT
specifies how a cluster or component that is stored on a mass storage volume is to be staged.

When the cluster or component is not stored on a mass storage volume, the attribute is ineffective until the direct access storage volume on which the cluster or component is stored is converted to a mass storage volume. The CONVERTV command used in this conversion is described in OS/VS Mass Storage System (MSS) Services: Reference Information.

When one of these parameters is specified for the data component and another parameter is specified for the index component, the components are staged separately as specified, except when the sequence set of the index component is embedded in the data. In that case, the parameter specified for the index component applies to both components.

STAGE
indicates that the cluster or component is to be staged from mass storage to a direct access storage staging drive when the cluster or component is opened. If the cluster or component cannot be staged at open time because of heavy staging activity of other objects, data is staged as the processing program needs it.

BIND
indicates that the cluster or component is not only to be staged, but also to be bound; that is, it is to be retained on the direct access storage staging drive until it is closed. If the cluster or component cannot be staged at open time because of heavy staging activity of other objects, data is staged as the processing program needs it.

CYLINDERFAULT
indicates that the cluster or component is not to be staged when it is opened, but that data from it is to be staged as the processing program needs it.

Abbreviation: CYLF

TO(date)|FOR(days)
specifies the retention period for the cluster being defined. If neither TO nor FOR is specified, the cluster can be deleted at any time.

TO(date)
specifies the date, in the form yyddd, where yy is the year and ddd is the number (001 through 365) of the day through which the cluster being defined is to be kept.

FOR(days)
specifies the number of days for which the cluster being defined is to be kept. The maximum number that can be specified is 9999. If the number specified is 0 through 1830, the cluster is retained for the number of days specified; if the number is between 1831 and 9999, the cluster is retained through the year 1999.

days
can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

UNIQUE|SUBALLOCATION
specifies whether the cluster's components are allocated an amount of space from the volume's available space (UNIQUE) or from a VSAM data space's available space (SUBALLOCATION).

UNIQUE
specifies that a VSAM data space is to be built and assigned exclusively to each component of the cluster. The data space is created when the cluster is defined. The cluster's volume(s) must be mounted. VSAM builds a DSCB in the volume's table of contents (VTOC) to describe the data space. The name of the data space, which is the same as the component's name, is put in the DSCB. A subentry is added to the volume entry (in the VSAM catalog) to describe the VSAM data space.

Abbreviation: UNQ

SUBALLOCATION
specifies that space from one of the VSAM data spaces on the volume is assigned to the cluster's components.

Abbreviation: SUBAL

The space allocation attribute interacts with other DEFINE CLUSTER parameters. You should ensure that the space allocation attribute you specify for the cluster is consistent with other attributes:

- REUSE: You cannot specify REUSE when you specify UNIQUE for a cluster or its components.

- KEYRANGES: When UNIQUE is specified, a data space is built and allocated for each key range. Each key range is on a separate volume.

- VOLUMES: When UNIQUE is not specified, VSAM data space must exist on the volume that is to contain the cluster's component. When UNIQUE is specified, and more than one volume is specified, VSAM must already own all the volumes except the first. If there is no VSAM space on a volume, you must execute a DEFINE SPACE CANDIDATE before your DEFINE UNIQUE.

UPDATEPW(password**)**
specifies an update password for the entry being defined. The update password permits read and write operations against the entry's records.

If a read password is the only password specified for the object (that is, it is the highest-level password), it propagates upward and becomes the password for all higher levels. If you specify a higher-level password and do not specify an update password, the update password is null.

Abbreviation: UPDPW

WRITECHECK|NOWRITECHECK
specifies whether the cluster or component is to be checked by a machine action called write check when a record is written into it.

WRITECHECK
specifies that a record is written and then read, without data transfer, to test for the data check condition.

Abbreviation: WCK

NOWRITECHECK
specifies that the cluster or component is not to be checked by a write check.

Abbreviation: NWCK

DATA AND INDEX COMPONENTS OF A CLUSTER

Attributes can be specified separately for the cluster's data and index components. The DATA and INDEX parameters are listed 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 a Key-Sequenced Cluster Specifying Data and Index Parameters: Example 1

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, if the cluster is being defined in a VSAM catalog, a VSAM data space exists on volume VSER02. It also assumes that an alias name D40 has been defined for the catalog D27UCAT1. This naming convention causes D40.MYDATA to be cataloged in D27UCAT1.

```
//DEFCLU1  JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE CLUSTER -
               (NAME(D40.MYDATA) -
               VOLUMES(VSER02) -
               RECORDS(1000 500)) -
          DATA -
               (NAME(D40.KSDATA) -
               KEYS(15 0) -
               RECORDSIZE(250 250) -
               FREESPACE(20 10) -
               BUFFERSPACE(25000)  ) -
          INDEX -
               (NAME(D40.KSINDEX) -
               IMBED) -
          CATALOG (D27UCAT1/USERUPPW)
/*
```

The DEFINE CLUSTER command builds a cluster entry, a data entry, and an index entry to define the key sequenced cluster D40.MYDATA. The parameters specified for the cluster as a whole are:

- NAME, which specifies that the cluster's name is D40.MYDATA.
- VOLUMES, which specifies that the cluster is to reside on volume VSER02.
- RECORDS, which specifies that the cluster's space allocation is 1000 data records. When the cluster is extended, it 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 D40.MYDATA are:

- NAME, which specifies that the data component's name is D40.KSDATA.
- KEYS, which specifies 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, which specifies fixed-length records of 250 bytes.
- BUFFERSPACE, which specifies 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, which 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 D40.MYDATA are:

- NAME, which specifies that the index component's name is D40.KSINDEX.
- IMBED, which specifies that sequence-set index records are to be placed in the data component's control areas (the sequence-set records will be replicated automatically).

The parameter which provides the catalog's password is:

- CATALOG, which specifies the catalog name and its update password. This parameter would not be necessary if (1) the catalog was not password protected, and (2) D40 was defined as the catalog's alias.

Define a Key-Sequenced Cluster and an Entry-Sequenced Cluster: Example 2

In this example, two VSAM clusters are defined. The first DEFINE command defines a key sequenced VSAM cluster, D40.EXAMPLE.KSDS1. The second DEFINE command defines an entry-sequenced VSAM cluster, D50.EXAMPLE.ESDS1. In both examples, it is assumed that alias names, D40 and D50, have been defined for user catalogs D27UCAT1 and D27UCAT2, respectively, and that neither user catalog is password-protected. This example assumes that VSAM data space that can contain the data sets already exists on volumes VSER02 and VSER03.

```
//DEFCLU2  JOB     ...
//STEP1    EXEC    PGM=IDCAMS
//SYSPRINT DD      SYSOUT=A
//SYSIN    DD      *
     DEFINE CLUSTER -
               (NAME(D40.EXAMPLE.KSDS1) -
               MODEL(D40.MYDATA) -
               VOLUMES(VSER02) -
               NOIMBED  )

     DEFINE CLUSTER -
               (NAME(D50.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 D40.EXAMPLE.KSDS1. Its parameters are:

- NAME, which specifies the name of the key sequenced cluster, D40.EXAMPLE.KSDS1. The cluster will be defined in the user catalog for which D40 has been established as an alias.
- MODEL, which identifies D40.MYDATA as the cluster to use as a model for D40.EXAMPLE.KSDS1. The attributes and specifications of D40.MYDATA that are not otherwise specified with the DEFINE command's parameters are used to define the attributes and specifications of D40.EXAMPLE.KSDS1. D40.MYDATA is located in the user catalog for which D40 has been established as an alias.
- VOLUMES, which specifies that the cluster is to reside on volume VSER02.

- NOIMBED, which 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, D50.EXAMPLE.ESDS1. Its parameters are:

- NAME, which specifies the name of the entry-sequenced cluster, D50.EXAMPLE.ESDS1. The cluster will be defined in the user catalog for which D50 has been established as an alias.
- RECORDS, which specifies that the cluster's space allocation is 100 records. When the cluster is extended, it is extended in increments of 500 records.
- RECORDSIZE, which specifies that the cluster's records are fixed length (the average record size equals the maximum record size) and 250 bytes long.
- VOLUMES, which specifies that the cluster is to reside on volume VSER03.
- NONINDEXED, which specifies that the cluster is to be an entry-sequenced cluster.

Define a Key-Sequenced Cluster (in a Unique Data Space) in a VSAM Catalog: Example 3

In this example, a key sequenced cluster is defined. The cluster is unique; that is, it is the only cluster in a VSAM data space.

```
//DEFCLU3  JOB     ...
//STEP1    EXEC    PGM=IDCAMS
//SYSPRINT DD      SYSOUT=A
//VOL4     DD      VOL=SER=VSER04,UNIT=DISK,DISP=OLD
//SYSIN    DD      *
     DEFINE CLUSTER -
          (NAME(ENTRY) -
          RECORDSIZE(80 80) -
          FILE(VOL4) -
          KEYS(10 10) -
          VOLUMES(VSER04) -
          UNIQUE -
          CYLINDERS(5 10)  ) -
       CATALOG(USERCAT3)
/*
```

The DEFINE CLUSTER command builds a cluster entry, a data entry, and an index entry to define the key sequenced cluster ENTRY. The DEFINE CLUSTER command also allocates a data space and allocates it for the cluster's exclusive use. The command's parameters are:

- NAME, which specifies the cluster's name is ENTRY.
- RECORDSIZE, which specifies that the records are fixed length, 80-byte records.
- FILE, which specifies the name of a DD statement that describes the cluster's volume and causes it to be mounted. The volume must be mounted so that VSAM can record the format-1 DSCBs of the key sequenced cluster in the volume's VTOC. If FILE is not specified, an attempt is made to dynamically allocate the volume.
- KEYS, which specifies that the length of the key field is 10 bytes and that the key field begins in the 11th byte (byte 10) of each data record.

- VOLUMES and UNIQUE, which specify that ENTRY is to reside alone in a data space on volume VSER04. Access method services will create two data spaces for the cluster: one for the data component, and one for the index. Both data spaces are to be on volume VSER04. This example assumes that volume VSER04 has enough available space to contain the new data space. This example also assumes that either the volume's entry is in the MYCAT catalog or that volume VSER04 is not owned by a VSAM catalog at the beginning of the job.
- CYLINDERS, which specifies that five cylinders are allocated for the cluster's data space. When the cluster's data or index component is extended, the component is to be extended in increments of 10 cylinders.
- CATALOG, which specifies the name of the catalog into which the cluster will be defined. In this example, USERCAT3 is not password-protected and an attempt will be made to dynamically allocate it.

Define a Relative Record Cluster in a Catalog: Example 4

In this example, a relative record cluster is defined. The cluster is suballocated (it can reside in a VSAM data space with other VSAM objects). Volume VSER01 does not have to be mounted or allocated at this time.

```
//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/USERUPPW)
/*
```

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 also obtains space for the cluster from the VSAM data space on volume VSER01, and allocates 10 tracks for the cluster's use. The command's parameters are:

- NAME, which specifies that the cluster's name is EXAMPLE.RRDS1.
- RECORDSIZE, which specifies that the records are fixed length, 100-byte records. Average and maximum record length must be equal for a relative record data set.
- VOLUMES, which 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, which specifies that ten tracks are to be allocated for the cluster. When the cluster is extended, it is to be extended in increments of 5 tracks.
- NUMBERED, which specifies that the cluster's data organization is to be relative record.
- CATALOG, which supplies the user catalog's update password, USERUPPW.

Define a Reusable Entry-Sequenced Cluster in a Catalog: Example 5

In this example, a reusable entry-sequenced cluster is defined. The cluster can be used as a temporary data set. Each time the cluster is opened, its high-used RBA can be reset to zero. The cluster is suballocated (that is, it can reside in a VSAM data space with other VSAM objects).

```
//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 -
              MASTERPW(ESD2MRPW) -
              UPDATEPW(ESD2UPPW)  ) -
            CATALOG(D27UCAT2/USERMRPW)
/*
```

The DEFINE CLUSTER command builds a cluster entry and a data entry to define the entry-sequenced cluster, EXAMPLE.ESDS2. The DEFINE CLUSTER command also obtains space for the cluster from one of the Class 1 VSAM data spaces on volume VSER03, and assigns ten tracks for the cluster's use. VSER03 does not have to be mounted or allocated at this time. The command's parameters are:

- NAME, which specifies that the cluster's name is EXAMPLE.ESDS2.
- RECORDSIZE, which specifies that the records are variable-length, with an average size of 2500 bytes and a maximum size of 3000 bytes.
- SPANNED, which specifies that data records can cross control interval boundaries.
- VOLUMES, which specifies that the cluster is to reside on volume VSER03. This example assumes that the volume is already cataloged in the user catalog, D27UCAT2.
- CYLINDERS, which 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 one cylinder.
- NONINDEXED, which specifies that the cluster's data organization is to be entry sequenced. This parameter overrides the INDEXED parameter.
- REUSE, which 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.
- MASTERPW and UPDATEPW, which specifies the master password, ESD2MRPW, and the update password, ESD2UPPW, for the cluster.
- CATALOG, which specifies that the cluster is to be defined in a user catalog, D27UCAT2. The example also supplies the user catalog's master password, USERMRPW.

Define a Key-Sequenced Cluster in a Catalog: Example 6

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, USERRCAT.

```
//DEFCLU6  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
      DEFINE CLUSTER -
             (NAME(EXAMPLE.KSDS2)) -
          DATA -
             (MASTERPW(DAT2MRPW) -
             UPDATEPW(DAT2UPPW) -
             READPW(DAT2RDPW) -
             RECORDS(500 100) -
             EXCEPTIONEXIT(DATEXIT) -
             ERASE -
             FREESPACE(20 10) -
             KEYS(6 4) -
             RECORDSIZE(80 100) -
             VOLUMES(VSER01) ) -
          INDEX -
             (MASTERPW(IND2MRPW) -
             UPDATEPW(IND2UPPW) -
             READPW(IND2RDPW) -
             RECORDS(300 300) -
             IMBED -
             VOLUMES(VSER01) ) -
          CATALOG(USERCAT/USERUPPW)
/*
```

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 also obtains space for the cluster from one of the VSAM data spaces on volume VSER01, and allocates space separately for the cluster's data and index components. Because the cluster is being defined into a recoverable catalog, an attempt will be made to dynamically allocate the catalog recovery area on VSER01.

The command's parameter that applies to the cluster is:

- NAME, which specifies that the cluster's name is EXAMPLE.KSDS2.

The command's parameters that apply only to the cluster's data component are enclosed in the parentheses following the DATA keyword:

- MASTERPW, UPDATEPW, and READPW, which specify the data component's master password, DAT2MRPW, update password, DAT2UPPW, and read password, DAT2RDPW.
- RECORDS, which 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, which 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, which specifies that the cluster's data is to be erased (overwritten with binary zeros) when the cluster is deleted.
- FREESPACE, which 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, which specifies the location and length of the key field in each data record. The key field is 6 bytes long and begins in the 5th byte (byte 4) of each data record.
- RECORDSIZE, which specifies that the cluster's records are variable length, with an average size of 80 bytes and a maximum size of 100 bytes.
- VOLUMES, which specifies that the cluster is to reside on volume VSER01. This example assumes that the volume is already cataloged in the catalog, USERCAT.

The command's parameters that apply only to the cluster's index component are enclosed in the parentheses following the INDEX keyword:

- MASTERPW, UPDATEPW, and READPW, which specifies the index component's master password, IND2MRPW, update password, IND2UPPW, and read password, IND2RDPW.
- RECORDS, which 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.
- IMBED, which specifies that the index's sequence set records are to be placed in the data component's control areas (the sequence set records will be replicated automatically).
- VOLUMES, which specifies that the index component is to reside on volume VSER01.

The CATALOG parameter supplies the catalog's update password.

Define an Entry-Sequenced Cluster Using a Model: Example 7

In this example, two entry-sequenced clusters are defined. The attributes of the second cluster defined is modeled from the first cluster.

```
//DEFCLU7  JOB     ...
//STEP 1   EXEC    PGM=IDCAMS
//STEPCAT  DD      DSNAME=USERCAT4,DISP=SHR
//SYSPRINT DD      SYSOUT=A
//SYSIN    DD      *
     DEFINE CLUSTER -
               (NAME(GENERIC.A.BAKER) -
               VOLUMES(VSER02) -
               RECORDS(100 100) -
               RECORDSIZE(80 80) -
               NONINDEXED ) -
          CATALOG(USERCAT4/USERMRPW)

     DEFINE CLUSTER -
               (NAME(GENERIC.B.BAKER) -
               MODEL(GENERIC.A.BAKER USERCAT4)) -
          CATALOG(USERCAT4/USERMPRW)
/*
```

Job control language statement:

- STEPCAT DD, which makes a catalog available for this job step: USERCAT4.

The first DEFINE CLUSTER command defines an entry-sequenced cluster, GENERIC.A.BAKER. Its parameters are:

- NAME, which specifies the name of the entry-sequenced cluster, GENERIC.A.BAKER.

- VOLUMES, which specifies that the cluster is to reside on volume VSER02.
- RECORDS, which specifies that the cluster's space allocation is 100 records. When the cluster is extended, it is extended in increments of 100 records.
- RECORDSIZE, which specifies that the cluster's records are fixed length (the average record size equals the maximum record size) and 80 bytes long.
- NONINDEXED, which specifies that the cluster is entry sequenced.
- CATALOG, which specifies that the cluster is to be defined in the USERCAT4 catalog. The master password of USERCAT4 is USERMRPW.

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. Its parameters are:

- NAME, which specifies the name of the entry-sequenced cluster, GENERIC.B.BAKER.
- MODEL, which identifies GENERIC.A.BAKER, cataloged in user catalog D27UCAT1, 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, which specifies that the cluster is to be defined in the USERCAT4 catalog. The master password of USERCAT4 is USERMRPW.

DEFINE GENERATIONDATAGROUP

DEFINE GENERATIONDATAGROUP

The DEFINE GENERATIONDATAGROUP command creates a catalog entry for a generation data group. The format of this command is:

```
DEFINE   GENERATIONDATAGROUP
           (NAME(entryname)
           LIMIT(limit)
           [EMPTY|NOEMPTY]
           [OWNER(ownerid)]
           [SCRATCH|NOSCRATCH]
           [TO(date)|FOR(days)])

         [CATALOG(catname[/password])]
```

DEFINE can be abbreviated: DEF

DEFINE GENERATIONDATAGROUP PARAMETERS

Required Parameters

GENERATIONDATAGROUP
specifies that a generation data group entry is to be defined.

Abbreviation: GDG

NAME(entryname)
specifies the name of the generation data group that is being defined.

LIMIT(limit)
specifies the maximum number, from 1 to 255, of generation data sets that can be associated with the generation data group to be defined. number can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

Abbreviation: LIM

Optional Parameters

CATALOG(catname[/password])
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. For information about the order in which a catalog is selected when the catalog's name is not specified, see "Order of Catalog Selection for DEFINE" on page 16.

catname
specifies the name of the catalog.

password
specifies the catalog's password. If the catalog is password protected, you must supply the update- or higher-level password. RACF: The update or higher RACF authority to the catalog is required.

Abbreviation: CAT

EMPTY|NOEMPTY
specifies what action is to be taken when the maximum number of generation data sets for the generation data group has been reached and another generation data set is to be cataloged. The disposition of the data set's DSCB in the volume's VTOC is determined with the SCRATCH|NOSCRATCH parameter.

EMPTY
specifies that all the generation data sets are to be uncataloged when the maximum is reached (each data set's non-VSAM entry is automatically deleted from the catalog).

Abbreviation: EMP

NOEMPTY
specifies that only the oldest generation data set is to be uncataloged when the maximum is reached.

Abbreviation: NEMP

OWNER(ownerid**)**
identifies the generation data set's owner.

Note to TSO users: If the owner is not identified with the OWNER parameter, the TSO userid is the default ownerid.

SCRATCH|NOSCRATCH
specifies whether 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 a result of EMPTY|NOEMPTY, or specifically, as a result of a user issued DELETE request). The user can override the SCRATCH|NOSCRATCH attribute when issuing the DELETE command.

SCRATCH
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.

Note: On MSS volumes, even if SCRATCH is specified, the data set will not be scratched when it is automatically uncataloged.

Abbreviation: SCR

NOSCRATCH
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. However, your program can process the data set as it processes any other non-VSAM data sets—that is, by using a JCL DD statement to describe the data set and allocate it.

Abbreviation: NSCR

TO(date**)|FOR(**days**)**
specifies the retention period for the generation data group being defined.

TO(date**)**
specifies the date, in the form yyddd, where yy is the year and ddd is the number (001 through 365) of the day through which the generation data group being defined is to be kept.

FOR(days)
specifies the number of days for which the generation data group being defined is to be kept. This value can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n').

The maximum number that can be specified is 9999. If the number specified is 0 through 1830, the generation data group is retained for the number of days specified; if the number is between 1831 and 9999, the generation data group is retained through the year 1999. If neither TO nor FOR is specified, the generation data group can be deleted at any time.

DEFINE GENERATIONDATAGROUP EXAMPLE

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 generation data group 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 generation data group. When the scheduler processes the DD statement, no space is allocated to GDG01.

  The model DSCB must exist on the generation data group's catalog volume.

Because no catalog was specified with a JOBCAT or STEPCAT DD statement or with the CATALOG parameter, VSAM assumes all entries built for this command sequence are to be cataloged in the master catalog.

The DEFINE GENERATIONDATAGROUP command defines a generation data group base catalog entry, GDG01. Its parameters are:

- NAME, which specifies the name of the generation data group, GDG01. Each generation data set in the group will have the name GDG01.GxxxxVyy, where "xxxx" is the generation number and "yy" is the version number.
- EMPTY, which 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 generation data set is added to the group.

- NOSCRATCH, which 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, which specifies that the maximum number of generation data sets in the group is 255. The LIMIT parameter is required.

The second job, DEFGDG2, is used to allocate space and catalog a generation data set in the newly defined generation data group.

Job control statement:

- GDGDD1 DD, which specifies a generation data set in the generation data group.

DEFINE NONVSAM

The DEFINE NONVSAM command defines a catalog entry for a non-VSAM data set. The format of this command is:

```
DEFINE    NONVSAM
            (NAME(entryname)
            DEVICETYPES(devtype[  devtype...])
            VOLUMES(volser[  volser...])
            [FILESEQUENCENUMBERS(number[  number..])]
            [OWNER(ownerid)]
            [TO(date)|FOR(days)])

          [CATALOG(catname[/password])]
```

DEFINE can be abbreviated: DEF

DEFINE NONVSAM PARAMETERS

Required Parameters

NONVSAM
specifies that a non-VSAM data set is to be defined.

Abbreviation: NVSAM

NAME(entryname)
specifies the name of the non-VSAM data set 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 generation data set with its generation data group 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 a generation data set and attach it to a generation data group. If the containing catalog is RACF defined, then the update or higher RACF authority to the generation data group is required.

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 device type for any device that is supported by your system. You can specify a generic device type (that is, 3380, 3375, 3330). You can only specify a device type named by the user (such as SYSDA or TAPE) if the name was established in the system device name table during system generation.

CAUTION: When a device type named by the user is specified, the device type put in the catalog record is the unique device type generated for that unit name in the system device name table during SYSGEN. This unique device type is a function of the order of that unit name in the Stage 1 SYSGEN deck. Allocation errors will result if a data set, cataloged with a device type named by the user, is referred to on a system where SYSGEN generated the unit names differently.

If the catalog is shared between systems, this device type may be different if the unit names were not ordered the

same in all Stage 1 SYSGEN decks. An I/O GEN may also change this device type if the order of unit names in the Stage 1 SYSGEN deck is different.

See the discussion of the SCRATCH|NOSCRATCH option of the DELETE command for a restriction on deleting data sets that have been defined with a device type named by the user.

If you expect to change the device type of the system residence volume, you can code DEVICETYPES(0000) and this field will be resolved at SUPERLOCATE, 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.

Abbreviation: DEVT

VOLUMES(volser[volser...]**)**
specifies the volumes to contain the non-VSAM data set. In a system with the Mass Storage System, a non-VSAM data set can be defined on a mass storage volume. You can specify more than one volume serial number if the data set resides on many volumes.

If the data set resides on magnetic tape and more than one file belongs to the data set on a single tape volume, you must repeat the volume's serial number in order to maintain a one to one correspondence between the volume serial numbers and the file sequence numbers.

For example, if your non-VSAM data set is contained in the first three files of magnetic tape volume TAPE10, you specify:

```
DEVICETYPES(3400-6 3400-6 3400-6) -
VOLUMES(TAPE10 TAPE10 TAPE10) -
FILESEQUENCENUMBERS(1 2 3) -
```

If you expect to change the serial number of the system residence volumes, you can code VOLUMES('******') and this field will be resolved at SUPERLOCATE, LOCATE, and DELETE time to that number. This will allow you to use the non-VSAM data sets without having to recatalog them to point to the new volume.

RACF commands can be used 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.

Abbreviation: VOL

Optional Parameters

CATALOG(catname[/password]**)**
identifies the catalog in which the non-VSAM data set is to be defined.

For information about the order in which a catalog is selected when the catalog's name is not specified, see "Order of Catalog Selection for DEFINE" on page 16.

catname
specifies the name of the catalog in which the entry is to be defined.

password
specifies the catalog's password. If the catalog is password protected, you must supply the update- or

higher-level password. RACF: The update or higher RACF authority to the catalog is required.

Abbreviation: CAT

FILESEQUENCENUMBERS(number[number...]**)**
specifies the file sequence number of the non-VSAM data set being defined. This number indicates the position of the file being defined with respect to other files of the tape. If the data set spans volumes, the file sequence number on each volume must be specified. The numbers must be specified in the same order as the volumes in the VOLUMES parameter.

For example, if your non-VSAM data set is contained in the first three files of magnetic tape volume TAPE10, you specify:

```
DEVICETYPES(3400-6 3400-6 3400-6) -
VOLUMES(TAPE10 TAPE10 TAPE10) -
FILESEQUENCENUMBERS(1 2 3) -
```

number
can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

Abbreviation: FSEQN

OWNER(ownerid**)**
identifies the owner of the non-VSAM data set.

Note to TSO users: If OWNER is not specified, the TSO userid is the default ownerid.

TO(date**)|FOR(**days**)**
specifies the retention period for the non-VSAM data set being defined. The non-VSAM data set is not automatically deleted when the expiration date is reached. When you do not specify a retention period, the non-VSAM data set can be deleted at any time. The expiration date will be placed in the catalog, but not in the format-1 DSCB.

TO(date**)**
specifies the date, in the form yyddd, where yy is the year and ddd is the Julian date (001, for January 1, through 365, for December 31), through which the non-VSAM data set is to be kept before it is allowed to be deleted.

FOR(days**)**
specifies the number of days for which the non-VSAM data set is to be kept before it is allowed to be deleted.

The maximum number that can be specified for days is 9999. If the number specified is 0 through 1830, the retention period is the number of days specified. If the number specified is between 1831 and 9999, the retention period is through the year 1999. days can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

DEFINE NONVSAM EXAMPLE

Define a Non-VSAM Data Set: Example 1

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/USERMRPW)
/*
```

Both DEFINE NONVSAM commands define a non-VSAM data set in catalog USERCAT4. The DEFINE NONVSAM commands' parameters are:

- NAME, which specifies the name of the non-VSAM data sets, EXAMPLE.NONVSAM and EXAMPLE.NONVSAM2.
- DEVICETYPES, which specifies the type of device that contains the non-VSAM data sets, an IBM 3380-1 Direct Access Storage Device.
- VOLUMES, which specifies the volume that contains the non-VSAM data sets, VSER02.
- CATALOG, which identifies the catalog that is to contain the non-VSAM entries, USERCAT4, and its update- or higher-level password, USERMRPW.

DEFINE PAGESPACE

The DEFINE PAGESPACE command defines an entry for a page space data set. The format of this command is:

```
DEFINE    PAGESPACE
            (NAME(entryname)
            {CYLINDERS(primary)|
              RECORDS(primary)|
              TRACKS(primary)}
            VOLUME(volser)
            [ATTEMPTS(number|2)]
            [AUTHORIZATION(entrypoint[  string])]
            [CODE(code)]
            [CONTROLPW(password)]
            [FILE(ddname)]
            [MASTERPW(password)]
            [MODEL(entryname[/password]
              [  catname[/password]])]
            [OWNER(ownerid)]
            [READPW(password)]
            [SWAP|NOSWAP]
            [TO(date)|FOR(days)]
            [UNIQUE|SUBALLOCATION]
            [UPDATEPW(password)])

          [CATALOG(catname[/password])]
```

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)|
RECORDS(primary)|
TRACKS(primary)
specifies the amount of space that is to be allocated by tracks, cylinders, or number of records. If RECORDS or TRACKS is specified, the quantity specified is rounded up to the nearest cylinder and the space is allocated in cylinders.

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, RECORDS, or TRACKS.

When you specify UNIQUE, and the page space's data space is the first data space on the volume that belongs to a recoverable catalog, an additional amount (the equivalent of one cylinder) is allocated for the recovery area data space.

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.

primary can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

Abbreviations: CYL, REC, and TRK

VOLUMES(volser**)**
specifies the volume that contains the page space.

In a system with the Mass Storage System, a page space cannot reside on a mass storage volume.

Unless you specify the MODEL parameter, you must specify the VOLUMES parameter.

The VOLUMES parameter interacts with other DEFINE PAGESPACE parameters. You should ensure that the volume(s) you specify for the page space are consistent with the page space's other attributes:

- SUBALLOCATION: If suballocation is specified, the volume must contain a VSAM data space.
- CYLINDERS, RECORDS, TRACKS: The volume must contain enough unallocated space to satisfy the page space's space requirement.
- FILE: The volume information supplied with the DD statement pointed to by FILE must be consistent with the information specified for the page space.
- CATALOG: If the page space is suballocated, the data space on the volume must have been defined in the same catalog as the page space, and must be owned by the catalog.

Abbreviation: VOL

Optional Parameters

CATALOG(catname**[/**password**])**
specifies the name and password of the catalog in which the page space is to be defined.

When the CATALOG parameter identifies a user catalog, you must also supply a STEPCAT or JOBCAT DD statement to describe and allocate the user catalog. For information about the order in which catalogs are selected, see "Order of Catalog Selection for DEFINE" on page 16.

catname
specifies the name of the catalog.

password
specifies a password. If the catalog is password-protected, you must supply the catalog's update- or higher-level password. If no password is specified, VSAM may ask the operator or TSO terminal user for the correct password. RACF: The update or higher RACF authority to the catalog is required.

Abbreviation: CAT

ATTEMPTS(number**|2)**
specifies the maximum number of times the operator or TSO terminal user can try to enter a correct password in response to a prompting message. A prompting message is issued only when the user has not already supplied the appropriate password. When you define a page space, you should specify ATTEMPTS(0), so that the operator is not prompted and is not allowed to enter a password from the console. This parameter only has effect when the entry's master password is not null.

number
can be any number from 0 to 7. number can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

Note to TSO users: At a TSO terminal, the logon password is checked first before the user is prompted to supply a password for the page space. Checking the logon password counts as one attempt to obtain a password. If ATTEMPTS is not specified, the user has one attempt to supply the page space's password, because the default is 2.

Abbreviation: ATT

AUTHORIZATION(entrypoint[string]**)**
specifies that a user-security-verification routine (USVR) is available for additional security verification. When a protected page space is accessed and the user supplies a correct password other than the page space's master password, the USVR receives control. For details on the USVR, see Catalog Administration Guide.

If a USVR is loaded from an unauthorized library during access method services processing, an abnormal termination will occur. See "Authorized Program Facility" in Catalog Administration Guide.

This parameter only has effect when the entry's master password is not null.

entrypoint
specifies the name of the user's-security-verification routine.

string
specifies information to be passed on to the USVR when it receives control to verify authorization.

Abbreviation: AUTH

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 will be prompted with the name of the entry.

This parameter only has effect when the cluster's or component's master password is not null.

CONTROLPW(password**)**
specifies a control-level password for the page space. Because the page space is a system data set, it cannot be opened or used by a user's program.

If a read or update password is the only password specified for the page space, it (the highest-level password) propagates upward and becomes the password for all higher unspecified levels.

The passwords are cataloged in both the page space entry and its data component's entry. The system automatically prevents users from accessing page spaces and their data components.

Abbreviation: CTLPW

FILE(ddname**)**
specifies the name of the DD statement that identifies the device and volume to be allocated to the page space.

If the page space is unique, and 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.

MASTERPW(password**)**
specifies a master password for the page space. The master password allows all access method services operations against the page space entry. The AUTHORIZATION, CODE, and ATTEMPTS parameters have no effect unless the entry has a master password associated with it. If MASTERPW is not specified, the highest level password specified becomes the password for all higher levels. Because the page space is a system data set, it cannot be opened or used by a user's program.

The passwords are cataloged in both the page space entry and its data component's entry. The system automatically prevents users from accessing page spaces and their data components.

Abbreviation: MRPW

MODEL(entryname[/password]
[catname[/password]])
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 may 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, specify only the entry type (page space) of the model to be used and the name of the entry to be defined.

entryname
specifies the name of the page space entry to be used as a model.

password
specifies a password. If the entry to be used as a model is password-protected and is cataloged in a password-protected catalog, a password is required.

If the protection attributes are to be copied, substitute the master password of either the entry being used as a model (following entryname) or the catalog in which the entry being used as a model is defined (following catname).

If you specify both passwords, the catalog's password is used. If protection attributes are not to be copied, any password can be used. RACF: The alter RACF authority to the model or catalog is required.

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 for either of these cases:

- You specify the catalog's password instead of the model entry's password.
- 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.

READPW(password)
specifies a read-level password for the page space. Because the page space is a system data set, it cannot be opened or used by a user's program.

The passwords are cataloged in both the page space entry and its data component's entry. The system automatically prevents users from accessing page spaces and their data components.

Abbreviation: RDPW

SWAP|NOSWAP
specifies whether page space will be 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.

TO(date)
specifies the date, in the form yyddd, where yy is the year and ddd is the number (001 through 365) of the day through which the cluster being defined is to be kept.

FOR(days)
specifies the number of days for which the page space is to be kept. The maximum number that can be specified is 9999. If the number specified is 0 through 1830, the page space is retained for the number of days specified; if the number is between 1831 and 9999, the page space is retained through the year 1999.

days
can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

UNIQUE|SUBALLOCATION
specifies whether the page space is allocated an amount of space from the volume's available space (UNIQUE) or from a previously defined VSAM data space (SUBALLOCATION).

UNIQUE
specifies that the page space is to be allocated a VSAM data space of its own. VSAM generates a name for the data space and builds a format-1 DSCB in the volume's VTOC to describe it.

Auxiliary storage management recommends that UNIQUE be used for page space data sets. This will cause VSAM to allocate a single nonshared data space for the page data set. Space from another data space will not be used, eliminating one of the possibilities for multiple extent data sets that will cause auxiliary storage management performance degradation.

Abbreviation: UNQ

SUBALLOCATION
specifies that the name of the data space, not of the page space, is to appear in the VTOC. A data space must have been defined on the volume on which the page space is defined.

Abbreviation: SUBAL

UPDATEPW(password)
specifies an update-level password for the page space. Because the page space is a system data set, it cannot be opened or used by a user's program.

If a read password is the only password specified for the page space (that is, it is the highest-level password), it propagates upward and becomes the password for all higher levels. If you specify a higher-level password and do not specify an update password, the update password is null.

The passwords are cataloged in both the page space entry and its data component's entry. The system automatically prevents users from accessing page spaces and their data components.

Abbreviation: UPDPW

DEFINE PAGESPACE 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) -
          CONTROLPW(PASSWD1) -
          UPDATEPW(PASSWD2) -
          READPW(PASSWD3))
/*
```

Job control language statement:

- VOLUME DD, which describes the volume on which the data space is to be defined.

Because no catalog is explicitly specified with a CATALOG parameter or with JOBCAT or STEPCAT DD statements, VSAM assumes that the data space and page space are to be defined in the master catalog.

The DEFINE PAGESPACE command defines a page space. This page space will have the UNIQUE attribute and hence a new data space will be created for its exclusive use. Its parameters are:

- NAME, which specifies the name of the page space, SYS1.PAGE2.
- CYLINDERS, which specifies that the page space is to occupy 10 cylinders. The page spaces are never extended.
- VOLUMES, which specifies that the page space is to reside on volume VSER05.
- CONTROLPW, which specifies the control password for the page space is PASSWD1.
- UPDATEPW, which specifies the update password for the page space is PASSWD2.
- READPW, which specifies the read password for the page space is PASSWD3.

The page space defaults to UNIQUE and NOSWAP.

Define a SWAP Page Space in a Catalog: Example 2

```
//DEFPGSP2 JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     DEFINE PAGESPACE -
          (NAME(SYS1.PAGE1) -
          CYLINDERS(10) -
          VOLUMES(VSER05) -
          SWAP -
          CONTROLPW(PASSWD1) -
          UPDATEPW(PASSWD2) -
          READPW(PASSWD3))
/*
```

Because no catalog is explicitly specified with a CATALOG parameter or with JOBCAT or STEPCAT DD statements, VSAM assumes that the data space and page space are to be defined in the master catalog.

The DEFINE PAGESPACE command defines a page space. Its parameters are:

- NAME, which specifies the name for the page space: SYS1.PAGE1. Because the page space is unique, the page space's name is put into the DSCB (in the volume's VTOC) that describes the space allocated to the page space.
- CYLINDERS, which specifies that the page space occupies 10 cylinders and cannot be extended.
- VOLUMES, which 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, which specifies the page space will be used to store local system queue area (LSQA) pages.
- CONTROLPW, which specifies the control password for the page space is PASSWD1.
- UPDATEPW, which specifies the update password for the page space is PASSWD2.
- READPW, which specifies the read password for the page space is PASSWD3.

The page space defaults to UNIQUE.

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 format of this command is:

```
DEFINE   PATH
           (NAME(entryname)
           PATHENTRY(entryname[/password])
           [ATTEMPTS(number|2)]
           [AUTHORIZATION(entrypoint[  string])]
           [CODE(code)]
           [CONTROLPW(password)]
           [FILE(ddname)]
           [MASTERPW(password)]
           [MODEL(entryname[/password]
             [  catname[/password]])]
           [OWNER(ownerid)]
           [READPW(password)]
           [TO(date)|FOR(days)]
           [UPDATE|NOUPDATE]
           [UPDATEPW(password)])

         [CATALOG(catname[/password])]
```

DEFINE can be abbreviated: DEF

DEFINE PATH PARAMETERS

Required Parameters

PATH
specifies that a path is to be defined.

NAME(entryname)
specifies the path's name.

PATHENTRY(entryname[/password])
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.

password
If the cluster or alternate index entry is password-protected, you must supply the entry's master password. When you identify the catalog with the CATALOG parameter, you can supply the catalog's master password instead of the entry's password. RACF: The alter RACF authority to the cluster, alternate index entry or catalog is required.

Abbreviation: PENT

Optional Parameters

ATTEMPTS(number|**2)**
specifies the maximum number of times the operator can try to enter a correct password in response to a prompting message.

This parameter only has effect when the path's master password is not null. A prompting message is issued only when the user has not already supplied the appropriate password.

number
is an integer from 0 to 7 and can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

Note to TSO users: At a TSO terminal, the logon password is checked first, before the user is prompted to supply a password for the path. Checking the logon password counts as one attempt to obtain a password. If ATTEMPTS is not specified, the user has one attempt to supply the path's password, because the default is 2.

Abbreviation: ATT

AUTHORIZATION(entrypoint[string])
specifies that a user-security-verification routine (USVR) is available for additional security verification. When a protected path is accessed and the user supplies a correct password other than the cluster's master password, the USVR receives control. For details on the USVR, see Catalog Administration Guide.

If a USVR is loaded from an unauthorized library during access method services processing, an abnormal termination will occur. See "Authorized Program Facility" in Catalog Administration Guide.

This parameter only has effect when the path's master password is not null.

entrypoint
specifies the name of the USVR.

string
specifies information to be passed on to the USVR when it receives control to verify authorization.

Abbreviation: AUTH

CATALOG(catname[/password])
identifies the catalog that contains the entry of the cluster or alternate index named in the PATHENTRY parameter. For information about the order in which a catalog is selected if the catalog's name is not specified, see "Order of Catalog Selection for DEFINE" on page 16.

If the cluster's or alternate index's entry is password-protected and its catalog is also password-protected, you must specify the master password for either the entry or the catalog.

catname
specifies the catalog's name.

password
specifies the catalog's master password. RACF: The alter RACF authority to the catalog is required.

If the catalog's volume is physically mounted, it is dynamically allocated. The volume must be mounted as permanently resident or reserved.

Abbreviation: CAT

CODE(code**)**
specifies a code name for the path.

If an attempt is made to access a password-protected path without first supplying an appropriate password, a prompting message is issued to the operator's console. The prompting message includes the code name, which identifies the path without revealing its entryname.

This parameter only has effect when the path's master password is not null. When code is not specified, the prompting message identifies the path with its entryname.

CONTROLPW(password**)**
specifies a control password for the path. Control interval processing is not permitted through a path; the control password will permit read and write operations against the base cluster.

If a read or update password is the only password specified for the object, it (the highest-level password) propagates upward and becomes the password for all higher unspecified levels.

Abbreviation: CTLPW

FILE(ddname**)**
specifies the name of a DD statement that identifies the recovery volume of the alternate index or cluster named with the PATHENTRY parameter. FILE is only used when the path is defined in a recoverable catalog.

The recovery volume is the first volume of the base cluster's index component when PATHENTRY names a key-sequenced cluster or an alternate index over a key-sequenced cluster.

The recovery volume is the first volume of the base cluster's data component when PATHENTRY names an entry-sequenced cluster or an alternate index over an entry-sequenced cluster.

When FILE is not specified, an attempt is made to dynamically allocate the cluster's recovery volume. The volume must be mounted as permanently resident or reserved.

MASTERPW(password**)**
specifies a master password for the path. The master password allows all operations against the path.

If the master password is not specified, the path's highest-level password propagates upward and becomes the password for all higher levels, including the master password.

If all passwords are null, ATTEMPTS, AUTHORIZATION, and CODE have no effect until the master password is specified.

Abbreviation: MRPW

MODEL(entryname**[/**password**]**
[catname**[/**password**]])**
identifies 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.

entryname
names the entry to be used as a model. entryname must name a path entry.

password
specifies a password. If the model entry is password-protected and it is cataloged in a password-protected catalog, you must supply the read (or higher level) password of either the model entry or its catalog. If you specify both passwords, the catalog's password is used. RACF: The read or higher RACF authority to the model or catalog is required.

If you are not specifying new protection attributes for the path (that is, the model's passwords and protection attributes are being copied), you must supply the master password of either the model entry or its catalog. RACF: The alter RACF authority to the model or catalog is required.

catname
names the model entry's catalog. You must identify the catalog that contains the model entry for either of these cases:

- If you specify the catalog's password instead of the model entry's password.
- 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. For information about the order in which a catalog is selected when the catalog's name is not specified, see "Order of Catalog Selection for DEFINE" on page 16.

OWNER(ownerid**)**
specifies the identification of the path's owner.

Note to TSO users: If the owner is not identified with the OWNER parameter, the TSO user's userid becomes the ownerid.

READPW(password**)**
specifies a read password for the path. The read password permits read operations against the base cluster's data records.

Abbreviation: RDPW

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.

TO(date**)**
specifies the date, in the form yyddd, where yy is the year and ddd is the Julian date (001, for January 1, through 365, for December 31), through which the path is to be kept before it is allowed to be deleted.

FOR(days**)**
specifies the number of days for which the entry is to be kept before it is allowed to be deleted. days can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

The maximum number that can be specified for days is 9999. If the number specified is 0 through 1830, the retention period is the number of days specified. If

the number specified is between 1831 and 9999, the retention period is through the year 1999.

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.

Note: NOUPDATE will be overridden by opening the path, allowing sharing of a control block structure that permits UPDATE.

Abbreviation: NUPD

UPDATEPW(password)
specifies the update password for the path. The update password permits read and write operations against the base cluster's data records.

If a read password is the only password specified for the object (that is, it is the highest-level password), it propagates upward and becomes the password for all higher levels. If you specify a higher-level password and do not specify an update password, the update password is null.

Abbreviation: UPDPW

DEFINE PATH EXAMPLE

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, USERRCAT.

```
//DEFPATH  JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE PATH -
               (NAME(EXAMPLE.PATH) -
               PATHENTRY(EXAMPLE.AIX/AIXMRPW) -
               READPW(PATHRDPW) ) -
          CATALOG(USERRCAT/USERUPPW)
/*
```

The DEFINE PATH command builds a path entry to define the path EXAMPLE.PATH. A copy of the path entry is placed in the catalog recovery area, because the path entry is being defined into a recoverable catalog. An attempt is made to dynamically allocate the catalog recovery area on volume VSER01. The command's parameters are:

- NAME, which specifies that the path's name is EXAMPLE.PATH.
- PATHENTRY, which identifies the alternate index, EXAMPLE.AIX, that the path provides access to.
- READPW, which specifies the path's read password, PATHRDPW.
- CATALOG, which in this example supplies the catalog's update password, USERUPPW.

DEFINE SPACE

The DEFINE SPACE command defines a VSAM data space in a VSAM catalog. The format of this command is:

```
DEFINE   SPACE
           ({CANDIDATE|
             CYLINDERS(primary[ secondary])|
             RECORDS(primary[ secondary])
               RECORDSIZE(average maximum)|
             TRACKS(primary[ secondary])}
           VOLUMES(volser[ volser...])
           [FILE(ddname)])

         [CATALOG(catname[/password])]
```

DEFINE can be abbreviated: DEF

DEFINE SPACE PARAMETERS

Required Parameters

SPACE
specifies that a data space is to be defined. You can also use the DEFINE SPACE command to reserve a volume for VSAM's future use.

Abbreviation: SPC

CANDIDATE|
CYLINDERS(primary**[** secondary**])|**
RECORDS(primary**[** secondary**]) RECORDSIZE(**average maximum**)|**
TRACKS(primary**[** secondary**])**
specifies the volume(s) to be reserved for VSAM's future use or specifies the amount of space to be allocated in terms of cylinders, number of records, or tracks.

You must specify only one of the following parameters: CANDIDATE, CYLINDERS, TRACKS, or RECORDS. When you specify RECORDS, you must also specify RECORDSIZE. When you specify CYLINDERS or TRACKS, you cannot specify RECORDSIZE.

CANDIDATE
specifies that the volumes listed in the VOLUMES parameter are reserved for future use by VSAM. An ownership indicator is set in a format-4 DSCB in the volume's table of contents (VTOC). The volumes are reserved for use by the catalog, but no space is allocated (other than the primary CRA allocation if the catalog is recoverable). Each volume so reserved is called a candidate volume. If CANDIDATE is not specified, you must specify one of the following parameters: CYLINDERS, TRACKS, or RECORDS.

If a VSAM data space already exists on the volume, you cannot specify CANDIDATE. (You can, however, use the ALTER command to identify the volume as a candidate volume for a VSAM object other than the volume's catalog.)

Abbreviation: CAN

CYLINDERS(primary**[** secondary**])|**
RECORDS(primary**[** secondary**])**
RECORDSIZE(average maximum**)|**
TRACKS(primary**[** secondary**])**
specifies the amount of space to be allocated in terms of tracks, cylinders, or number of records.

If RECORDS is specified, the space required is calculated in terms of the number of records, but the space is allocated by tracks. However, if you specify TRACKS or RECORDS and the minimum number of tracks exceeds a cylinder, space is allocated in terms of cylinders.

primary
specifies the initial amount of space (primary extent) to be allocated to the data space. primary can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

secondary
specifies the amount of space that is to be allocated to the data space each time it is extended. Once the primary extent is filled, the data space can expand to include a maximum of 15 secondary extents if you have specified a secondary extent amount. secondary can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

The secondary space allocation is rounded upward to the nearest whole cylinder regardless of your specification in terms of TRACKS or RECORDS. Consequently, the total amount of space allocated to the data space might exceed the predicted limit.

When the data space is the first data space on a volume that belongs to a recoverable catalog, the primary amount is increased by one cylinder.

Abbreviations: CYL, REC, and TRK

RECORDSIZE(average maximum**)**
specifies the average and maximum lengths, in bytes, of the records.

average and maximum can be any integer value between 1 and 32 761, expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. When you specify the RECORDS parameter, you must also specify RECORDSIZE; otherwise, you cannot specify RECORDSIZE.

Abbreviation: RECSZ

VOLUMES(volser**[** volser...**])**
specifies the volumes on which data spaces are to be defined or volumes to be reserved as candidate volumes for VSAM's future use. If two or more volumes are specified and an amount of space is specified (that is, TRACKS, CYLINDERS, or RECORDS and RECORDSIZE is specified) the amount specified for the primary allocation is allocated on each volume. The specified volumes must all be of the same device type. When the volume already has a data space on it, subsequent data spaces can be defined and must be cataloged in the catalog that owns the volume. A data space can be defined on a mass storage volume.

Abbreviation: VOL

Optional Parameters

CATALOG(catname[/password]**)**
identifies the catalog in which the data space is to be defined. The CATALOG parameter supplies the name and password, when required, of the catalog that contains the volume entry that describes the volume on which the data space is to be allocated.

If the DEFINE SPACE command is used to allocate the volume's first VSAM data space or to identify the volume as a candidate volume, the CATALOG parameter identifies the catalog that is to contain the volume's catalog entry and is to own the volume. (For more information about VSAM volume ownership, see Catalog Administration Guide.)

If the catalog's volume is physically mounted, it is dynamically allocated. For information about the order in which catalogs are selected, see "Order of Catalog Selection for DEFINE" on page 16.

catname
specifies the name of the catalog.

password
specifies a password. If the catalog is password protected, you must supply the update- or higher-level password. RACF: The update or higher RACF authority to the catalog is required.

Abbreviation: CAT

FILE(ddname**)**
specifies the name of the DD statement that identifies the device type and volumes to be used for space allocation. You cannot use concatenated DD statements to describe more than one volume.

If FILE is not specified and the volume is physically mounted, the volume(s) identified with the VOLUMES parameter is dynamically allocated. Each dynamically allocated volume must be mounted as permanently resident or reserved.

DEFINE SPACE EXAMPLE

Define a Data Space: Example 1

In this example, a data space is defined. The data space is to be used to allocate space for VSAM clusters that are subsequently defined.

```
//DEFSPC1  JOB     ...
//JOBCAT   DD      DSN=D27UCAT1,DISP=SHR
//STEP1    EXEC    PGM=IDCAMS
//SYSPRINT DD      SYSOUT=A
//SYSIN    DD      *
     DEFINE SPACE -
          (CYLINDERS(30 10) -
          VOLUMES(VSER05)) -
          CATALOG(D27UCAT1/USERMRPW)
/*
```

Access method services defines the data space and allocates its space on volume VSER05. The data space is cataloged in D27UCAT1, because a JOBCAT DD statement is specified. All future data spaces and clusters on volume VSER05 must also be cataloged in the same catalog. Non-VSAM data sets on the volume reside in areas that are not allocated to a VSAM data space.

The DEFINE SPACE command defines a VSAM data space. Its parameters are:

- CYLINDERS, which specifies that 30 cylinders are to be allocated for the data space. When the data space is extended, it is to be extended in increments of 10 cylinders.
- VOLUMES, which specifies the volume serial number of the volume on which the data space is to be defined: VSER05.
- CATALOG, which in this example supplies the user catalog's name, D27UCAT1, and its master password, USERMRPW.

DEFINE USERCATALOG|MASTERCATALOG

DEFINE USERCATALOG|MASTERCATALOG

The DEFINE USERCATALOG|MASTERCATALOG command defines a VSAM user or master catalog. When you use this command, you can specify attributes for the catalog as a whole and for the components of the catalog.

The format of this command is:

```
DEFINE   USERCATALOG|MASTERCATALOG
           (NAME(entryname)
           {CYLINDERS(primary[ secondary])|
             RECORDS(primary[ secondary])|
             TRACKS(primary[ secondary])}
           VOLUME(volser)
           [ATTEMPTS(number|2)]
           [AUTHORIZATION(entrypoint[ string])]
           [BUFFERSPACE(size|3072)]
           [CODE(code)]
           [CONTROLPW(password)]
           [DESTAGEWAIT|NODESTAGEWAIT]
           [FILE(ddname)]
           [MASTERPW(password)]
           [MODEL(entryname[/password][ catname[/password]])]
           [OWNER(ownerid)]
           [READPW(password)]
           [RECOVERABLE|NOTRECOVERABLE]
           [TO(date)|FOR(days)]
           [UPDATEPW(password)]
           [WRITECHECK|NOWRITECHECK])

         [DATA
           ([BUFFERSPACE(size)]
           [CYLINDERS(primary[ secondary])|
             RECORDS(primary[ secondary])|
             TRACKS(primary[ secondary])]
           [DESTAGEWAIT|NODESTAGEWAIT]
           [RECORDSIZE(average maximum)]
           [RECOVERABLE|NOTRECOVERABLE]
           [WRITECHECK|NOWRITECHECK])]

         [INDEX
           ([CYLINDERS(primary)|
             RECORDS(primary)|
             TRACKS(primary)]
           [DESTAGEWAIT|NODESTAGEWAIT]
           [WRITECHECK|NOWRITECHECK])]

         [CATALOG(mastercatname[/password])]
```

DEFINE can be abbreviated: DEF

DEFINE USERCATALOG|MASTERCATALOG PARAMETERS

Required Parameters

USERCATALOG|MASTERCATALOG
specifies that a catalog is to be defined.

USERCATALOG
specifies that a user catalog is to be defined. USERCATALOG is followed by the parameters specified for the catalog as a whole. RACF: The update or higher RACF authority to the master catalog is required.

Abbreviation: UCAT

MASTERCATALOG
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. For a description of this procedure, see Catalog Administration Guide.

Abbreviation: MCAT

NAME(entryname**)**
specifies the name of the catalog being defined.

CYLINDERS(primary[secondary]**)|**
RECORDS(primary[secondary]**)|**
TRACKS(primary[secondary]**)**
specifies the amount of space to be allocated in terms of cylinders, tracks, or number of records. 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.

If you specify less than one cylinder of space for a recoverable catalog, your DEFINE command will fail.

For a description of the difference in space allocation depending on which parameters you specify, see Catalog Administration Guide. This guide also provides information about estimating the amount of space to be specified for a catalog.

primary[secondary]
specify the size of the primary and secondary extents to be allocated. Once the primary extent is filled, the space can expand to include a maximum of 13 additional secondary extents if you have specified a secondary allocation amount.

primary and secondary can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. Secondary allocation should be specified in case the catalog has to be extended.

Abbreviations: CYL, REC, and TRK

VOLUME(volser**)**
specifies the volume that is to contain the catalog. The volume cannot be currently owned by any other VSAM catalog.

The VOLUME parameter interacts with other DEFINE CATALOG parameters. You should 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.

Abbreviation: VOL

Optional Parameters

ATTEMPTS(number|2)
specifies the maximum number of times the operator can try to enter a correct password in response to a prompting message.

number
is an integer from 0 to 7 and can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form. If 0 is specified, the operator is not prompted and is not allowed to enter a password from the console.

Note to TSO users: At a TSO terminal the logon password is checked first, before the user is prompted to supply a password for the catalog. Checking the logon password counts as one attempt to obtain a password. If ATTEMPTS is not specified, the user has one attempt to supply the catalog's password, because the default is 2.

Abbreviation: ATT

AUTHORIZATION(entrypoint[string])
specifies that a user-security-verification routine (USVR) is available for additional security verification. When a protected catalog is accessed and the user supplies a correct password other than the catalog's master password, the USVR receives control. For information on the user-security-verification routine, see Catalog Administration Guide.

entrypoint
specifies the name of the USVR.

string
specifies information to be passed on to the USVR when it receives control to verify authorization.

Abbreviation: AUTH

BUFFERSPACE(size|3072)
specifies the minimum space, in bytes, to be provided for buffers when using the catalog (one not residing on a 3380). Decimal values you can specify are 3072, 4096, 5120, 6144, 7168, and 8192.

size
is the amount of space, in bytes, to be provided for buffers. Size can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form, but must not exceed 16776704.

Abbreviation: BUFSP

CATALOG(mastercatname[/password])
specifies the name and password of the master catalog. If the master catalog is password protected, its update- or higher-level password must be provided in this parameter or in response to prompting.

mastercatname
is the name of the master catalog that is required when a user catalog is being defined.

password
specifies the update- or higher-level password of the master catalog that is required when a user catalog is being defined, if the master catalog is password protected.

Abbreviation: CAT

CODE(code**)**
specifies a code name for the catalog being defined. If an attempt is made to access a password-protected catalog without a password, the code name is used in a prompting message; the code enables the operator to be prompted for the password without disclosing the name of the catalog. If CODE is not specified, the operator will be prompted with the name of the catalog.

CONTROLPW(password**)**
specifies a control password for the catalog being defined. The control password permits the same operations as the update password.

If a read or update password is the only password specified for the catalog, it (the highest-level password) propagates upward and becomes the password for all higher levels. When the master password is specified but the control password is not specified, the control password is null.

Abbreviation: CTLPW

DESTAGEWAIT|NODESTAGEWAIT
specifies whether a user catalog that is stored on a mass storage volume is to be destaged synchronously or asynchronously with respect to the program that closes it.

If the user catalog is not stored on a mass storage volume, the attribute is ineffective until the direct access storage volume, the catalog is stored on is converted to a mass storage volume (using the CONVERTV command described in OS/VS Mass Storage System (MSS) Services: Reference Information).

If one of these parameters is specified for the data component of a catalog and the other parameter is specified for the index component, the parameter specified for the index component applies to both components, because the sequence set of the index of a catalog is embedded in the data.

DESTAGEWAIT
indicates that destaging is to be completed before control is returned from VSAM to the program that issues the CLOSE macro. VSAM can thus notify the program whether destaging was successful.

Abbreviation: DSTGW

NODESTAGEWAIT
indicates that notification of unsuccessful destaging is to be made only by a message to the operator.

Abbreviation: NDSTGW

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.

MASTERPW(password**)**
specifies a master password for the catalog being defined. A catalog must have a master password in order for VSAM data sets cataloged in it to be password-protected. The AUTHORIZATION, CODE, and ATTEMPTS parameters have no effect unless the catalog has a master password associated with it. The master password allows all operations; it is required to open the catalog as a data set.

If a master password is not specified for the catalog, the catalog's highest-level password propagates upward and becomes the password for all higher levels, including the master password. The master password is null when no other passwords are specified.

Abbreviation: MRPW

MODEL(entryname[/password][catname[/password]]**)**
specifies that an existing master or user catalog is to be used as a model for the user catalog being defined.

It is possible to use an already defined master or user catalog as a model for another user catalog.

When one entry is used as a model for another, its attributes are copied as the new entry is defined. You may 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.

entryname
specifies the name of the master or user catalog to be used as a model.

password
specifies a password. If the catalog to be used as a model is password-protected, a password is required.

If you specify both passwords (that is, the password following entryname and the password following catname), the password following catname is used for authorization.

If the protection attributes are to be copied, enter the master password of the catalog being used as a model. RACF: The alter RACF authority to the catalog is required. If passwords are not to be copied, any password except the master can be used.

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.

READPW(password**)**
specifies a read password for the catalog being defined. The read password permits the user to list the catalog's entries. (Passwords and protection attributes are listed only when the master-level password is supplied.)

Abbreviation: RDPW

RECOVERABLE|NOTRECOVERABLE
specifies whether a catalog recovery space is to be created on each volume owned by the catalog.

RECOVERABLE
specifies that a catalog recovery space is to be created.

On the catalog's volume, the catalog recovery space is allocated from the catalog's data space. On

subsequent volumes owned by the catalog, the catalog recovery space is allocated from the first data space defined on the volume.

Primary allocation for a CRA is one cylinder. Once a primary extent is filled, the CRA can expand to include a maximum of 15 additional extents. Allocation for each secondary extent is one cylinder.

The number of records a CRA can contain varies with the device type (for example, for a 3330 with 20 control intervals per track, each extent will accommodate approximately 370 records).

Abbreviation: RVBL

NOTRECOVERABLE
specifies that no catalog recovery space is to be created.

Abbreviation: NRVBL

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.

TO(*date*)
specifies the date, in the form yyddd, where yy is the year and ddd is the number (001 through 365) of the day through which the catalog being defined is to be kept.

FOR(*days*)
specifies the number of days for which the catalog being defined is to be kept.

The maximum number that can be specified is 9999. If the number specified is 0 through 1830, the catalog is retained for the number of days specified; if the number is between 1831 and 9999, the catalog is retained through the year 1999.

days
can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form.

UPDATEPW(*password*)
specifies an update password for the catalog being defined. The update password permits entries to be added to the catalog being defined. The catalog must be update password-protected in order to prevent unauthorized users from cataloging data sets.

If a read password is the only password specified for the catalog (that is, it is the highest-level password), it propagates upward and becomes the password for all higher levels. If you specify a higher-level password and do not specify an update or read password, the update password is null.

When you anticipate using the catalog to contain many non-VSAM entries, you should consider allowing the catalog's update password to be null (that is, to specify a master password but not an update password). A null update password allows you to alter and delete non-VSAM entries without having to supply the catalog's password.

Abbreviation: UPDPW

WRITECHECK|NOWRITECHECK
specifies whether the catalog is to be checked by a direct access device operation called write check when a record is written to the device.

WRITECHECK
specifies that a record is to be written and then read, without data transfer.

Abbreviation: WCK

NOWRITECHECK
specifies that the catalog is not to be checked by a write check when a record is written to the device.

Abbreviation: NWCK

DATA AND INDEX COMPONENTS OF USERCATALOG

Attributes can be specified separately for the catalog's data and index components. The DATA and INDEX parameters are listed 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

Defining a VSAM User Catalog, Having Calculated Its Space Requirements: Example 1

In this example, a user catalog is defined. The user catalog's data space is not available to contain other VSAM objects. The catalog is to be large enough to contain information about:

- 100 key-sequenced clusters (that is, indexed VSAM data sets)
- 10 entry-sequenced clusters (that is, sequential nonindexed VSAM data sets)
- 5 key-sequenced clusters with alternate indexes to be upgraded
- 5 entry-sequenced clusters with alternate indexes to be upgraded
- 10 alternate indexes
- 10 paths
- 200 aliases
- 10 generation data sets
- 1000 non-VSAM data sets.
- 5 volumes (the number to be controlled by the catalog). Each volume is a 3330 volume and contains 20 data spaces.

To determine how much space, in records and tracks, is required for the primary allocation of a catalog still to be defined in your system, see Catalog Administration Guide. The number of records for the primary allocation of the catalog in this example is shown in the completed worksheet in Figure 3 on page 148.

When you define the catalog's space parameters, use the following format:

```
DEFINE USERCATALOG -
     (TRACKS(prim secn) ... )
```

where:

- prim = the minimum amount of space for the catalog data space's primary allocation. In this example, all the data space is assigned to the catalog itself. The calculated value of prim should be rounded upward to the next whole number of tracks that is an even multiple of the catalog allocation unit for the device type being used (For the size of an allocation unit, see Catalog Administration Guide.)

  Because the catalog resides on a 3330-1 volume, the value of prim is:

  prim = .09N + 3 = 152.4 = = 153 tracks

Variable Quantities	Formulas	Estimates
Basic requirement = 10 records		10
A = number of key-sequenced clusters	Ax3	300
A^1 = number of key-sequenced clusters with alternate indexes to be upgraded	A^1x4	20
B = number of entry-sequenced clusters	Bx2	20
B^1 = number of entry-sequenced clusters with alternate indexes to be upgraded	B^1x3	15
C = number of relative-record clusters	Cx2	0
D = number of alternate indexes	Dx3	30
E = number of path entries	E	10
F = number of nonVSAM data set entries	F	1000
G = number of generation data group entries	G	10
H = number of alias entries	H	200
I = number of page spaces	Ix2	0
J = number of volumes, depending on device type, owned by the catalog:		
J^1 = number of 2305 volumes	J^1x2	0
J^2 = number of 2314/2319 volumes	J^2x3	20
J^3 = number of 3330 and 3340/3344 volumes	J^3x4	0
J^4 = number of 3330 Model 11 and 3350 volumes	J^4x6	
K = for each key-sequenced cluster and alternate index (KSDS) with space on more than two volumes, add "1" for each additional group of one to five volumes:		
K^1 = number of KSDSs with 3 to 7 volumes	K^1	0
K^2 = number of KSDSs with 8 to 12 volumes	K^2x2	0
K^3 = number of KSDSs with 13 to 17 volumes	K^3x3	0
L = for each entry-sequenced cluster and relative-record cluster (ESDS) with space on more than five volumes, add "1" for each additional group of one to eight volumes:		
L^1 = number of ESDSs with 6 to 13 volumes	L^1	0
L^2 = number of ESDSs with 14 to 21 volumes	L^2 x2	0
M = for each group of four data spaces on a volume, add "1"	M	25
N = number of entry records required for the catalog's data component (total of above)	N	1660

Figure 3. Completed Worksheet for Determining a Catalog's Space Requirements

- secn = the minimum amount of space for each secondary extent of the catalog's data space.

  The number of tracks required for the catalog's primary extent is at least 153. This figure is an even multiple of an allocation unit for a 3330 device (3); therefore, no further rounding is necessary. If VSAM extends the catalog, it should extend it in increments of 18 tracks.

```
//DEFCAT2  JOB     ...
//STEP1    EXEC    PGM=IDCAMS
//SYSPRINT DD      SYSOUT=A
//SYSIN    DD      *
     DEFINE USERCATALOG( -
          NAME(USERCAT3) -
          VOLUME(VSER04) -
          TRACKS(153 18) )
/*
```

The DEFINE USERCATALOG command defines a user catalog. Its parameters are:

- NAME, which specifies the name of the catalog, USERCAT3.
- VOLUME, which specifies that the catalog is to reside on the volume whose serial number is VSER04.
- TRACKS, which specifies that the catalog's data space is to be 153 tracks. When the user catalog is extended, VSAM extends it in increments of 18 tracks.
- Values and attributes that apply by default to the catalog are BUFFERSPACE = 3072 bytes, ATTEMPTS = 2, NOWRITECHECK, NODESTAGEWAIT, and NOTRECOVERABLE.

Define a User Catalog Leaving Available Suballocatable Space: Example 2

In this example, a user catalog is defined. The catalog will actually occupy only a portion of the space defined, leaving the remainder of the space available for the suballocation of other VSAM data sets.

```
//DEFCAT3  JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE USERCATALOG( -
                NAME(D27UCAT1) -
                MASTERPW(USERMRPW) -
                UPDATEPW(USERUPPW) -
                FOR(365) -
                CYLINDERS(10 1)) -
                VOLUME(VSER02) ) -
           DATA( -
                CYLINDERS(3 1) -
           INDEX( -
                CYLINDERS(1) ) -
           CATALOG(AMAST1/MASTUPW1)
/*
```

The DEFINE USERCATALOG command defines a user catalog, D27UCAT1. Its parameters are:

- NAME, which names the user catalog, D27UCAT1.
- MASTERPW and UPDATEPW, which specify master and update passwords for the catalog: USERMRPW and USERUPPW.
- FOR, which specifies that the user catalog is to be retained for 365 days.
- CYLINDERS, which specifies that 10 cylinders are to be allocated for the user catalog's data space. When the user catalog's data space is extended, it is to be extended in increments of 1 cylinder.
- VOLUME, which specifies that the catalog is to reside on volume VSER02. Volume VSER02 will be dynamically allocated.
- DATA and INDEX, which specify that VSAM is to allocate 4 cylinders for the catalog itself. VSAM determines the proportion of space that is allocated to the catalog's data and index components. The remainder of the catalog's data space (6 cylinders) is available to contain VSAM objects that reside in a nonunique data space. If the catalog's data component is extended, it is to be extended in increments of 1 cylinder. The catalog's index component is never extended.
- CATALOG, which specifies that the user catalog connector entry is to be defined in the master catalog, AMAST1.

CATALOG provides the catalog's update password, MASTUPW1, so that the operator is not prompted to provide it.

- Values and attributes that apply by default to the catalog are BUFFERSPACE = 3072 bytes, ATTEMPTS = 2, NOTRECOVERABLE, NOWRITECHECK, and NODESTAGEWAIT.

Define a User Catalog Using the MODEL Parameter: Example 3

In this example, the user catalog defined previously, D27UCAT1, is used as a model for the user catalog being defined, D27UCAT2.

```
//DEFCAT4  JOB     ...
//STEP1    EXEC    PGM=IDCAMS
//STEPCAT  DD      DSNAME=D27UCAT1,DISP=SHR
//SYSPRINT DD      SYSOUT=A
//SYSIN    DD      *
     DEFINE USERCATALOG( -
                NAME(D27UCAT2) -
                MASTERPW(USERMRPW) -
                UPDATEPW(USERUPPW) -
                CYLINDERS(15 5) -
                VOLUME(VSER03) -
                MODEL(D27UCAT1/MRPWD27 -
                  D27UCAT1/MRPWD27)  ) -
           CATALOG(AMAST1/MASTUPW1)
/*
```

Job control language statement:

- STEPCAT DD, which makes a catalog available for this job step: D27UCAT1.

The DEFINE USERCATALOG command defines catalog D27UCAT2. Its parameters are:

- NAME, which names the catalog, D27UCAT2.
- MASTERPW and UPDATEPW, which specify master and update passwords for the catalog: USERMRPW and USERUPPW.
- CYLINDERS, which specifies that 15 cylinders are to be allocated for the catalog's data space. The catalog itself will reside within this data space. When the catalog's data space is extended, it is to be extended in increments of 5 cylinders.
- VOLUME, which specifies that the catalog is to reside on volume VSER03. Volume VSER03 will be dynamically allocated.
- MODEL, which identifies D27UCAT1 as the catalog to use as a model for D27UCAT2. The attributes and specifications of D27UCAT1 that are not otherwise specified with the above parameters are used to define the attributes and specifications of D27UCAT2. The master catalog, AMAST1, contains a user catalog connector entry which points to D27UCAT1. This is why D27UCAT1 is specified as MODEL's catname subparameter. Values and attributes that apply to D27UCAT2 as a result of using D27UCAT1 as a model are FOR = 365 days, CYLINDERS = 3 (primary) and 1 (secondary) for the data component and 1 (primary) for the index component, BUFFERSPACE = 3072 bytes, ATTEMPTS = 2, NOWRITECHECK, and NODESTAGEWAIT.
- CATALOG, which specifies that the user catalog connector is to be defined in the AMAST1 catalog. The update password of AMAST1 is MASTUPW1.

Define a VSAM Recoverable User Catalog: Example 4

In this example, a recoverable user catalog is defined.

```
//DEFCAT5  JOB    ...
//STEP1    EXEC   PGM=IDCAMS
//VOL1     DD     VOL=SER=VSER01,UNIT=DISK,DISP=OLD
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
     DEFINE USERCATALOG( -
                NAME(USERRCAT) -
                MASTERPW(USERMRPW) -
                UPDATEPW(USERUPPW) -
                TO(99365) -
                ATTEMPTS(3) -
                CODE(BEQUIET) -
                RECOVERABLE -
                TRACKS(100) -
                FILE(VOL1) -
                VOLUME(VSER01)   ) -
           DATA( -
                RECORDS(500 100)) -
                CATALOG(AMAST1/MASTUPW1)
/*
```

Job control language statement:

- VOL1 DD, which describes the volume on which the catalog is to be defined.

The DEFINE USERCATALOG command defines a user catalog, USERRCAT. Its parameters are:

- NAME, which names the user catalog, USERRCAT.
- MASTERPW and UPDATEPW, which specify master and update passwords for the catalog: USERMRPW and USERUPPW.
- TO, which specifies that the user catalog is to be retained until 31 December 1999.
- ATTEMPTS, which specifies that the operator may be prompted for the correct password up to three times.
- CODE, which specifies that the catalog is identified, in the prompting message to the operator, with the name "BEQUIET".
- RECOVERABLE, which specifies that the catalog's volumes contain catalog recovery areas and the LISTCRA, EXPORTRA, IMPORTRA, and RESETCAT commands can be used to restore entries in the catalog.
- TRACKS, which specifies that 100 tracks are to be allocated for the user catalog's data space. No secondary allocation is specified. If additional VSAM space is required on the volume, it can be explicitly defined using the DEFINE SPACE command.
- FILE, which points to the VOL1 DD statement. The VOL1 DD statement directs the operating system scheduler to mount and allocate the volume on which the user catalog is to reside. In this example, the volume is not dynamically allocated.
- VOLUME, which specifies that the user catalog is to reside on volume VSER01.

- DATA, which specifies that VSAM is to allocate space to accommodate 500 records for the user catalog. VSAM determines the total amount of space that is allocated to the catalog's data and index components. The remainder of the catalog's data space is available to contain the catalog recoverable area (CRA) and other VSAM objects that reside in a nonunique data space. If the catalog's data component is extended, it is extended in increments to accommodate 100 records.

  You may wish to estimate the amount of space which VSAM will actually allocate to the catalog in order to:

  - Ensure that the total amount of space specified via TRACKS is large enough.
  - Determine how much suballocatable space will be available.

  Because the catalog resides on a 3330-1, the total amount of space will be:

  $$aN + b = (0.1116 \times 500) + 4 = 59.8 = 60 \text{ tracks}$$

- CATALOG, which specifies that the user catalog connector is to be defined in the AMAST1 catalog. The update password of AMAST1 is MASTUPW1.
- Values and attributes that apply by default to the catalog are BUFFERSPACE = 3072 bytes, NOWRITECHECK, and NODESTAGEWAIT.

DELETE

The DELETE command deletes catalogs, VSAM data sets, and non-VSAM data sets. The format of this command is:

```
DELETE   (entryname[/password][  entryname[/password] ...])
         [ALIAS|
            ALTERNATEINDEX|
            CLUSTER|
            GENERATIONDATAGROUP|
            NONVSAM|
            PAGESPACE|
            PATH|
            SPACE|
            USERCATALOG
         [ERASE|NOERASE]
         [FILE(ddname)]
         [FORCE|NOFORCE]
         [PURGE|NOPURGE]
         [SCRATCH|NOSCRATCH]

         [CATALOG(catname[/password])]
```

DELETE can be abbreviated: DEL

DELETE PARAMETERS

Required Parameters

(entryname[/password][entryname[/password]...])
names the entries to be deleted. If more than one entry is to be deleted, the list of entrynames must be enclosed in parentheses.

This parameter must be the first parameter following DELETE.

entryname
is the name of the entry to be deleted or the volume serial number of the volume that contains a data space to be deleted. A generic name may be coded in order to delete multiple entries with one entryname. (For example, GENERIC.*.BAKER is a generic name where "*" is any 1- to 8-character simple name.)

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 it from the SYSRES volume.

password
specifies a password for a password-protected entry. Passwords may be specified for each entryname, or the catalog's password may be specified through the CATALOG parameter for the catalog that contains the entries to be deleted. Only certain types of catalog entries can be password protected, as specified below.

To delete:

- Alias, specify the catalog's update- or higher-level password. RACF: The RACF alter authority to the entry to which the alias is related is required.

- Alternate index, specify the entry's master password, or the catalog's master password. RACF: The RACF alter authority to the alternate index or to the catalog is required.
- Cluster, specify the entry's master password, or the catalog's master password. RACF: The RACF alter authority to the cluster or to the catalog is required.
- Data space, specify the catalog's update- or higher-level password. RACF: The RACF alter authority to the catalog is required.
- Generation data group, specify the catalog's update- or higher-level password. RACF: The RACF alter authority to the GDG base or to the catalog is required.
- Non-VSAM entry, specify the catalog's update- or higher-level password. RACF: The RACF alter authority to the non-VSAM data set or to the catalog is required.
- Page space, specify the entry's master password, or the catalog's master password. RACF: The RACF alter authority to the page space or to the catalog is required.
- Path, specify the entry's master password, or the catalog's master password. RACF: The RACF alter authority to the path or to the catalog is required.
- User catalog, specify the entry's master password with the entryname so that access method services does not issue a prompting message to the operator or TSO terminal user. RACF: The RACF alter authority to the user catalog is required.

Optional Parameters

ALIAS|ALTERNATEINDEX|CLUSTER|
GENERATIONDATAGROUP|NONVSAM|PAGESPACE|
PATH|SPACE|USERCATALOG

specifies the type of object or entry to be deleted. SPACE is required to delete a data space, and USERCATALOG is required to delete a catalog.

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 attribute that is still to be upgraded and it is the only such alternate index associated with the base cluster, the base cluster's upgrade set entry is also deleted.

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.

If the cluster is defined in a recoverable catalog, DELETE requires the catalog recovery volume for the cluster to be mounted. If the CRA volume is not available, then any volume with the required CRA volume serial number satifies DELETE's requirements.

Abbreviation: CL

GENERATIONDATAGROUP
specifies that the entry to be deleted is a generation data group entry. To delete a generation data group that is not empty, you must specify the FORCE parameter.

Abbreviation: GDG

NONVSAM
specifies that the entry to be deleted is a non-VSAM data set entry.

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).

RACF commands can be used 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.

Abbreviation: NVSAM

PAGESPACE
specifies that an inactive page space is to be deleted. A page space is identified as "active" during the operator's IPL procedure.

Abbreviation: PGSPC

PATH
specifies that a path entry is to be deleted. No entries associated with the path are deleted.

SPACE
specifies that all empty VSAM data spaces on a volume are to be deleted. This parameter is required when you want to delete the volume's empty data spaces.

To delete a data space that is not empty, you must specify both the SPACE parameter and the FORCE parameter.

If all data spaces on a volume have been deleted and the volume is not a candidate or catalog volume, its volume entry is also deleted.

Note: When a volume entry in the catalog is deleted, VSAM's ownership of the volume is relinquished. When the catalog that owns the volume is recoverable, the data space that contains the volume's recovery area is not deleted unless all other data spaces on the volume have been deleted. CRAs must be deleted in a separate command.

Abbreviation: SPC

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 volume entry), you must specify USERCATALOG. To delete a user catalog that is not empty, you must specify both USERCATALOG and FORCE.

Note: The JOBCAT/STEPCAT is ignored for this command. EXPORT DISCONNECT must be used to delete a user catalog entry from a user catalog. A master catalog for another system is considered to be a user catalog by the processing system.

Abbreviation: UCAT

CATALOG(catname[/password]**)**
specifies the name of the catalog that contains the entries to be deleted. For information about the order in which catalogs are searched, see "Order of Catalog Search for DELETE" on page 17.

Note: This parameter cannot be used to delete a user catalog.

catname
identifies the catalog that contains the entry to be deleted.

password
specifies the password of the catalog that contains the entries to be deleted.

If entries to be deleted are password-protected and the catalog is also password-protected, a password must be supplied either through CATALOG or with the name of each entry to be deleted. The password that must be specified for each entry type is given under the description of the entryname parameter.

If a password is supplied through the CATALOG parameter, it takes precedence over any password supplied with the entryname parameter and is used for all entries to be deleted.

A JOBCAT or STEPCAT DD statement may be needed when dynamic allocation is required, because the name of the catalog cannot be passed to dynamic allocation via the CATALOG parameter.

Abbreviation: CAT

ERASE|NOERASE
specifies whether the data component of a cluster or alternate index to be deleted is 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. When you specify ERASE, you must identify the catalog of the entry that is still to be deleted with a JOBCAT or STEPCAT DD statement unless:

- The entry is in the master catalog, or
- The first qualifier in the entry's qualified name is the catalog's name or alias.

ERASE
specifies that the data component is 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.

Abbreviation: ERAS

NOERASE
specifies that the data component is not to be overwritten with binary zeros when the cluster or alternate index is deleted.

Abbreviation: NERAS

FILE(ddname**)**
specifies the name of the DD statement that identifies:

- The volume that contains a unique data set to be deleted.
- The partitioned data set from which a member (or members) is to be deleted.
- The data set to be deleted if ERASE is specified.
- The volume that contains the data space to be deleted.
- The catalog recovery volume(s) when the entry being deleted is in a recoverable catalog. If the volumes are of a different device type, concatenated DD statements must be used. (The catalog recovery volume is the volume whose recovery space contains a copy of the entry being deleted.) (For more details on catalog recovery volume mounting requirements, see Catalog Administration Guide.)
- The volume(s) owned by the catalog when you delete a catalog using the FORCE option.

If you do not specify FILE and VSAM requires access to a volume or volumes during the delete processing, VSAM will attempt to dynamically allocate the volumes. When ERASE is specified and FILE is not specified, the entryname will attempt to be dynamically allocated. Dynamic allocation requires that the volume(s) must 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, concatenated DD statements must be used. All volumes that contain associations to a cluster being deleted must also be included on the ddcard referenced by the FILE parameter.

When deleting multivolume non-VSAM data sets with the SCRATCH option, DELETE SCRATCH processing will require access to each volume in the entry's catalog record before the scratch can be issued. This will require either all volumes to be mounted, online, and allocable 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 demount/mount requests to be issued for the unit as required during delete processing. If access to the volumes cannot be provided, then DELETE NOSCRATCH can be used to uncatalog the non-VSAM data set and the user will assume the responsibility of scratching the format-1 DSCBs from all the volumes.

FORCE|NOFORCE
specifies whether objects that are not empty should be deleted.

FORCE
allows you to delete data spaces, generation data groups, and user catalogs without first ensuring that these objects are empty.

If you delete a data space using FORCE:

- All VSAM data spaces on the volume(s) are scratched from the VTOC. VSAM's ownership of the volume(s) is given up.
- Before the data spaces are scratched, VSAM verifies that no VSAM clusters on the volume are open. The clusters remain cataloged, but their entries are marked unusable. Subsequently, you can delete the cluster from the catalog.
- You cannot specify FORCE to delete the space on a volume that contains the VSAM master catalog or a user catalog.
- You must supply the master password of the catalog that contains the volume's entry.

If you delete a generation data group using FORCE:

- The GDG entry is deleted even though it might point to non-VSAM entries in the catalog.
- Each 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. The data set can be located with its DSCB in the volume's VTOC.

If you delete a user catalog using FORCE:

- The catalog is deleted even though it contains catalog entries for objects that have not been deleted. All cataloged objects are automatically deleted, but the contents of each cluster and alternate index are not erased.
- Before data spaces are deleted, VSAM verifies that no VSAM clusters on the volume are open. All data spaces are deleted from each volume owned by the catalog. The volumes are then available to be used by the system or to be assigned to other VSAM catalogs.
- VSAM clusters and alternate indexes, even though they might reside in unique data spaces, cannot be located, because the data space's DSCB is scratched when the catalog is deleted.
- Non-VSAM data set entries in the catalog are deleted, but the non-VSAM data set's space on the volume is undisturbed. The non-VSAM data set can be located with its DSCB in the volume's VTOC.
- You must supply the master password of the catalog being deleted.

Abbreviation: FRC

NOFORCE
causes the DELETE command to terminate when you request the deletion of a data space generation data group, or catalog that is not empty.

Abbreviation: NFRC

PURGE|NOPURGE
specifies whether the entry is to be deleted regardless of the retention period specified. This parameter cannot be used if a data space entry is to be deleted.

PURGE
specifies that the entry is to be deleted even if the retention period, specified in the TO or FOR parameter, has not expired.

Abbreviation: PRG

NOPURGE
specifies that the entry is not to be deleted if the retention period has not expired.

Abbreviation: NPRG

SCRATCH|NOSCRATCH
specifies whether a data set is to be removed from the VTOC of the volume on which it resides. This parameter can only be specified for a cluster, an alternate index, a page space, or a non-VSAM data set.

SCRATCH
specifies that a non-VSAM data set or a data space is to be removed from the VTOC of the volume on which it resides.

When SCRATCH is specified for a cluster, alternate index, or page space defined with the UNIQUE attribute, the entry for the data set is removed from the VTOC. For a cluster, alternate index, or page space defined with the SUBALLOCATE attribute, the entry is deleted from the catalog and thus removed from the data space. The entry is not removed from the VTOC.

When you specify SCRATCH, you must identify the catalog of the entry to be deleted with a JOBCAT or STEPCAT DD statement, unless the entry is in the master catalog, or the first qualifier in the entry's qualified name is the catalog's name or alias, or the FILE parameter is specified.

To remove the format-1 DSCB from the VTOC you should use the SCRATCH function of the OS/VS IEHPROGM utility.

Note: If you specify SCRATCH when deleting a non-VSAM data set that was defined with a device type named by the user, the DELETE will fail.

Abbreviation: SCR

NOSCRATCH
specifies that the entry is to be deleted from the catalog only, without mounting the volume that contains the object defined by the entry.

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).

NOSCRATCH cannot be specified for VSAM clusters, alternate indexes, or page spaces in VSAM recoverable catalogs.

For more detailed information about the use of DELETE NOSCRATCH, see "VSAM Catalog Cleanup" in Catalog Administration Guide.

Abbreviation: NSCR

DELETE EXAMPLES

Deleting a Non-VSAM Data Set's Entry: Example 1

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.

```
//DELET4   JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//DD1      DD    VOL=SER=VSER02,UNIT=DISK,DISP=OLD,
//   DSNAME=EXAMPLE.NONVSAM
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
          EXAMPLE.NONVSAM -
          FILE (DD1) -
          PURGE -
          CATALOG(USERCAT4/USERUPPW)
/*
```

The DELETE command deletes the non-VSAM data set EXAMPLE.NONVSAM. Its parameters are:

- EXAMPLE.NONVSAM, which is the entryname of the object to be deleted.
- FILE, which 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, which specifies that the non-VSAM data set's retention period or date is to be ignored.
- CATALOG, which identifies the catalog that contains the entries, USERCAT4, and its update password, USERUPPW.

Deleting a Key-Sequenced VSAM Cluster in a Catalog: Example 2

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 dynamically allocates the key-sequenced data set so that the data can be overwritten (as specified by the ERASE option).

```
//DELET1   JOB   ...
//JOBCAT   DD    DSN=D27UCAT2,DISP=SHR
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
          D40.EXAMPLE.KSDS1 -
          PURGE -
          ERASE -
          CATALOG(D27UCAT2/USERMRPW)
/*
```

Job control language statement:

- JOBCAT DD, which makes a catalog available for the job: D27UCAT2.

The DELETE command deletes the key-sequenced VSAM cluster from the D27UCAT2 catalog. Its parameters are:

- D40.EXAMPLE.KSDS1, which is the entryname of the object being deleted. D40.EXAMPLE.KSDS1 is a key-sequenced VSAM cluster.
- PURGE, which specifies that the cluster is to be deleted regardless of its retention period or date.
- ERASE, which specifies that the cluster's data component is to be overwritten with binary zeros, regardless of a NOERASE attribute that might have been specified when the cluster was defined or altered.
- CATALOG, which identifies the catalog that contains the cluster's entries, D27UCAT2. The catalog's master password is required, unless the entry's master password is specified with the entryname.

Deleting an Entry-Sequenced VSAM Cluster in a VSAM Catalog and All Empty Data Spaces On a Volume: Example 3

In this example, an entry-sequenced VSAM cluster is deleted. Alternate indexes and paths related to the entry-sequenced cluster are automatically deleted by access method services. Next, all empty VSAM data spaces on volume VSER03 are deleted, and the volume's volume entry is deleted.

```
//DELET2   JOB  ...
//JOBCAT   DD   DSNAME=D27UCAT2,DISP=OLD
//STEP1    EXEC PGM=IDCAMS
//DD2      DD  VOL=SER=VSER03,UNIT=DISK,DISP=OLD
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
     DELETE -
          EXAMPLE.ESDS2/ESD2MRPW -
          PURGE -
          CLUSTER
     DELETE -
          VSER03 -
          SPACE -
          FILE (DD2)
/*
```

Job control language statement:

- JOBCAT DD, which makes a catalog available for the job: D27UCAT2.

The first DELETE command deletes the only VSAM cluster on volume VSER03. The DELETE command's parameters are:

- EXAMPLE.ESDS2, which is the entryname of the cluster to be deleted, and ESD2MRPW, which is the cluster's master password.
- PURGE, which specifies that the cluster is to be deleted regardless of its retention period or date. Alternate indexes and paths related to the entry-sequenced cluster are also deleted, regardless of their retention periods or dates.
- CLUSTER, which specifies that the entryname EXAMPLE.ESDS2 identifies a VSAM cluster entry.

The second DELETE command causes VSAM to examine each data space cataloged in the volume's entry. If the data space is empty, it is deleted. VSAM deletes the volume entry when there are no data spaces cataloged in it, and when there are no data sets identified that specify the volume as a candidate volume. VSAM also updates the volume's VTOC to reflect the deleted data spaces. The DD2 DD statement ensures that volume VSER03 is mounted when its data spaces are deleted.

In this example, JCL has been used instead of allowing access method services to dynamically allocate the volume. If the volume were mounted as permanently resident or reserved, access method services could dynamically allocate the volume without the need of JCL or the FILE parameter. The DELETE command's parameters are:

- VSER03, which identifies the volume that contains empty VSAM data spaces.
- SPACE, which specifies that the entryname identifies a volume entry. When a volume's data spaces are to be deleted, the SPACE parameter is required.
- FILE, which specifies the ddname of a DD statement that describes the VSAM volume and causes it to be mounted.

Deleting Two Key-Sequenced Clusters in a Catalog: Example 4

In this example, two key-sequenced clusters, D40.MYDATA and ENTRY, are deleted from a catalog. This example illustrates how more than one cataloged object is deleted with a single DELETE command.

```
//DELET3   JOB   ...
//JOBCAT   DD    DSNAME=USERCAT3,DISP=SHR
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
          (D40.MYDATA -
          ENTRY) -
          PURGE -
          CLUSTER
/*
```

Job control language statement:

- JOBCAT DD, which makes the catalog USERCAT3 available for the job.

The DELETE command deletes the key-sequenced clusters D40.MYDATA and ENTRY. The high-level qualifier of the object D40.MYDATA causes the catalog D27UCAT1 to be searched, because it is assumed that D40 is an alias of the catalog D27UCAT1. The object ENTRY is found in the catalog USERCAT3, which is made available to the job through the JOBCAT DD statement. Both entries will be dynamically allocated so that their respective catalog recovery areas can be updated. The DELETE command's parameters are:

- D40.MYDATA and ENTRY, which identify the objects to be deleted. D40.MYDATA and ENTRY are the entrynames of two key-sequenced clusters. It is assumed that neither cluster is password-protected at this time. If either cluster is password-protected, the operator is prompted to supply the cluster's master password.
- PURGE, which specifies that the cluster is to be deleted regardless of its retention period or date.
- CLUSTER, which specifies that D40.MYDATA and ENTRY identify cluster catalog records.

Deleting Nonempty VSAM Data Spaces on a Volume for a VSAM Catalog: Example 5

In this example, all of the data spaces on volume VSER05 are deleted, whether or not they are empty. All VSAM data spaces on the volume are scratched from the VTOC and the ownership of the volume is surrendered by the catalog, AMASTCAT. Any clusters or alternate indexes contained within the data spaces on volume VSER05 are marked unusable in the catalog. Because all VSAM data spaces on the volume are deleted, the volume's catalog entry is also deleted.

```
//DELET5  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
          VSER05 -
          SPACE -
          FORCE -
          CATALOG(AMASTCAT/USERUPPW)
/*
```

The DELETE command's parameters are:

- VSER05, which identifies the volume with its serial number. Because the volume's VTOC must be updated, access method services will dynamically allocate the volume.
- SPACE, which specifies that a volume entry is to be modified or deleted. When a volume's data space is to be deleted, the SPACE parameter is required.
- FORCE, which specifies that all VSAM data spaces on the volume are to be deleted, whether or not they are empty, and that volume ownership is to be released.
- CATALOG, which identifies the catalog that owns the volume, AMASTCAT. The catalog's update- or higher-level password is required.

Deleting a User Catalog: Example 6

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 except the catalog's volume cataloged in it. If the catalog is not empty, it cannot be deleted unless the FORCE parameter is specified.

```
//DELET6  JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
          D27UCAT1/USERMRPW -
          PURGE -
          USERCATALOG
/*
```

The DELETE command deletes the catalog. If a VSAM catalog is deleted, the volume that contained the catalog is now available for ownership by another VSAM catalog. The DELETE command also deletes the catalog's user catalog connector entry in the master catalog. The DELETE command's parameters are:

- D27UCAT1, the name of the user catalog. The catalog's update (or higher level) password is required.
- PURGE, which specifies that 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, it will not be deleted.
- USERCATALOG, which identifies D27UCAT1 as a user catalog.

Deleting an Alias Entry in a Catalog: Example 7

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/USERMRPW)
/*
```

The DELETE command removes an alias entry from catalog USERCAT4. Its parameters are:

- EXAMPLE.NONVSAM1, the entryname of the object to be deleted. EXAMPLE.NONVSAM1 identifies an alias entry.

- ALIAS, which specifies the type of entry to be deleted. VSAM verifies that EXAMPLE.NONVSAM1 is an alias entry, then deletes it. If EXAMPLE.NONVSAM1 identified another entry by mistake, VSAM would not delete the entry, but would note the discrepancy with a message to the programmer.
- CATALOG, which identifies the catalog that contains the entry, USERCAT4, and its master password, USERMRPW.

Deleting Generically Named Entries in a Catalog: Example 8

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 this example shows how all catalog entries with the same generic name are deleted.

```
//DELET8   JOB  ...
//JOBCAT   DD   DSNAME=USERCAT4,DISP=OLD
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     DELETE -
          GENERIC.*.BAKER -
          PURGE -
          CATALOG(USERCAT4/USERMRPW)
/*
```

Job control language statement:

- JOBCAT DD, which makes a catalog available for this job: 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 name GENERIC.*.BAKER.
- PURGE, which specifies that each entry is to be purged regardless of the retention period or date specified when it was defined.
- CATALOG, which identifies the catalog that contains the entries, USERCAT4, and its master password, USERMRPW.

List a Generation Data Group's Entries, Then Delete the Group and Its Data Sets in a Catalog: Example 9

In this example, a generation data group, GDG01, and its associated (generation data set) entries are listed. Next, the only generation data set in the group is deleted. Finally, 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
/*
```

Because no catalog was specified in a CATALOG parameter or with a JOBCAT or STEPCAT DD statement, VSAM assumes all entries required for this command sequence are in the master catalog.

The LISTCAT command lists the generation data group GDG01 and its associated generation data set entries. Its parameters are:

- ENTRIES, which specifies that the entry GDG01 is to be listed. Because the entry GDG01 is a generation data group entry, its associated generation data set's (non-VSAM) entries are also listed. In addition, if one of the generation data sets has aliases, the alias entries associated with the generation data set's entry are listed.
- ALL, which 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 in the generation data group: GDG01.G0001V00. Its parameters are:

- GDG01.G0001V00, the entryname of the object being deleted. GDG01.G0001V00 identifies the only generation data set in the generation data group GDG01.
- PURGE, which specifies that the generation data set's retention period or date is to be ignored.

The second DELETE command removes the catalog entry of the generation data group base from the catalog. Its parameters are:

- GDG01, the entryname of the object being deleted. GDG01 identifies the generation data group base entry.
- GENERATIONDATAGROUP, which specifies the type of entry to be deleted. VSAM verifies that GDG01 is a generation data group entry, then deletes it. If GDG01 incorrectly specified another type of entry, VSAM would not delete the entry, but would note the discrepancy with a message to the programmer.
- PURGE, which specifies that the generation data group's retention period or date is to be ignored.

Deleting a Member of a Partitioned (Non-VSAM) Data Set in a Catalog: Example 10

In this example, the MEM1 member of partitioned data set EXAMPLE.NONVSAM2 is deleted. Next, the non-VSAM data set itself is deleted.

```
//DELET10    JOB   ...
//JOBCAT     DD    DSNAME=USERCAT4,DISP=SHR
//STEP1      EXEC  PGM=IDCAMS
//SYSPRINT   DD    SYSOUT=A
//SYSIN      DD    *
      DELETE -
           EXAMPLE.NONVSAM2(MEM1) -
           CATALOG(USERCAT4/USERMRPW)
      DELETE -
           EXAMPLE.NONVSAM2 -
           PURGE -
           CATALOG(USERCAT4/USERMRPW)
/*
```

Job control language statement:

- JOBCAT DD, which makes a catalog available for this job: USERCAT4.

The first DELETE command deletes one of the members of a partitioned data set, EXAMPLE.NONVSAM2(MEM1), from the user catalog, USERCAT4. Its parameters are:

- EXAMPLE.NONVSAM2(MEM1), which is the entryname of one of the members of the partitioned (non-VSAM) data set, EXAMPLE.NONVSAM2. The entryname identifies the object to be deleted.
- CATALOG, which identifies the catalog that contains the entry, USERCAT4, and its master password, USERMRPW.

The second DELETE command deletes all remaining members, as well as the data set itself, of the partitioned non-VSAM data set EXAMPLE.NONVSAM2. Its parameters are:

- EXAMPLE.NONVSAM2, which is the entryname of the object to be deleted.
- PURGE, which specifies that the non-VSAM data set's retention period or date is to be ignored. If PURGE had not been specified and the non-VSAM data set's retention period has not yet expired, VSAM would not delete its entry.
- CATALOG, which identifies the catalog that contains the entry, USERCAT4, and its master password, USERMRPW.

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.

Deleting a Page Space: Example 11

In this example, page space SYS1.PAGE2 is deleted.

```
//DELET11    JOB   ...
//STEP1      EXEC  PGM=IDCAMS
//SYSPRINT   DD    SYSOUT=A
//SYSIN      DD    *
     DELETE -
          SYS1.PAGE2/MASTMPW1 -
          PURGE -
          PAGESPACE
/*
```

The DELETE command removes the page space entry, SYS1.PAGE2, from the master catalog. Its parameters are:

- SYS1.PAGE2, the entryname of the object to be deleted. SYS1.PAGE2 identifies a page space entry. The catalog's update- or higher-level password is also required.
- PURGE, which specifies that the page space entry is to be deleted regardless of the retention period or date specified when it was defined.
- PAGESPACE, which specifies the type of entry to be deleted. VSAM verifies that SYS1.PAGE2 is a page space entry, then deletes it. If SYS1.PAGE2 incorrectly identified another type of entry, VSAM would not delete it, but would send an error message to the programmer.

EXAMINE

The EXAMINE command analyzes and reports on the structural consistency of the index component and/or data component of a key-sequenced data set (KSDS) cluster. The format of the EXAMINE command is:

EXAMINE	**NAME(**clustername[/password]**)** [INDEXTEST|NOINDEXTEST] [DATATEST|NODATATEST] [ERRORLIMIT(value)]

EXAMINE PARAMETERS

Required Parameters

NAME(clustername[/password]**)**
specifies which user cluster is to be analyzed for structural consistency by EXAMINE. EXAMINE expects the user to have specified a particular key-sequenced data set by its cluster name. The appropriate cluster component is then selected dynamically during program execution. When analyzing a VSAM catalog, use a STEPCAT DD statement to point to that catalog and apply a VERIFY DATASET command before using the EXAMINE command. A STEPCAT DD statement is not required for the master catalog.

clustername
specifies the identification of the key-sequenced data set cluster to be analyzed.

password
specifies, optionally, the master password associated with a catalog or the control password for a data set.

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

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 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 2147483647 errors, but you can specify any number between 0 and 2147483647. Note that testing continues even though the error limit is reached.

Abbreviation: ELIMIT

EXAMINE EXAMPLES

Examine Both Components of a Key-Sequenced Data Set: Example 1

This example shows how to get a list of data set structural errors that can be used 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, which specifies the cluster name only. The data set has no control level password protection.
- INDEXTEST, which causes the index component to be examined.
- DATATEST, which 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 a VSAM User Catalog: Example 2

This example shows how to determine whether your VSAM catalog has structural errors.

```
//EXAMEX3   JOB
//STEP1     EXEC    PGM=IDCAMS
//STEPCAT   DD      DSN=VCAT01.VCAT, DISP=SHR
//SYSPRINT  DD      SYSOUT=A
//SYSIN     DD      *
     VERIFY DATASET(VCAT01.VCAT)
     EXAMINE -
        NAME(VCAT01.VCAT) -
        NOINDEXTEST -
        DATATEST -
        ERRORLIMIT(1000)
/*
```

Use the EXAMINE command to analyze the data component of a VSAM catalog. Its parameters are:

- NAME, which specifies the catalog name. The catalog must be connected to the master catalog.
- NOINDEXTEST, which specifies that the index component is not to be examined.
- DATATEST, which causes the data component to be examined.
- ERRORLIMIT(1000), which restricts the printing of detailed error messages to 1000 errors.

Note that the STEPCAT DD statement and the VERIFY DATASET command are required when you use EXAMINE against a VSAM user catalog.

EXPORT DISCONNECT

The EXPORT DISCONNECT command disconnects a user catalog. The format of this command is:

EXPORT	usercatname[/password] **DISCONNECT**

EXPORT DISCONNECT PARAMETERS

Required Parameters

usercatname[/password]
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 update- or higher-level password of the master catalog. RACF: The update or higher RACF authority for the master catalog is required.

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 volume that contains a VSAM user catalog can be physically moved to the system to which the catalog will be connected.

A JOBCAT/STEPCAT can be used to delete a user catalog connector entry from a user catalog. Any alias entries will also be deleted. In this case, the password should be the master password of the user catalog specified in the JOBCAT/STEPCAT.

Note: 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.

Abbreviation: DCON

Export Disconnect of a User Catalog: Example 1

In this example, the user catalog D27UCAT1 is exported and 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 -
          D27UCAT1/MASTMPW1 -
          DISCONNECT
/*
```

The EXPORT command removes the user catalog connector entry for D27UCAT1 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 EXPORT command's parameters are:

- D27UCAT1, which identifies the object to be disconnected. When a user catalog is disconnected, the master catalog's master password is required.
- DISCONNECT, which identifies the object as a user catalog. When a user catalog's connector entry is to be deleted, DISCONNECT is required.

EXPORT

The EXPORT command exports a VSAM cluster or an alternate index. The format of this command is:

EXPORT	entryname[/password] {OUTFILE(ddname)|OUTDATASET(entryname)} [ERASE|NOERASE] [INFILE(ddname)] [INHIBITSOURCE|NOINHIBITSOURCE] [INHIBITTARGET|NOINHIBITTARGET] [PURGE|NOPURGE] [TEMPORARY|PERMANENT]

EXPORT can be abbreviated: EXP

EXPORT PARAMETERS

Required Parameters

entryname[/password]
names the cluster or alternate index to be exported. This parameter must be the first parameter following EXPORT.

password
If you are exporting:

- A cluster or alternate index, you must supply the object's master password. RACF: The alter RACF authority to the cluster or alternate index is required.
- A cluster or alternate index with any password-protected paths defined over them, you must supply the master password of the catalog containing the objects in order to export the protection attributes of all the paths. RACF: The alter RACF authority to the catalog is required.

OUTFILE(ddname)|OUTDATASET(entryname)
specifies the name of the DD statement or the data set that is to receive the data being exported.

OUTFILE(ddname)
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 may be specified in the DD statement to override this default and improve performance.

Abbreviation: OFILE

OUTDATASET(entryname)
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 "For a Target Data Set" on page 10.

Abbreviation: ODS

Optional Parameters

ERASE|NOERASE
specifies whether the data component of the cluster or alternate index to be exported is 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).

ERASE
specifies that the data component is 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.

Abbreviation: ERAS

NOERASE
specifies that the data component is not to be overwritten with binary zeros when the cluster or alternate index is deleted.

Abbreviation: NERAS

INFILE(ddname)
specifies the name of the DD statement that identifies the cluster or alternate index to be exported. If the cluster or alternate index has been defined with a maximum logical record length greater than 32760 bytes, EXPORT processing terminates with an error message.

In addition to the DD statement for INFILE, you must identify the entry's catalog with a JOBCAT or STEPCAT DD statement unless:

- The object's entry is in the master catalog, or
- The first qualifier of the object's name is the catalog's name or alias.

When INFILE and its DD statement are not specified for an object that is to be exported, an attempt is made to dynamically allocate it.

Abbreviation: IFILE

INHIBITSOURCE|NOINHIBITSOURCE
specifies whether the original data records (the data records of the source cluster or alternate index) can be accessed for any operation other than retrieval after a copy is exported. This specification can later be altered through the ALTER command.

INHIBITSOURCE
specifies that the source object in the original system cannot be accessed for any operation other than retrieval. This parameter can be specified only 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.)

Abbreviation: INHS

NOINHIBITSOURCE
specifies that the original data records in the original system can be accessed for any kind of operation.

Abbreviation: NINHS

INHIBITTARGET|NOINHIBITTARGET
specifies whether 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
specifies that the target object cannot be accessed for any operation other than retrieval after it has been imported into another system.

Abbreviation: INHT

NOINHIBITTARGET
specifies that the target object can be accessed for any type of operation after it has been imported into another system.

Abbreviation: NINHT

PURGE|NOPURGE
specifies whether 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.

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

TEMPORARY|PERMANENT
specifies whether the cluster or alternate index to be exported is to be deleted from the original system.

TEMPORARY
specifies that the cluster or alternate index 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.

Note: Be sure to properly protect the file of the temporary object if you want to deny unauthorized access to that file.

Abbreviation: TEMP

PERMANENT
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.

Abbreviation: PERM

EXPORT EXAMPLES

Exporting a Key-Sequenced Cluster: Example 1

In this example, a key-sequenced cluster, D40.EXAMPLE.KSDS1, is exported from a user catalog, D27UCAT1. 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 -
          D40.EXAMPLE.KSDS1 -
          OUTFILE(RECEIVE) -
          TEMPORARY -
          INHIBITSOURCE
/*
```

Job control language statement:

- RECEIVE DD, which 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 the key-sequenced cluster, D40.EXAMPLE.KSDS1, to a portable file, TAPE2, and modifies the cluster's catalog entries (its cluster, data, and index entries). Access method services attempts to dynamically allocate D40.EXAMPLE.KSDS1. Because the high-level qualifier of the cluster name is the name of the alias of D27UCAT2, the cluster can be allocated without specifying a JOBCAT or STEPCAT DD statement. The EXPORT command's parameters are:

- D40.EXAMPLE.KSDS1, which identifies the cluster to be exported. Because the cluster is not password-protected, no password is provided with the cluster's entryname.
- OUTFILE, which 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, which 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, "Importing a Key-Sequenced Cluster.")
- INHIBITSOURCE, which 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.

Exporting an Entry-Sequenced Cluster: Example 2

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 -
          D50.EXAMPLE.ESDS1 -
          OUTFILE(RECEIVE) -
          PURGE
/*
```

Job control language statement:

- RECEIVE DD, which 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, D50.EXAMPLE.ESDS1, to a portable file, TAPE1. The cluster is deleted from the system after it is copied into the portable file. Access method services attempts to dynamically allocate D50.EXAMPLE.ESDS1. Because the high-level qualifier of the cluster is the name of the alias of the catalog D27UCAT2, the cluster will attempt to be dynamically allocated without specifying a JOBCAT or STEPCAT DD statement. The EXPORT command's parameters are:

- D50.EXAMPLE.ESDS1, which identifies the entry-sequenced cluster to be exported. When a cluster is exported, its master password is required if it is password-protected.
- OUTFILE, which 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, which allows the cluster to be deleted regardless of its retention period or date.

Because EXPORT defaults to PERMANENT, access method services assumes the cluster is to be deleted after TAPE1 contains a copy of it.

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.

EXPORTRA

The EXPORTRA command recovers VSAM and non-VSAM catalog entries from catalog recovery areas, and recovers data for VSAM clusters and alternate indexes by means of catalog recovery areas. This command is for use with recoverable catalogs only.

The format of the EXPORTRA command is:

```
EXPORTRA  OUTFILE(ddname)
          CRA(
            [ddname1
              [ALL[  INFILE(ddname2)]|
                ENTRIES(
                  [entryname[  ddname3]]
                  [[entryname[  ddname3]]...]]|
                NONE})
            [[ddname1...)...]])
          [FORCE|NOFORCE]
          [MASTERPW(password)]
```

EXPORTRA can be abbreviated: XPRA

EXPORTRA PARAMETERS

Required Parameters

OUTFILE(ddname)
identifies the DD statement that describes the data set (usually a movable disk pack or magnetic tape reel) that is to contain a copy of the VSAM objects and catalog entries identified with the CRA parameter.

With the exception of block size, the data set characteristics of the target data set that is to contain the cluster to be exported should not be specified. Block size is defaulted by the EXPORTRA command to 2048. Block size may be specified in the DD statement to override this default and improve performance.

The characteristics of the target data set are described in "For a Target Data Set" on page 10.

Abbreviation: OFILE

CRA(
[ddname1
[ALL[INFILE(ddname2)]|
ENTRIES(
[entryname[ddname3]]
[[entryname[ddname3]]...]]|
NONE})
[[ddname1...)...])
The CRA parameter and its subparameters describe the catalog recovery area(s) to be referred to, identify the specific entries to be copied from the catalog recovery area, and specify the volumes that contain the entries' data. A separate set of subparameters is specified to describe each catalog recovery area to which it refers.

ddname1
identifies the DD statement that describes a volume to whose catalog recovery area is referred. Each ddname1 you specify is followed by one of the following subparameters, which specifies the action to be taken with regard to the catalog recovery area:

ALL
specifies that each entry in ddnamel's catalog recovery area is to be copied to the OUTFILE data set. In addition, the contents (that is, data records in logical sequential order) of each VSAM cluster and alternate index are copied, following the copy of the catalog recovery area entry.

If any of these VSAM clusters or alternate indexes has data that resides on volumes other than that identified via ddnamel, you use INFILE to describe all the volumes.

INFILE(ddname2**)**
identifies the DD statement that describes multiple volumes to be used in the recovery process. INFILE is required if the VSAM clusters and/or alternate indexes whose entries are contained in ddnamel's catalog recovery area reside on multiple volumes. The ddname2 DD statement should include all volumes containing the data set, including the recovery area volume.

Abbreviation: IFILE

ENTRIES(
(entryname[ddname3])
[(entryname[ddname3])...])
names each entry in the catalog recovery area that is to be copied to the OUTFILE data set. All other entries in the catalog recovery area are not copied.

entryname
names an object whose catalog entry is contained in the ddnamel volume's catalog recovery area. You can specify the entryname of the following entry types: alternate index, cluster, and non-VSAM.

When you specify a cluster entry, the cluster's data and index entries and all associated path entries are also copied. In addition, the cluster's contents (that is, its data records, in logical sequential order) are copied, following the copies of the cluster's catalog entries.

When you specify an alternate index entry, the alternate index's entries and contents and all associated path entries are copied as though it were a key-sequenced cluster.

ddname3
identifies the DD statement that describes each volume that contains part of the cluster or alternate index. ddname3 is ignored when specified for a non-VSAM entry.

ddname3 is required when the entry is a VSAM cluster or alternate index whose data resides on volumes other than the volume identified with the ddnamel DD statement. The ddname3 DD statement must also include the volume referenced by the ddnamel DD statement.

Abbreviation: ENT

NONE
specifies that no object in the volume's catalog recovery area is to be copied. You must specify ddnamel NONE for each volume referenced in a ddname2 or ddname3 DD statement if you do not want the entries in its catalog recovery area to be copied.

Note: If ddname2 contains both multiple volume and single volume entries, a NONE statement is required to

prevent processing of the single volume entries that exist on the second and subsequent volumes. The NONE parameter must be specified before the ALL parameter. (See "Using EXPORTRA for All Entries on Multiple Volumes: Example 2" on page 182.)

Optional Parameters

FORCE|NOFORCE
specifies whether the catalog entry's copy is to be copied from the catalog recovery area, even though the copy itself might be inaccurate.

FORCE
specifies that the object's catalog entry copy is copied from the volume's catalog recovery area, even though a loss of data integrity might result. Time stamp and extent mismatches are ignored.

Abbreviation: FRC

NOFORCE
specifies that the catalog recovery area's copy is not copied if it is not accurate, and the EXPORTRA command terminates with an error message.

Abbreviation: NFRC

MASTERPW(password)
specifies the master catalog's master password. The master catalog's master password is required when the master catalog is password-protected. RACF: The alter RACF authority to the master catalog is required.

Abbreviation: MRPW

Using EXPORTRA for All Entries on One Volume: Example 1

This example shows the EXPORTRA function for one volume, V0, owned by the VSAM master catalog. All the entries contained in the catalog recovery area on V0 and the data sets themselves are copied to a SAM data set on another disk volume. All the data sets reside entirely within the one volume.

```
//EXPORTRA JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//VLV0     DD   UNIT=DISK,VOL=SER=V0,AMP='AMORG',DISP=OLD
//VOLOUT   DD   UNIT=DISK,VOL=SER=333003,DSN=OUT.FILE,
//         DISP=(NEW,KEEP),SPACE=(CYL,(5,1)),
//         DCB=BLKSIZE=6000
//SYSIN    DD   *
     EXPORTRA -
        OUTFILE(VOLOUT) -
            CRA ( -
            (VLV0 ALL))
/*
```

Job control language statements:

- VLV0 DD, which identifies and allocates the volume whose catalog recovery area contents are to be copied. The data sets reside wholly within the one volume.
- VOLOUT DD, which identifies the sequential file that is to receive the copied data sets and catalog recovery records. The DCB BLKSIZE parameter overrides the EXPORTRA default of 2048 to improve performance.

The EXPORTRA command copies everything appearing in the catalog recovery area of volume V0. The EXPORTRA command parameters are:

- OUTFILE, which is required and identifies the sequential access method (SAM) non-VSAM data set that is to receive the copied information.
- CRA, which is required and identifies the catalog recovery area and volume from which the copy is to take place. VLV0 identifies the DD statement for the catalog recovery volume. The ALL subparameter specifies that all entries are to be copied from the catalog recovery area. The INFILE subparameter is not required, because the data sets are contained wholly within the one volume.

Using EXPORTRA for All Entries on Multiple Volumes: Example 2

This example shows the EXPORTRA function for one volume owned by the VSAM master catalog. All data sets contained on the volume, V0, and their corresponding CRA entries describing the data sets are copied to a sequential (SAM) data set on a tape volume. The catalog recovery area contains data sets which reside on volumes V1 and V2 in addition to the recovery volume, V0. The entries in the catalog recovery areas of the additional volumes are not to be recovered.

```
//EXPORTRA   JOB   ...
//STEP1      EXEC  PGM=IDCAMS
//SYSPRINT   DD    SYSOUT=A
//VLV0       DD    UNIT=DISK,VOL=SER=V0,AMP='AMORG',
//              DISP=OLD
//VLV1       DD    UNIT=DISK,VOL=SER=V1,AMP='AMORG',
//              DISP=OLD
//VLV2       DD    UNIT=DISK,VOL=SER=V2,AMP='AMORG',
//              DISP=OLD
//VOLA       DD    UNIT=(DISK,3),VOL=SER=(V0,V1,V2),
//              AMP='AMORG',DISP=OLD
//VOLOUT     DD    DSN=OUT.FILE,LABEL=(1,SL),
//              DISP=(NEW,KEEP),
//              UNIT=TAPE,VOL=SER=TAPE01,
//              DCB=(BLKSIZE=8192,DEN=3)
//SYSIN      DD    *
     EXPORTRA -
        OUTFILE(VOLOUT) -
        CRA ( -
            (VLV0 ALL INFILE(VOLA)) -
            (VLV1 NONE) -
            (VLV2 NONE) -
            (VLV1 ALL))-
        MASTERPW(MCATMRPW)
/*
```

Job control language statements:

- VLV0 DD, which identifies the volume whose catalog recovery area contents are to be exported.
- VLV1 DD, which identifies a volume containing part of a multiple volume data set to be exported and single volume data sets to be exported.
- VLV2 DD, which identifies a volume containing part of a multivolume data set to be exported. The catalog recovery area contents are not to be exported.
- VOLA DD, which identifies all the volumes containing the data sets to be exported.
- VOLOUT DD, which identifies the sequential tape file that is to receive the exported data sets and catalog recovery records. The DCB BLKSIZE parameter overrides the default of 2048 to improve performance.

The EXPORTRA command copies every entry appearing in the catalog recovery area of volume V0 and the data sets described by the CRA entries. Because they contain part of multivolume data sets, two additional volumes, V1 and V2, are referred to, but the entries in their catalog recovery areas are not exported. The EXPORTRA command parameters are:

- OUTFILE, which is required and identifies the sequential (SAM) non-VSAM data set that is to receive the exported information.
- CRA, which is required and identifies the catalog recovery area and volumes from which the entries are to be copied. VLV0 identifies the DD statement of the recovery volume. The ALL subparameter specifies that all entries in the catalog recovery area of the volume identified by VLV0 are to be copied.

  INFILE, which is required because the data sets to be copied are contained on multiple volumes. VOLA refers to the DD statement that identifies these multiple volumes, including the recovery volume.

  VLV1 NONE, which is required to prevent processing of single volume entries that exist on this volume that is referenced in the INFILE subparameter. This subparameter must precede VLV1 ALL.

VLV2 NONE, which is required because the volume it identifies is referenced in the INFILE subparameter.

VLV1 ALL is required and identifies the single volume data sets. This subparameter must come after VLV1 NONE.

- MASTERPW, which specifies the master password of the master catalog. This parameter is required when the master catalog is password protected.

Using EXPORTRA for Selected Entries: Example 3

This example shows the EXPORTRA function for selected entries from the catalog recovery area of one volume owned by the VSAM master catalog. Three data sets and their corresponding entries from the catalog recovery area on volume V0 are copied to a sequential (SAM) data set on another disk volume. The remainder of the data sets and the catalog recovery area entries are not copied. Two of the selected data sets are multivolume VSAM data sets. The data sets and their corresponding entries in the catalog recovery areas of the additional volumes are not recovered.

```
//EXPORTRA   JOB   ...
//STEP1      EXEC  PGM=IDCAMS
//SYSPRINT   DD    SYSOUT=A
//VLV0       DD    UNIT=DISK,AMP='AMORG',DISP=OLD,
//            VOL=SER=V0
//VLV1       DD    UNIT=DISK,AMP='AMORG',DISP=OLD,
//            VOL=SER=V1
//VLV2       DD    UNIT=DISK,AMP='AMORG',DISP=OLD,
//            VOL=SER=V2
//VOLA       DD    UNIT=(DISK,3),AMP='AMORG',DISP=OLD,
//            VOL=SER=(V0,V1,V2)
//VOLB       DD    UNIT=(DISK,2),AMP='AMORG',DISP=OLD,
//            VOL=SER=(V0,V1)
//VOLOUT     DD    UNIT=DISK,VOL=SER=333003,DSN=OUT.FILE,
//            DISP=(NEW,KEEP),SPACE=(CYL,(5,1))
//SYSIN      DD    *
     EXPORTRA -
        OUTFILE(VOLOUT) -
        CRA ( -
            (VLV0 ENTRIES ( -
               (LARGE.DATASET.A VOLA) -
               (LARGE.DATASET.B VOLB) -
               (LARGE.DATASET.C))) -
            (VLV1 NONE) -
            (VLV2 NONE)) -
            FORCE
/*
```

Job control language statements:

- VLV0 DD, which identifies the volume whose catalog recovery area contains the entries for the three VSAM data sets to be copied.
- VLV1 and VLV2 DD, each of which identifies a volume containing part of a multivolume data set but whose catalog recovery area contents are not to be copied.
- VOLA DD, which identifies the multivolumes of LARGE.DATASET.A.
- VOLB DD, which identifies the multivolumes of LARGE.DATASET.B.
- VOLOUT DD, which identifies the sequential file that is to receive the copied data sets and catalog recovery records.

The EXPORTRA command exports the three entries (LARGE.DATASET.A, LARGE.DATASET.B, and LARGE.DATASET.C) from the catalog recovery area on volume V0. Because they contain part of multivolume

data sets, two additional volumes are referred to, but the entries in their catalog recovery areas are not exported. The EXPORTRA command parameters are:

- OUTFILE, which is required and identifies the non-VSAM data set that is to receive the exported information.
- CRA, which is required and identifies the catalog recovery area and volumes from which the export is to take place. VLV0 identifies the DD statement of the recovery volume. The ENTRIES subparameter identifies the three entries which are to be copied.

  LARGE.DATASET.A requires the VOLA subparameter because the VSAM data set is contained on three volumes. Similarly, LARGE.DATASET.B requires the VOLB subparameter. However, LARGE.DATASET.C is contained wholly within the catalog recovery volume, and, therefore, does not require an additional ddname3 subparameter.

  The VLV1 NONE and VLV2 NONE subparameters are necessary, because the volumes they identify are referenced in previous ddname3 subparameters (VOLA and VOLB); however, the contents of their catalog recovery areas are not to be copied.
- FORCE, which specifies that timestamp mismatches are to be ignored.

IMPORT CONNECT

The IMPORT CONNECT command connects a user catalog. The format of this command is:

```
IMPORT   CONNECT
         OBJECTS((usercatname
           DEVICETYPE(devtype)
           VOLUMES(volser)))

         [CATALOG(mastercatname[/password])]
```

IMPORT CONNECT PARAMETERS

Required Parameters

CONNECT
specifies that a user catalog is to be connected. The user catalog is connected to the master catalog in the receiving system. When CONNECT is coded, OBJECTS must also be coded to name the catalog and to identify the volume serial number and device type of the volume that contains the user catalog.

Abbreviation: CON

OBJECTS((usercatname
DEVICETYPE(devtype)
VOLUMES(volser)))
specifies the user catalog to be connected.

usercatname
specifies the name of the user catalog being connected.

DEVICETYPE(devtype)
specifies the device type of the volume that contains a user 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 on which the user catalog resides.

VOLUMES is required when a user catalog is to be import connected. VSAM user catalogs may only be on one volume so only one volume should be specified when connecting a VSAM user catalog.

Optional Parameters

CATALOG(catname[/password])
specifies the name of the catalog in which the catalog you are connecting is to be defined. This parameter is required when the catalog is password-protected or when you want to direct the catalog's entry to a particular catalog other than the master catalog.

catname
is the name of the catalog in which the catalog you are connecting is to be defined. If the specified catalog is not the master catalog, the catalog must be identified by a JOBCAT or STEPCAT DD statement. If you are import connecting a user catalog, the specified catalog is usually the master catalog.

password
specifies the master catalog's update- or higher-level password. RACF: The update or higher RACF authority to the master catalog is required.

Abbreviation: CAT

Importing to Connect a User Catalog: Example 1

In this example, a user catalog, D27UCAT1, is connected to the system's master catalog, AMAST1. This example reconnects the user catalog, D27UCAT1, that was disconnected in the EXPORT DISCONNECT example.

```
//IMPORT1  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     IMPORT -
          OBJECTS( -
                (D27UCAT1 -
                VOLUME(VSER02) -
                DEVICETYPE(3380) ) -
                ) -
          CONNECT -
          CATALOG(AMAST1/MASTMPW1)
/*
```

The IMPORT command builds a user catalog connector entry that identifies the user catalog D27UCAT1 in the master catalog AMAST1. Its parameters are:

- OBJECTS, which is required when a user catalog is being imported. The subparameters of OBJECTS identify the user catalog, D27UCAT1; the user catalog's volume, VSER02; and the device type of the user catalog's volume, 3380.
- CONNECT, which 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, which identifies the master catalog, AMAST1, and specifies its update- or higher-level password, MASTMPW1.

IMPORT

The IMPORT command restores a VSAM cluster or alternate index from a portable data set created by the EXPORT command. The format of this command is:

```
IMPORT  {INFILE(ddname)|INDATASET(entryname)}
        {OUTFILE(ddname[/password])|
          OUTDATASET(entryname[/password])}
        [ERASE|NOERASE]
        [INTOEMPTY]
        [OBJECTS
          ((entryname
            [FILE(ddname)]
            [(KEYRANGES((lowkey  highkey)
            [(lowkey highkey)...]))]
            [NEWNAME(newname)]
            [ORDERED|UNORDERED]
            [VOLUMES(volser[  volser...]))])
          [(entryname...)...]))]
        [PURGE|NOPURGE]
        [SAVRAC|NOSAVRAC]

        [CATALOG(catname[/password]))]
```

IMPORT may be abbreviated: IMP.

IMPORT 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 or alternate index to be imported.

INFILE(ddname)
specifies the name of a DD statement that identifies the portable copy of the cluster or alternate index 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 specified via 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 or alternate index 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[/password]**)|OUTDATASET(**entryname[/password]**)**
specifies the name of a DD statement or the name of a cluster or alternate index 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:

- The data set's entry is in the master catalog, or
- The first qualifier of the data set's qualified name is the catalog's name or alias.

Notes:

1. When importing a cluster that was permanently exported, the OUTFILE parameter should be used.
2. 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.

OUTFILE(ddname[/password]**)**
specifies the name of a DD statement that identifies the data set name and volume(s) of the cluster or alternate index to be imported.

If the object was permanently exported and/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(s), the device type, DISP=OLD, and AMP='AMORG'.

If the object has its data and index components on different device types, do not specify OUTFILE—use OUTDATASET.

If NEWNAME 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. The volume that is to receive the imported cluster or alternate index must be owned by a VSAM catalog. Also, this should be the same name declared on the NEWNAME parameter. Failure to do so will result in the deletion of the original cluster.

password
is the update- or higher-level password of the target object. You must supply the password when the target data set is empty. (See the VOLUMES parameter for more information about importing into an empty data set.) Otherwise, IMPORT obtains the required VSAM data set passwords from the portable data set. RACF: The update or higher RACF authority to the target object is required.

Abbreviation:

OUTDATASET(entryname[/password]**)**
specifies the name of the cluster or alternate index that is to be imported. When you specify OUTDATASET, the VSAM data set you identify is dynamically allocated.

You may 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 declared on the NEWNAME parameter. Failure to do so will result in the deletion of the original cluster.

password
: is the update- or higher-level password of the target object. (See password under OUTFILE for password requirements of the target data set.) RACF: The update or higher RACF authority to the target object is required.

Abbreviation: ODS

Optional Parameters

CATALOG(catname[/password]**)**
: specifies the name of the catalog in which the imported object is to be cataloged. This parameter is required when the catalog is password-protected.

catname
: specifies the name of the catalog in which the entry to be imported is to be defined.

password
: specifies the catalog's update- or higher-level password. RACF: The update or higher RACF authority to the catalog is required. When you import an alternate index whose base cluster is password-protected, you should supply the catalog's master password. RACF: The alter RACF authority to the catalog is required. Otherwise, you (or the operator) will be prompted to supply the base cluster's master password. RACF: The alter RACF authority to the base cluster is required. When you import an integrated catalog facility user catalog and INTOEMPTY is not specified, the master password of the master catalog is required. RACF: The alter RACF authority to the master catalog is required.

Abbreviation: CAT

ERASE|NOERASE
: specifies whether the data component of the cluster or alternate index is to be erased (that is, overwritten with binary zeros). This parameter can be used only 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
: specifies that the data component is 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.

Abbreviation: ERAS

NOERASE
specifies that the data component is not to be overwritten with binary zeros when the cluster or alternate index is deleted.

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 will fail. The password and RACF profiles associated with the empty data set will be retained.

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 fails because a catalog entry with the same name already exists, the path on the portable data set is ignored.

Abbreviation: IEMPTY

OBJECTS
((entryname
[FILE(ddname**)]**
{KEYRANGES((lowkey highkey**){(**lowkey highkey**)...})}**
[NEWNAME(newname**)]**
[ORDERED|UNORDERED]
[VOLUMES(volser**[** volser...**])])**
[(entryname...**)...])]**
specifies the new or changed attributes for the cluster, alternate index, or any associated paths to be imported.

Access method services matches each entryname you specify through the OBJECTS parameter 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.

entryname
specifies the name of the data component, index component, cluster, alternate index, or path for which attributes are being specified. The entryname must appear on the portable data set; otherwise, the parameter list will be 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 or an alternate index.

This parameter is used when the components are defined as unique and 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.

KEYRANGES((lowkey highkey**)[(**lowkey highkey**)...])**
specifies portions of a key-sequenced cluster or alternate index to be placed on separate volumes.

The data is divided, by key, among the volumes as contained in the portable data set or as specified in the VOLUMES parameter.

If a volume serial number is duplicated on the portable data set or in VOLUMES, multiple key ranges of an alternate index or cluster attribute are placed on that volume.

If the number of volumes is greater than the number of key ranges, the excess volumes are used for overflow records from any key range without consideration of range boundaries. If there are fewer volumes than key ranges, the excess key ranges are placed on the last volume specified.

By using the KEYRANGE parameter, you may specify key range pairs for a key-sequenced data set that previously was not divided by key or that was divided in different key range groups.

The maximum number of key range pairs is 20. Key ranges must be in ascending order, and may not overlap. Gaps may exist within a specified set of ranges, but records cannot be inserted within a gap.

When you specify KEYRANGES, the entryname can be either the cluster or alternate index name or the name of the data component. If you specify KEYRANGES with the cluster or alternate index name, the values specified are defined for the data component. If you specify KEYRANGES with the data component name, the values are defined for the data component, and any specification of KEYRANGES with the cluster or alternate index name is overridden.

Keys can contain 1 to 64 characters; if coded in hexadecimal, they can contain 1 to 128 hexadecimal characters. All EBCDIC characters are allowed.

lowkey
: specifies the low key of the key range. If lowkey is shorter than the actual keys, it will be padded with binary zeros.

highkey
: specifies the high key of the key range. If highkey is shorter than the actual keys, it will be padded with binary ones.

Abbreviation: KRNG

NEWNAME(newname**)**
: specifies the new name of an imported cluster or alternate index or its components, or an associated path. When you specify NEWNAME, only the name specified as entryname is changed.

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.

Abbreviation: NEWNM

ORDERED|UNORDERED
: specifies whether volumes are to be used in the order in which they were listed when the data set was originally defined (as contained on the portable data set) or in the VOLUMES parameter.

If the data set is divided into key ranges, all the records within the range specified by the first low-key high-key pair are placed on the first volume; all the records within the second range are placed on the second volume, and so on.

If it is impossible to allocate volumes in the given order and ORDERED is specified, the command is terminated.

When you specify ORDERED|UNORDERED, 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 ORDERED|UNORDERED is specified with the cluster or alternate index name, the specified attribute is defined for the data component. For a key-sequenced cluster or alternate index, the specified attribute is also defined for the index component.
- If ORDERED|UNORDERED is specified with the data component name, the specified attribute is defined for the data component. Any specification of ORDERED|UNORDERED with the cluster or alternate index name is overridden.
- For a key-sequenced cluster and for an alternate index, if ORDERED|UNORDERED is specified with the index component name, the specified attribute is defined for the index component. Any specification of ORDERED|UNORDERED with the cluster or alternate index name is overridden.

If OBJECTS is specified and neither ORDERED or UNORDERED is specified, then UNORDERED is the default.

Abbreviations: ORD and UNORD

VOLUMES(volser[volser...]**)**
specifies the volumes on which the cluster or alternate index is to reside.

If VOLUMES is not coded, the original volume is the receiving volume.

If the portable data set was created with an EXPORT command using the PERMANENT option on a Release 1 version of access method services (DOS/VS Release 29, OS/VS1 Release 2, OS/VS2 Release 1.6, or OS/VS2 Release 2), this parameter is required. Portable data sets created by Release 1 using the PERMANENT option do not contain volume information.

When you specify 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.

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 may reside on different device types, each volume of a multivolume component must be of the same type.

If the receiving volume is of a type different from that which originally contained the cluster or alternate index, the job may be terminated 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 may be imported to reside on a device type other than what 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 may fail.

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:

- Using the DEFINE command, 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 may 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.
- Using the IMPORT command, 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 or alternate index is to be deleted and replaced, regardless of the retention time specified in the TO or FOR parameter. This parameter can be used only 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.

SAVRAC
specifies that RACF data set profiles already existing 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 will cause 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 that these added attributes are acceptable.

The ownership creation group and access list will be altered by the caller of the SAVRAC option.

Caution: You should ensure that valid profiles do exist for clusters being imported when SAVRAC is specified. If this is not done, an invalid and possibly improper profile may 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.

A profile will be defined for the imported cluster if either 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.

If you import into a catalog in which there is a component with a duplicate name that is marked as having been temporarily exported, it and any associated profiles will be deleted before the portable data set is imported.

IMPORT EXAMPLES

Importing a Key-Sequenced Cluster: Example 1

In this example, a key-sequenced cluster, D40.EXAMPLE.KSDS1, that was previously exported, is imported. (See the previous EXPORT example, "Exporting a Key-Sequenced Cluster.") OUTFILE and its associated DD statement are provided to allocate the data set.

The original copy of D40.EXAMPLE.KSDS1 is replaced with the imported copy, TAPE2. Access method services finds and deletes the duplicate name, D40.EXAMPLE.KSDS1, in the catalog D27UCAT1. (A duplicate name exists, because TEMPORARY was specified when the cluster was exported.) Access method services then redefines D40.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(D40.EXAMPLE.KSDS1) -
           CATALOG(D27UCAT1/USERMRPW)
/*
```

Job control language statement:

- SOURCE DD, which describes the portable data set, TAPE2. TAPE2 resides on a magnetic tape file that will not be 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 plus 4.

The IMPORT command copies the portable data set, TAPE2, into the system and assigns it the name D40.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 IMPORT command's parameters are:

- INFILE, which points to the SOURCE DD statement. The SOURCE DD statement describes the portable file, TAPE2, to be imported.
- OUTDATASET, which 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 D27UCAT2, access method services can dynamically allocate the cluster without specifying a JOBCAT or STEPCAT DD statement.
- CATALOG, which identifies the catalog, D27UCAT1, in which the imported cluster is to be defined. The catalog's update- or higher-level password is required.

Importing an Entry-Sequenced Cluster in a VSAM Catalog: Example 2

In this example, an entry-sequenced cluster, D50.EXAMPLE.ESDS1, is imported from a portable file, TAPE1. This example is associated with EXPORT example, "Exporting an Entry-Sequenced Cluster." The cluster is defined in a different catalog than that 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(D40.EXAMPLE.ESDS3) -
          OBJECTS( -
                (D50.EXAMPLE.ESDS1 -
                   NEWNAME(D40.EXAMPLE.ESDS3) -
                   VOLUMES(VSER02) ) -
                ) -
          CATALOG(D27UCAT1/USERUPPW)
/*
```

Job control language statement:

- SOURCE DD, which describes the portable file, TAPE1. TAPE1 resides on a magnetic tape file that will not be mounted by the operator until access method services opens TAPE1 for processing.

The IMPORT command moves the contents of the portable file, TAPE1, into the system. When TAPE1 is moved, access method services reorganizes the data records. The IMPORT command's parameters are:

- INFILE, which points to the SOURCE DD statement. The SOURCE DD statement describes the data set to be imported, TAPE1.
- OUTDATASET gives the name of the data set that is being imported; the name is the renamed cluster. Because the high-level qualifier of the data set name is the name of the alias of the catalog, D27UCAT1, the data set can be dynamically allocated without specifying a JOBCAT or STEPCAT DD statement.
- OBJECTS, which changes some of the attributes for the object being imported:
  - D50.EXAMPLE.ESDS1, which identifies the entry-sequenced cluster as it is currently named on TAPE1.
  - NEWNAME, which specifies that the cluster's entryname is to be changed to D40.EXAMPLE.ESDS3.
  - VOLUMES, which identifies the new volume on which the cluster is to reside.
- CATALOG, which identifies the catalog, D27UCAT1, that is to contain the cluster's catalog entry. The catalog's update- or higher-level password is required.

IMPORTRA

The IMPORTRA command restores catalog entries from a portable data set created by the EXPORTRA command. This command is for use with **recoverable catalogs only**.

The format of the IMPORTRA command is:

```
IMPORTRA  {INFILE(ddname)|INDATASET(entryname)}
          [OUTFILE(ddname)]
          [OBJECTS(
            (entryname
              [DEVICETYPES(devtype[  devtype...])]
              [FILE(ddname)]
              [VOLUMES(volser[  volser...])])
            [(entryname...)...])]
          [SAVRAC|NOSAVRAC]

          [CATALOG(catname[/password])]
```

IMPORTRA can be abbreviated: MPRA.

IMPORTRA PARAMETERS

Required Parameters

INFILE(ddname)
names the DD statement that describes the portable data set (the data set that resulted when you issued the EXPORTRA command). If the portable data set resides on a nonlabeled tape or in a direct access data set created by DOS/VS access method services, the DCB BLKSIZE parameter must be specified on the referenced DD statement.

If you specified BLKSIZE for the portable data set that was created when you executed your EXPORTRA job, you must specify the same block size for your IMPORTRA job. (**Note:** If you did not specify a block size for EXPORTRA, a default value of 2048 was used. Consequently, if you do not specify BLKSIZE for IMPORTRA, IMPORTRA sets the block size to 2048.) All other characteristics of the portable data set are established by IMPORTRA.

Abbreviation: IFILE

INDATASET(entryname)
specifies the data set name of the portable data set (the data set that resulted when you issued the EXPORTRA command). When you specify INDATASET instead of INFILE, the portable data set is dynamically allocated.

Abbreviation: IDS

Optional Parameters

CATALOG(catname[/password])
identifies the target catalog.

If you do not include the CATALOG parameter, the JOBCAT or STEPCAT catalogs are used. If you have not specified a JOBCAT or STEPCAT catalog, the VSAM master catalog is used. If you don't specify the CATALOG parameter when the target catalog is password-protected, the system operator will be prompted to supply the correct password.

catname
names the catalog. When you specify a user catalog, you must describe and allocate the catalog with a JOBCAT or STEPCAT DD statement.

password
is the target catalog's update- or higher-level password, if the catalog is password protected. If any of the data sets being imported is an alternate index whose base cluster is password-protected, you should supply the catalog's master password. RACF: The update or higher RACF authority to the catalog is required.

Abbreviation: CAT

OBJECTS(
entryname
[**DEVICETYPES(**devtype[devtype...]**)**]
[**FILE(**ddname**)**]
[**VOLUMES(**volser[volser...]**)**]**)**
[(entryname...)...])
The OBJECTS parameter group specifies new or changed attributes for objects on the portable data set.

OBJECTS
specifies new or changed attributes for a cluster, an alternate index, or a non-VSAM data set that is being imported. By specifying the OBJECTS parameter, you may override certain attributes contained on the portable data set.

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.

Abbreviation: OBJ

entryname
specifies the object's entryname. You can specify the entryname and associated attributes for up to 255 objects.

You can specify the entryname of these types of objects only: VSAM cluster or an alternate index, non-VSAM data set, user catalog, and a VSAM object's data or index component.

DEVICETYPES(devtype[devtype...]**)**
specifies the device type on which a user catalog being imported resides or the device type(s) on which a non-VSAM data set being imported resides.

If entryname names a user catalog, you may specify only one device type. If entryname names a non-VSAM data set that resides on different device types, the device types listed in the DEVICETYPES parameter must be specified in the same order as the volume serial numbers listed in the VOLUMES parameter.

This parameter can be specified only when the entryname names a non-VSAM data set or user catalog. For a non-VSAM data set or user catalog, if DEVICETYPES is not coded, the data set or user catalog must still reside on the same device type(s) as contained in the entry in the portable data set.

Abbreviation: DEVT

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 or an alternate index.

This parameter is required when the components are defined as unique and when the data and index components reside on different device types.

When components reside on different device types, FILE must 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.

VOLUMES(volser**[** volser...**])**
specifies the volume(s) on which a cluster or alternate index is to reside, the volume on which a user catalog resides, or the volume(s) on which a non-VSAM data set resides. If entryname names a user catalog, you may specify only one volume serial number as a subparameter of VOLUMES.

For a cluster or an alternate index, if VOLUMES is not coded, the original volume is the receiving volume. For a user catalog or a non-VSAM data set, if VOLUMES is not coded, the data set or user catalog must still reside on the same volume(s) as it did in the portable data set.

When you specify VOLUMES for a cluster or alternate index, 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.

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. (See also the description of the FILE parameter.)

Although the index and data components may reside on different device types, each volume of a multivolume component must be of the same type.

Abbreviation: VOL

OUTFILE(ddname**)**
names the DD statement that contains a data set name and the volume serial number of each volume that is to contain the imported VSAM clusters and alternate indexes. You must use concatenated DD statements if the data sets are on

different device types. If the OUTFILE parameter is not supplied, the required data sets are dynamically allocated as needed.

The dsname specified with the DD statement cannot be one of the names cataloged in the target catalog (that is, the catalog that is to contain the imported catalog entries) and it cannot be one of the entrynames within the portable data set.

Catalog Administration Guide describes how this dsname is used by IMPORTRA processing. The DD statement(s) also specify the volume serial number(s), device type, DISP=OLD, and AMP='AMORG'.

Abbreviation: OFILE

SAVRAC|NOSAVRAC
specifies, for a RACF-protected object, whether existing profiles are to be used or whether new profiles are to be created.

SAVRAC|NOSAVRAC applies to all VSAM components being imported back into the system.

SAVRAC
specifies that existing RACF data set profiles are 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 that these added attributes are acceptable. SAVRAC should be specified when RACF data set profiles already exist for objects being imported from the portable data set.

The ownership creation group and access list will be altered by the caller of the SAVRAC option.

Caution: You should ensure that valid profiles do exist for all components (for example, cluster, data, index) being imported when SAVRAC is specified. If this is not done, an invalid and possibly improper profile may be saved and used inappropriately.

Remember that paths are imported with their corresponding cluster or alternate index, and the same caution applies to these entries. In particular, keep in mind that additional paths may be brought in during the import.

NOSAVRAC
specifies that a profile will be defined for the imported components if either the RACF Automatic Data Set Protection option has been specified for you or if the exported component had a RACF indication in the catalog at the time it was exported. NOSAVRAC should be specified when you wish new profiles to be created.

There may be situations in which neither the SAVRAC nor NOSAVRAC mode would be suitable for all entities restored from the catalog recovery area. In such cases, the following procedure is suggested:

- Issue two separate EXPORTRA commands, resulting in two portable data sets. One EXPORTRA will create a portable data set containing entities to which SAVRAC should apply, the other to which NOSAVRAC should apply.
- Issue two IMPORTRA commands, one specifying SAVRAC and the other NOSAVRAC.

IMPORTRA EXAMPLE

This example shows how EXPORTRA and IMPORTRA can be used to recover a VSAM catalog. The first part illustrates the use of EXPORTRA. The second part shows how IMPORTRA is used.

Recovering a VSAM Catalog: Example 1—Part 1 (EXPORTRA)

This example performs the EXPORTRA function against the VSAM master catalog, AMASTCAT, and against all the volumes owned by it. All the data sets listed in the catalog recovery areas on the catalog volume and on the other volumes owned by AMASTCAT are exported to a SAM data set on another disk volume. Using the FORCE option causes EXPORTRA to ignore time stamp mismatches between the volumes and the catalog.

```
//RECOVER  JOB   ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//CRAVOL1  DD   DISP=OLD,VOL=SER=VSER01,UNIT=DISK,
//      AMP='AMORG'
//CRAVOL2  DD   DISP=OLD,VOL=SER=VSER04,UNIT=DISK,
//      AMP='AMORG'
//PORT     DD   DSNAME=PORT,DISP=(NEW,KEEP),
//      SPACE=(CYL,(3,3)),VOL=SER=231401,
//      UNIT=DISK,DCB=BLKSIZE=8000
//SYSIN    DD   *

     EXPORTRA -
          CRA( -
            (CRAVOL1 ALL) -
            (CRAVOL2 ALL) -
             ) -
          FORCE -
          OUTFILE(PORT) -
          MASTERPW(MCATMRPW)
/*
```

Job control language statements:

- CRAVOL1 DD, which identifies and allocates the first volume whose catalog recovery area contents is to be exported.
- CRAVOL2 DD, which identifies and allocates the second volume whose catalog recovery area contents is to be exported.
- PORT DD, which identifies the sequential file that is to receive the exported data sets and catalog recovery records. The DCB BLKSIZE parameter overrides the EXPORTRA default of 2048 to improve performance.

The EXPORTRA command exports everything appearing in the catalog recovery areas of volumes VSER01 and VSER04. The EXPORTRA command's parameters are:

- CRA, which is required and identifies the catalog recovery areas and volumes from which the export is to take place. The ddnames of the DD statements for these objects must be identical to the names specified for this parameter. The ALL subparameter specifies that everything is to be exported from each catalog recovery area.
- FORCE, which specifies that timestamp mismatches are to be ignored.
- OUTFILE, which is required and identifies the sequential (SAM) non-VSAM data set that is to receive the exported information. The ddname of the DD statement for this object must be identical to the name specified with this parameter.
- MASTERPW, which specifies the master password of the master catalog. This parameter is required in order to export

information from catalog recovery areas controlled by a recoverable catalog.

Recovering a VSAM Catalog: Example 1—Part 2 (IMPORTRA)

This example imports all the data sets that were exported using EXPORTRA in the previous example. The receiving catalog is the VSAM master catalog, and the CATALOG parameter is used to supply its master password.

```
//RESTORE   JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//PORT      DD    DSNAME=PORT,DISP=OLD,UNIT=DISK,
//      VOL=SER=333333,DCB=BLKSIZE=8000
//VSAMIN    DD    DSNAME=DUMMY.NAME,DISP=OLD,UNIT=DSIK,
//      VOL=SER=VSER01,AMP='AMORG'
//          DD    VOL=SER=VSER04,UNIT=DISK,DISP=OLD
//SYSIN     DD    *

     IMPORTRA -
          INFILE(PORT) -
          OUTFILE(VSAMIN) -
          CATALOG(AMASTCAT/MCATMRPW)
/*
```

Job control language statements:

- PORT DD, which identifies and allocates the portable data set created previously by EXPORTRA. The DCB BLKSIZE parameter is included (even though it need not be, because the information is contained in the DSCB for the data set) to illustrate the fact that the block size specified in the IMPORTRA job must be the same as that specified in the EXPORTRA job that created the portable data set.
- VSAMIN DD, which identifies and allocates the volumes of the objects to be imported. This permits them to be reloaded from the portable data set. The dsname appearing on the DD statement is required. It is a dummy name that must not appear either in the catalog or among the names on the portable data set. Because the volumes are of different device types, concatenated DD statements must be used.

The IMPORTRA command imports the data sets written on the portable data set by EXPORTRA previously. The IMPORTRA command's parameters are:

- INFILE, which is required and which identifies the portable data set.
- OUTFILE, which identifies each VSAM volume involved in the import. The DD statement identified by OUTFILE also identifies the dummy data set, DUMMY.NAME.
- CATALOG, which specifies the name of the master catalog. The master password of the target catalog is required in order to import catalog information.

LISTCAT

The LISTCAT command lists catalog entries. The format of this command is:

```
LISTCAT   [ALIAS]
            [ALTERNATEINDEX]
            [CLUSTER]
            [DATA]
            [GENERATIONDATAGROUP]
            [INDEX]
            [NONVSAM]
            [PAGESPACE]
            [PATH]
            [SPACE]
            [USERCATALOG]
          [CREATION(days)]
          [ENTRIES(entry name[/password]
            [  entryname[/password]...])|
            LEVEL(level)]
          [EXPIRATION(days)]
          [NAME|
            HISTORY|
            VOLUME|
            ALLOCATION|
            ALL]
          [NOTUSABLE:]
          [OUTFILE(ddname)]

          [CATALOG(catname[/password])]
```

LISTCAT can be abbreviated: LISTC

LISTCAT PARAMETERS

Required Parameters

The LISTCAT command has no required parameters.

When the LISTCAT command is issued as a job step (that is, not through TSO) and no parameters are specified; an entire catalog is listed. See the section "Order of Catalog Search for LISTCAT" on page 18 for a description of how the catalog to be listed is selected.

Note to TSO users: When LISTCAT is invoked from a TSO terminal and no operands are specified, the prefix of the TSO user becomes the highest level of entryname qualification and only those entries with a matching highest level of qualification are listed. It is as if you specified:

```
LISTCAT LEVEL(TSO user prefix)
```

Optional Parameters

[ALIAS][ALTERNATEINDEX][CLUSTER][DATA]
[GENERATIONDATAGROUP][INDEX][NONVSAM]
[PAGESPACE][PATH][SPACE][USERCATALOG]
specifies that certain types of entries are to be listed. Only those entries whose type is specified are listed. For example, when you specify CLUSTER but not DATA or INDEX, the cluster's entry is listed and its associated data and index entries are not listed.

When you specify 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.

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

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, the generation data sets associated with the generation data group are listed.

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.

SPACE
specifies that entries for volumes containing data spaces defined in this catalog are to be listed. Candidate volumes are included. If entries are identified by entryname, SPACE can be coded only when no other entry type parameter is coded.

Abbreviation: SPC

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

CATALOG(catname[/password]**)**
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 "Order of Catalog Search for LISTCAT" on page 18 for information about the order in which catalogs are searched.

catname
is the name of the catalog.

password
specifies the read- or higher-level password of the catalog that contains entries to be listed. RACF: The read or higher RACF authority to the catalog is required. If the entries to be listed are password protected, a password must be supplied either through this parameter or through the ENTRIES parameter. If passwords are to be listed, you must specify the master password. RACF: The alter RACF authority to the catalog is required.

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, such as CLUSTER and DATA, are to be listed only if they were created the specified number of days ago or earlier.

days
specifies the number of days ago. This value can be expressed in decimal, hexadecimal, or binary. If it is expressed in either hexadecimal or binary, it must be preceded by either 'X' or 'B' and be enclosed in single quotation marks.

The maximum number that can be specified is 9999; zero indicates that all entries are to be listed.

Abbreviation: CREAT

ENTRIES(entryname[/password][entry name[/password]...])|
LEVEL(level**)**
specifies the names of entries to be listed.

Entries are only listed from an OS CVOL if the ENTRIES or LEVELS parameter is specified and:

- The CATALOG parameter is not specified.
- No JOBCAT or STEPCAT DD is specified.

- The higher-level qualifier of the name specified using ENTRIES or LEVELS is the alias of the OS CVOL.

When a LISTCAT LEVEL of a CVOL is made, a maximum of 1456 entries can be listed.

Note to TSO users: TSO will prefix the userid to the specified data set name when the ENTRIES parameter is unqualified. The userid is not prefixed when the LEVEL parameter is specified.

ENTRIES(entryname[/password] [entryname[/password]...]**)**
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. If you want data space information, you must specify the volume serial number (as the entryname) of the volume containing the data space; you must also specify SPACE and no other entry type parameters.

password
specifies a password when the entry to be listed is password protected and a password is not specified with the CATALOG parameter. The password can be any of the entry's passwords. The entry's protection attributes are listed only when you specify the entry's (or its catalog's) master password. RACF: The alter RACF authority to the catalog or entry is required.

When you don't supply a password for a password-protected entry, the operator or TSO terminal user is prompted for the entry's password. You cannot supply a password for these types of entries: non-VSAM data set, generation data group, alias, user catalog connector, or data space.

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.

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 will be listed. If ENTRIES(A.*.B) is specified, entries 1 and 2 will be listed. If LEVEL(A.*.B) is specified, entries 1, 2, 3, and 4 will be listed. If LEVEL(A) is specified, entries 1, 2, 3, 4, 5, 6, and 7 will be listed.

When using a generic name with the ENTRIES parameter, entries must have one qualifier in addition to those specified in the command.

EXPIRATION(days**)**
specifies that entries of the indicated type, such as CLUSTER and DATA, are to be listed only if they will expire in the specified number of days or earlier.

days
specifies the number of days. This value can be expressed in decimal, hexadecimal, or binary. If it is expressed in either hexadecimal or binary, it must be preceded by either 'X' or 'B' and be enclosed in single quotation marks. The maximum number that can be specified is 9999 and indicates that all entries are to be listed. Any value that exceeds the year 2000 will default to 99.999(yyddd). Zero indicates that only entries that have already expired are to be listed.

Abbreviation: EXPIR

NAME|HISTORY|VOLUME|ALLOCATION|ALL
specifies the fields to be included for each entry listed. Appendix A, "Interpreting LISTCAT Output Listings" on page 256, shows the listed information that results when you specify nothing (which defaults to NAME), HISTORY, VOLUME, ALLOCATION, and ALL.

NAME
specifies that the name and entry type of the entries are to be listed.

Some entry types are listed along 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, "Interpreting LISTCAT Output Listings" on page 256.

Note to 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 for a recoverable catalog's entries, the catalog recovery area's volume, device type, and control interval number. It can be specified for CLUSTER, DATA, INDEX, ALTERNATEINDEX, PATH, GENERATIONDATAGROUP, PAGESPACE, and NONVSAM.

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, data space (volume) entries, non-VSAM data set entries, and user catalog connector entries.

Note to 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.

NOTUSABLE
specifies that only those data and index entries with the "unusable" indicator on are to be listed. A data or index component is marked "unusable" when a system failure occurs that results in damage to the entry's cataloged information.

When the cataloged information is reset, the damaged entry and its backup copy (in the catalog recovery area) might not match when the space allocation information is compared. VSAM marks the catalog entry "unusable" until the space allocation information is corrected. See Catalog Administration Guide for more details.

Abbreviation: NUS

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 "For an Alternate Target Data Set" on page 10.

Abbreviation: OFILE

LISTCAT EXAMPLES

Listing a Key-Sequenced Cluster's Entry in a Catalog: Example 1

In this example, a key-sequenced cluster entry is listed.

```
//LISTCAT1 JOB   ...
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     LISTCAT -
          ENTRIES(D40.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 D27UCAT1; this naming convention directs the catalog search to the appropriate catalog. Its parameters are:

- ENTRIES, which identifies the entry to be listed.
- CLUSTER, which 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, which specifies that all fields of the cluster entry are to be listed.

Alter a Catalog Entry, Then List the Modified Entry: Example 2

In this example, the free space attributes for the data component (KSDATA) of cluster MYDATA are modified. Next, the cluster entry, data entry, and index entry of MYDATA are listed to determine the effect, if any, the modification has on the cluster's other attributes and specifications.

```
//LISTCAT2 JOB     ...
//JOBCAT   DD   DSNAME=D27UCAT1,DISP=SHR
//STEP1    EXEC   PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     ALTER -
          D40.KSDATA -
          FREESPACE(10 10)
     IF LASTCC = 0 -
          THEN -
          LISTCAT -
                ENTRIES(D40.MYDATA) -
                ALL
/*
```

Job control language statement:

- JOBCAT DD, which makes a catalog available for this job: D27UCAT1.

The ALTER command modifies the free space specifications of the key-sequenced VSAM cluster D40.MYDATA. The command's parameters are:

- D40.KSDATA, which is the entryname of the data component being altered. D40.KSDATA identifies the data component of a key-sequenced VSAM cluster, D40.MYDATA. In order 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, which 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 executes.

The LISTCAT command lists the cluster's entry and its data and index entries. Its parameters are:

- ENTRIES, which specifies the entryname of the object being listed. Because D40.MYDATA is a key-sequenced cluster, the cluster entry, its data entry, and its index entry are listed.
- ALL, which specifies that all fields of each entry are to be listed.

List Catalog Entries: Example 3

This example illustrates how all catalog entries with the same generic name are listed.

```
//LISTCAT3   JOB    ...
//JOBCAT     DD     DSNAME=USERCAT4,DISP=SHR
//STEP1      EXEC   PGM=IDCAMS
//SYSPRINT   DD     SYSOUT=A
//SYSIN      DD     *
     LISTCAT -
          ENTRIES(GENERIC.*.BAKER) -
          ALL
/*
```

Job control language statement:

- JOBCAT DD, which makes a catalog available for this job: USERCAT4.

The LISTCAT command lists each catalog entry with the generic name GENERIC.*.BAKER, where "*" is any 1- to 8-character simple name. Its parameters are:

- ENTRIES, which specifies the entryname of the object to be listed. Because GENERIC.*.BAKER is a generic name, more than one entry might be listed.
- ALL, which specifies that all fields of each entry are to be listed.

LISTCRA

The LISTCRA command lists or compares the contents of a given catalog recovery area (CRA). This command is for use with recoverable catalogs only.

The format of the LISTCRA command is:

LISTCRA	INFILE(ddname[ddname...]) [CATALOG(catname[/password] ddname)] [COMPARE|NOCOMPARE] [DUMP|NAME|SEQUENTIALDUMP] [MASTERPW(password)] [OUTFILE(ddname)]

LISTCRA can be abbreviated: LISTR

LISTCRA PARAMETERS

Required Parameters

INFILE(ddname[ddname...]**)**
identifies the DD statement(s) that describes the catalog recovery area's volume. You can list or compare more than one catalog recovery area. However, if COMPARE is specified, all volumes whose catalog recovery areas are listed must be owned by the same VSAM catalog.

Note: The ddname list for the INFILE parameter is limited to 9 ddnames.

Optional Parameters

CATALOG(catname[/password] ddname**)**
identifies the catalog that owns the volume(s) identified with the INFILE parameter. The CATALOG parameter is required when you specify the COMPARE option either with the DUMP or NAME output format. CATALOG cannot be specified with SEQUENTIALDUMP.

password
When the catalog is password protected, you must supply the catalog's master password. The password is required, even though the master catalog's password may have been supplied in the MASTERPW parameter. RACF: The alter RACF authority to the catalog is required.

ddname
When CATALOG is specified, you must describe and allocate the catalog to be compared with a DD statement. ddname identifies the catalog's DD statement, and is required when you specify the CATALOG parameter. You cannot specify JOBCAT or STEPCAT as the ddname. However, if the catalog being compared is not the VSAM master catalog, you must also identify the catalog by a STEPCAT or JOBCAT DD statement in addition to the DD statement referenced by ddname.

If the catalog information is entered incorrectly, the NOCOMPARE default is taken, and all the CRA records are listed.

Abbreviation: CAT

COMPARE|NOCOMPARE
specifies whether the list is to be limited to those entries in the catalog recovery area that do not match their corresponding catalog entries.

COMPARE
When you specify COMPARE, only those entries that do not match are listed. All the catalog's entries are compared, except the catalog's self-describing records. You must identify the catalog (that contains the catalog entries that are still to be compared) with the CATALOG parameter.

The COMPARE parameter cannot be specified if you specify SEQUENTIALDUMP.

Note: If you specify the COMPARE option, page space entries will always result in a mismatch. This is a normal condition and does not require any corrective action. The mismatch is caused by the OPEN indicator not being set in the page space record in the CRA, whereas it is set in the catalog.

Abbreviation: CMPR

NOCOMPARE
When you specify NOCOMPARE (or allow it to default), all the catalog recovery area's records are listed.

Abbreviation: NCMPR

See Appendix B, "Interpreting LISTCRA Output Listings" on page 301 for examples of each type of output listing.

DUMP|NAME|SEQUENTIALDUMP
specifies the amount of cataloged information to be listed.

DUMP
specifies that each listed entry is printed in its entirety in both hexadecimal and character form. Records are listed alphamerically and grouped by components.

NAME
specifies that each listed entry includes only the entry's name, its volume serial numbers, and the name and entry type of each associated entry. All listed entries are sorted alphamerically and grouped.

SEQUENTIALDUMP
specifies that each listed entry is to be printed in its entirety in both hexadecimal and character form. Records are listed in the sequence in which they appear in the catalog recovery area.

Abbreviation: SDUMP

The DUMP, NAME, SEQUENTIALDUMP, COMPARE, and NOCOMPARE options can be specified to produce five different kinds of output listing. See Appendix B, "Interpreting LISTCRA Output Listings" on page 301, for examples of each type of listing.

MASTERPW(password)
specifies the master catalog's master password. This parameter is required when the master catalog is password-protected.

Abbreviation: MRPW

OUTFILE(ddname**)**
identifies the DD statement that describes an alternate target data set. See "For an Alternate Target Data Set" on page 10 for more details about alternate target data sets.

When OUTFILE is not specified, the listing is printed on the output device described with the SYSPRINT DD statement. When OUTFILE is specified, the listing produced by LISTCRA is sent to the alternate target data set; the remainder of the access method services output is printed on the output device described with the SYSPRINT DD statement. Refer to Appendix B, "Interpreting LISTCRA Output Listings" on page 301, for an example of the use of the OUTFILE parameter.

Abbreviation: OFILE

Listing a Catalog Recovery Area: Example 1

This example lists in dump format the catalog recovery areas for those volumes owned by the VSAM master catalog, AMASTCAT, and compares those catalog recovery areas with the actual catalog records themselves.

```
//LISTAREA  JOB   ...
//STEP1     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//CRAVOL1   DD    DISP=OLD,VOL=SER=SG2001,UNIT=DISK
//CRAVOL2   DD    DISP=OLD,VOL=SER=VSER04,UNIT=DISK
//CATVOL    DD    DSNAME=AMASTCAT,DISP=OLD
//SYSIN     DD    *
     LISTCRA -
          INFILE( -
            CRAVOL1 -
            CRAVOL2) -
          MASTERPW(MCATMRPW) -
          COMPARE -
          DUMP -
          CATALOG(AMASTCAT/MCATMRPW CATVOL)
/*
```

Job control language statements:

- CRAVOL1 DD, which identifies and allocates the volume that contains the first catalog recovery area to be listed.
- CRAVOL2 DD, which identifies and allocates the volume that contains the second catalog recovery area to be listed.
- CATVOL DD, which identifies and allocates the catalog to be compared against, and makes the catalog available to LISTCRA as a data set. Because this example is listing the master catalog, no STEPCAT or JOBCAT DD statement is required. However, if the example were listing a user catalog, a STEPCAT or JOBCAT DD statement would be required in addition to the CATVOL DD statement to describe the user catalog as a catalog.

The LISTCRA command causes the catalog recovery areas on volumes SG2001 and VSER04 to be listed and their contents compared with the VSAM system catalog AMASTCAT. The LISTCRA command's parameters are:

- INFILE, which is required and specifies the catalog recovery areas to be listed by identifying the DD statement that describes each CRA's volume.
- MASTERPW, which specifies the master password of the master catalog. This parameter is required in order to open the catalog recovery areas for LISTCRA processing.
- COMPARE, which specifies that the catalog records are to be compared with their copies in the catalog recovery areas.
- DUMP, which specifies that the results of the record comparison (that is, the LISTCRA output listing) are to be printed in dump format.
- CATALOG, which is required because the COMPARE option was specified. This parameter identifies the catalog to be compared against by identifying the DD statement that describes and allocates the catalog as a data set.

PRINT

The PRINT command prints VSAM data sets, non-VSAM data sets, and catalogs. The format of this command is:

```
PRINT   {INFILE(ddname[/password])|
          INDATASET(entryname[/password])}
        [CHARACTER|DUMP|HEX]
        [FROMKEY(key)|FROMADDRESS(address)|
          FROMNUMBER(number)|SKIP(number)]
        [OUTFILE(ddname)]
        [TOKEY(key)|TOADDRESS(address)|
          TONUMBER(number)|COUNT(number)]
```

PRINT PARAMETERS

Required Parameters

INFILE(ddname[/password])|
INDATASET(entryname[/password])
identifies the data set or component to be printed. If the logical record length of a non-VSAM source data set is greater than 32760 bytes, your PRINT command will terminate with an error message.

INFILE(ddname[/password])
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[/password])
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.

password
If a VSAM data set or component is password protected, a password must be supplied. The password to be supplied is:

- The master password of the catalog if you are listing a catalog. RACF: The alter RACF authority to the catalog is required.
- The read- or higher-level password of the data set or component if the data set or component is not a catalog. RACF: The read or higher RACF authority to the data set or component is required.
- The master password of the cluster if you are listing a component of a password-protected cluster. RACF: The alter RACF authority to the cluster is required.

Passwords are applicable only to VSAM data sets and their components.

Abbreviation: IDS

Optional Parameters

CHARACTER|DUMP|HEX
specifies the format of the listing.

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 4).

Abbreviation: CHAR

```
IDCAMS  SYSTEM SERVICES                    TIME:  19:26:13     03/08/75  PAGE  6

LISTING OF DATA SET -EXAMPLE.LISTC.DATA

RBA OF RECORD - 0
ABCD000000000001ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ012345
6789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGH
IJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWZYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRST
UVWXYZ0123456789ABCDEFGHIJKLMNOPQRST....
```

Figure 4. Example of Character Format of PRINT Command

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 5).

```
IDCAMS  SYSTEM SERVICES                    TIME:  19:25:14     03/08/75  PAGE  2

LISTING OF DATA SET -EXAMPLE.LISTC

KEY OF RECORD - F0F0F0F0F0F0F0F0F0F0F0F0F1
0000   C1C2C3C4 F0F0F0F0 F0F0F0F0 F0F0F0F1   C1C2C3C4 C5C6C7C8 C9D1D2D3 D4D5D6D7   *ABCD000000000001ABCDEFGHIJKLMNOP*
0020   D8D9E2E3 E4E5E6E7 E8E9F0F1 F2F3F4F5   F6F7F8F9 C1C2C3C4 C5C6C7C8 C9D1D2D3   *QRSTUVWXYZ0123456789ABCDEFGHIJKL*
0040   D4D5D6D7 D8D9E2E3 E4E5E6E7 F8F9F0F1   F2F3F4F5 F6F7F8F9 C1C2C3C4 C5C6C7C8   *MNOPQRSTUVWXYZ0123456789ABCDEFGH*
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   *QRSTUVWXYZ0123456789ABCDEFGHIJKL*
0160   D4D5D6D7 D8D9E2E3 E4E5E6E7 E8E9F0F1\  F2F3F4F5 F6F7F8F9 C1C2C3C4 C5C6C7C8   *MNOPQRSTUVWXYZ0123456789ABCDEFGH*
0180   C9D1D2D3 D4D5D6D7 D8D9E2E3 00000000                                       *IJKLMNOPQRST....                *
```

Figure 5. Example of Dump Format of PRINT Command

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 6 on page 219).

```
IDCAMS  SYSTEM SERVICES                    TIME:  19:26:13     03/08/75  PAGE  4

LISTING OF DATA SET -EXAMPLE.LISTC.DATA

RBA OF RECORD - 0
C1C2C3C4F0F0F0F0F0F0F0F0F0F0F0F1C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8
C9D1D2D3D4D5D6D7D8D9E2D3D4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5
F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3
E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8
C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5
F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3E4E5E6E7E8E9F0F1F2F3F4F5F6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E3
E4E5E6E7E8E9F0F1F2F3F4F5f6F7F8F9C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6D7D8D9E2E300000000
```

Figure 6. Example of Hexadecimal Format of PRINT Command

**FROMKEY(key)|FROMADDRESS(address)|
FROMNUMBER(number)|SKIP(number)**
specifies the location in the data set being listed from which 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.

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 whose key matches 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 performed. 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.

Abbreviation: FKEY

FROMADDRESS(address)
specifies the relative byte address (RBA) of the first record you want listed. The RBA value must be the beginning of a logical record. If you specify this parameter for a key-sequenced data set, the listing will be in physical sequential order instead of in logical sequential order.

FROMADDRESS can be specified only for VSAM key-sequenced 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.

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.

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.

address, number
can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n'). The expression cannot be longer than one fullword (8 decimal or hexadecimal numbers, or 32 binary numbers).

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, which is identified by the ddname SYSPRINT, is the default. The target data set must meet the requirements stated in "For a Target Data Set" on page 10.

Abbreviation: OFILE

TOKEY(key**)|TOADDRESS(**address**)|**
TONUMBER(number**)|COUNT(**number**)**
specifies the location in the data set being listed at which you want the listing to stop. If no value is specified, 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. The location at which the listing is to stop must follow the location at which 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 performed.

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.

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 will be in physical sequential order instead of in logical sequential order.

TOADDRESS can be specified only for 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.

Abbreviation: TADDR

TONUMBER(number**)**
specifies the relative record number of the last record you want printed. TONUMBER can only be specified for a VSAM relative record data set.

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, number
can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n'); the expression cannot be longer than one fullword (8 decimal or hexadecimal numbers, or 32 binary numbers).

Print a Catalog: Example 1

This example shows how to print a catalog. You may find this function of the PRINT command helpful in the event of a problem with your catalog.

```
//PRINT3   JOB  ...
//JOBCAT   DD   DSNAME=USERCAT4,DISP=OLD
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *

/* PRINT THE ENTIRE CATALOG */
    PRINT -
      INDATASET(USERCAT4)
/*
```

Job control language statement:

- JOBCAT DD, which is required and which describes and allocates the user catalog to be printed.

The PRINT command prints the entire catalog, because there are no delimiting parameters specified.

Print a Catalog: Example 2

This example shows various ways to print a catalog. You may find this function of the PRINT command helpful in the event of a problem with your catalog.

```
//PRINT3   JOB  ...
//JOBCAT   DD   DSNAME=D27UCAT1,DISP=OLD
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *

/* PRINT THE LOW KEY RANGE OF THE CATALOG */
    PRINT -
      INDATASET(D27UCAT1) -
      TOKEY (X'3F')

/* PRINT THE HIGH KEY RANGE OF THE CATALOG */
     PRINT -
       INDATASET(D27UCAT1) -
       FROMKEY (X'40')

/* PRINT A SPECIFIC RECORD */
/* FROM THE LOW KEY RANGE */
    PRINT -
      INDATASET(D27UCAT1) -
      FROMKEY (X'0000000E') -
      COUNT (1)

/* PRINT THE ENTIRE CATALOG */
    PRINT -
      INDATASET(D27UCAT1)
/*
```

The job control language statements are:

- JOBCAT DD, which is required and which describes and allocates the user catalog to be printed.

The first PRINT command prints the low-key range of the catalog. Its parameters are:

- INDATASET, which specifies the name of the catalog to be printed.

- TOKEY, which specifies that printing is to terminate after reaching a record whose key is greater than X'3F' (in its first byte). The low-key range of a VSAM catalog contains records with a range of keys having a value from X'00' to X'3F' in the first byte.

The second PRINT command prints the high-key range of the catalog. Its parameters are:

- INDATASET, which specifies the name of the catalog to be printed.
- FROMKEY, which specifies that printing is to begin at the first record whose key is X'40' or greater (in its first byte). The high-key range of a VSAM catalog contains records with a range of keys having a value from X'40' to X'FF' in the first byte.

The third PRINT command prints one catalog record from the low-key range of the catalog. Its parameters are:

- INDATASET, which specifies the name of the catalog to be printed.
- FROMKEY, which specifies that printing is to begin with the record whose key is X'0000000E' in the first 4 bytes. Records in the low-key range of a VSAM catalog have the number of the containing control interval as the first 4 bytes of their key. Records are 505 bytes in length and each record is contained in a 512-byte control interval.
- COUNT, which specifies that only one record is to be printed.

The fourth PRINT command prints the entire catalog, because there are no delimiting parameters specified.

Print a Key-Sequenced Cluster's Data Records in a Catalog: Example 3

In this example, the data records of a key-sequenced cluster, D40.EXAMPLE.KSDS1, are printed in dump format. That is, each character of the record is printed in its hexadecimal and alphameric forms.

```
//PRINT1   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     PRINT -
          INDATASET(D40.EXAMPLE.KSDS1)
/*
```

The PRINT command prints data records of the key-sequenced cluster, D40.EXAMPLE.KSDS1. Its parameter is:

- INDATASET, which names the data set to be printed. Because the data set is cataloged in D27UCAT1 and the high-level qualifier of the data set is the name of the alias of D27UCAT1, access method services can dynamically allocate the cluster without the need of JOBCAT or STEPCAT DD statements.

Because neither FROMADDRESS, FROMKEY, SKIP, TOKEY, TOADDRESS, or COUNT is specified, access method services assumes that all the cluster's data records are to be printed.

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 7 on page 224.

```
KEY OF RECORD - 00F0F0F0F0F1C9E240C4C1405CC6C9
0000  00F0F0F0 F0F1C9E2 40C4C140 5CC6C9D3   C540C9F0 C6F8F05C 40F5F040 D9C5C3D6 *.00001IS DA *FILE I0D80* 50 RECO*
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 7. An Example of the Printed Record in DUMP Format (Result of Print Command)

Copy Records from a Non-VSAM Data Set into an Entry-Sequenced VSAM Cluster, Then Print the Records: Example 4

In this example, the first 15 records from a non-VSAM data set, EXAMPLE.NONVSAM, are copied into an entry-sequenced cluster, D50.EXAMPLE.ESDS1. If the records were copied correctly, the cluster's records are printed in hexadecimal format. Finally, even if the records were not copied correctly, the non-VSAM data set's first 15 records are printed in character format.

```
//PRINT2   JOB  ...
//JOBCAT   DD   DSNAME=USERCAT4,DISP=SHR
//         DD   DSNAME=D27UCAT2,DISP=SHR
//STEP1    EXEC PGM=IDCAMS
//VSDSET2  DD   DSNAME=D50.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 statements:

- JOBCAT DD, which makes two catalogs available for this job: USERCAT4 and D27UCAT2. Concatenated JOBCAT DD statements were used to identify both catalogs.
- VSDSET2 DD, which identifies the entry-sequenced VSAM cluster, D50.EXAMPLE.ESDS1, that the records are copied into.

  Note: If the AMP=(BUFND=n) parameter was specified, performance would improve when the data set's records are accessed. BUFND was allowed to default in this example, 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, D50.EXAMPLE.ESDS1. Its parameters are:

- INDATASET, which identifies the source data set, EXAMPLE.NONVSAM. Because a JOBCAT DD statement is included with the job, VSAM will search for the catalog entry describing the non-VSAM data set in either USERCAT4 or D27UCAT2 user catalog, or in the master catalog.

- OUTFILE, which points to the VSDSET2 DD statement. The VSDSET2 DD statement identifies the output data set, D50.EXAMPLE.ESDS1. VSAM assumes that D50.EXAMPLE.ESDS1 is cataloged either in USERCAT4 or D27UCAT2 user catalog, or in the master catalog.
- COUNT, which specifies that 15 records are to be copied. Because the SKIP parameter was not specified, access method services assumes that 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 executes.

The first PRINT command prints the records in the entry-sequenced cluster, D50.EXAMPLE.ESDS1. Its parameters are:

- INFILE, which points to the VSDSET2 DD statement. The VSDSET2 DD statement identifies the cluster, D50.EXAMPLE.ESDS1. VSAM assumes that the cluster is cataloged either in the USERCAT4 or the D27UCAT2 user catalog, or in the master catalog.
- HEX, which specifies that each record is to be printed as a group of hexadecimal characters. An example of the printed record is shown in Figure 8.

```
RBA OF RECORD - 0
C2D6E8C5D9404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040404040
4040404040404040404040404040404040404040
```

Figure 8. An Example of the Printed Record in Hexadecimal (Result of PRINT Command)

Note: 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 execute only when the IF statement is satisfied, enclose the commands in a DO...END command sequence.

The second PRINT command, which executes even if the REPRO command failed, prints the first 15 records of the non-VSAM data set, EXAMPLE.NONVSAM. Its parameters are:

- INDATASET, which identifies the non-VSAM data set EXAMPLE.NONVSAM. VSAM will search for the catalog entry in either USERCAT4 or D27UCAT2 user catalog or in the master catalog.
- COUNT, which specifies that 15 records are to be printed. Because SKIP was not specified, access method services assumes that the first 15 records are to be printed.
- CHARACTER, which specifies that each record is to be printed as a group of alphameric characters. An example of the printed record is shown in Figure 9.

```
RECORD SEQUENCE NUMBER - 3
CLARK
```

Figure 9. An Example of a Printed Alphameric Character Record (Result of PRINT Command)

REPRO

The REPRO command copies VSAM and non-VSAM data sets, copies catalogs, and unloads and reloads VSAM catalogs. The format of this command is:

```
REPRO    {INFILE(ddname[/password]
           [  ENVIRONMENT(DUMMY)])|
             INDATASET(entryname[/password]
               [  ENVIRONMENT(DUMMY)])}
           {OUTFILE(ddname[/password])|
            OUTDATASET(entryname[/password])}
             [SYSTEMKEYNAME(key name)])]
           [FROMKEY(key)|FROMADDRESS(address)|
             FROMNUMBER(number)|SKIP(count)]
           [REPLACE|NOREPLACE]
           [REUSE|NOREUSE]
           [TOKEY(key)|TOADDRESS(address)|
             TONUMBER(number)|COUNT(count)]

           [ENCIPHER
              ({EXTERNALKEYNAME(key name)|
                INTERNALKEYNAME(key name)|PRIVATEKEY}
              [CIPHERUNIT(number|1)]
              [DATAKEYFILE(ddname)|DATAKEYVALUE(value)]
              [SHIPKEYNAMES(key name[  key name...])]
              [STOREDATAKEY|NOSTOREDATAKEY]
              [STOREKEYNAME(key name)]
              [USERDATA(value)])]
           [DECIPHER
              ({DATAKEYFILE(ddname)|
                DATAKEYVALUE(value)|SYSTEMKEY]
              [SYSTEMDATAKEY(value)]
```

Note: The parameters ENCIPHER and DECIPHER apply only with the IBM Programmed Cryptographic Facility (5740-XY5) or the IBM Cryptographic Unit Support (5740-XY6). (For more information, see VSAM Administration Guide.)

REPRO PARAMETERS

Required Parameters

INFILE(ddname[/password][ENVIRONMENT(DUMMY)])|
INDATASET(entryname[/password][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 terminate with an error message.

INFILE(ddname[/password])
specifies the name of the DD statement that identifies the data set to be copied.

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[/password])
specifies the name of the entry to be copied. 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 entryname.

Abbreviation: IDS

password
is the read- or higher-level password of the data to be copied. If the data set is password-protected, the read password must be supplied. If a catalog is to be copied, the master password is required. Passwords are applicable only to VSAM data sets.

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 will be ignored during the copy operation and will not be copied. (See VSAM Administration Guide for further information.)

Abbreviations: ENV and DUM

OUTFILE(ddname[/password])|OUTDATASET(entryname[/password])
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 terminate with an error message.

OUTFILE(ddname[/password])
specifies the name of a DD statement that identifies the target data set.

For VSAM data sets, the data set name can be that of a path.

If the DD statement identifies a SYSOUT data set, the attributes must match those specified in "For a Target Data Set" on page 10.

Abbreviation: OFILE

OUTDATASET(entryname[/password])
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.

password
specifies the update- or higher-level password for a password-protected target data set or path.

Abbreviation: ODS

Optional Parameters

FROMKEY(key)|FROMADDRESS(address)|
FROMNUMBER(number)|SKIP(number)
specifies the location in the source data set from which 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.

The only parameter that can be used for a SAM data set is SKIP.

If you are copying a catalog, none of these parameters can be specified; the entire catalog must be copied.

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 (that is, a portion of the key followed by *). If you specify generic keys, copying begins at the first record whose key matches that portion of the key you specified.

You cannot specify a key longer than that defined for the data set. If you do, the data set is not copied. If the specified key is not found, the next higher key is used as the starting point for copying.

FROMKEY may be specified only when an alternate index, a key-sequenced VSAM data set, or an indexed-sequential (ISAM) non-VSAM data set is being copied.

key
can contain 1 to 255 EBCDIC characters.

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 will be 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.

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.

Abbreviation: FNUM

SKIP(number)
specifies the number of logical records you want to skip before beginning to copy records. For example, if you want to copy beginning with record number 500, you specify SKIP(499).

SKIP should not be specified when you access the data set through a path; the results are unpredictable.

address, number
can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form; the expression cannot be longer than one fullword (8 decimal or hexadecimal numbers, or 32 binary numbers).

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.

Because the catalog reload function automatically performs a REPLACE function, REPLACE|NOREPLACE is ignored if the target data set is a VSAM catalog.

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. When this is the case, the source record replaces the target record.

REPLACE
When a key-sequenced data set, other than a catalog, is copied, each source record whose key matches 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 whose relative record number 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.

Abbreviation: NREP

REUSE|NOREUSE
specifies whether 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 regardless of 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.

If REUSE is specified and the data set was originally defined with the NOREUSE option, the data set must be empty; otherwise, the REPRO command terminates with an error message.

Abbreviation: RUS

NOREUSE
specifies that records are to be written at the end of an entry-sequenced data set when OUTFILE or OUTDATASET identifies a nonempty data set.

Abbreviation: NRUS

TOKEY(key)|TOADDRESS(address)|
TONUMBER(number)|COUNT(number)
specifies the location in the data set being copied at which copying is to end. The location at which the copying is to end must follow the location at which it is to begin. If no value is coded, the copying ends with the logical end of the data set or component. You can use only one of the four choices.

The only parameter that can be specified for a SAM data set is COUNT.

If you are copying a catalog, none of these parameters can be specified; the entire catalog 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 (that is, a portion of the key followed by *). If you specify generic keys, copying stops after the first record encountered that matches your key specifications.

You cannot specify a key longer than that defined for the data set. If you do, the data set is not copied. If the specified key is not found, the next lower key is used as the end point for copying.

TOKEY can be specified only when an alternate index, a key-sequenced data set, or an indexed-sequential (ISAM) non-VSAM data set is being copied.

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 key-sequenced data set, the records will be copied in physical sequential order instead of in logical sequential order.

TOADDRESS can be used only 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.

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.

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; the results are unpredictable.

address, number
address, number, and count can be expressed in decimal (n), hexadecimal (X'n'), or binary (B'n') form; the expression cannot be longer than one fullword (8 decimal or hexadecimal numbers, or 32 binary numbers).

CRYPTOGRAPHIC PARAMETERS

ENCIPHER
specifies that the target data set is to contain an enciphered copy of the source data set.

The ENCIPHER parameter indicates that the source data set is to be enciphered as it is copied to the target data set.

Abbreviation: ENCPHR

EXTERNALKEYNAME(key name**) |**
INTERNALKEYNAME(key name**) |PRIVATEKEY**
specifies whether keys are managed privately by you, or managed by the IBM Programmed Cryptographic Facility (5740-XY5) or the IBM Cryptographic Unit Support (5740-XY6).

EXTERNALKEYNAME(key name**)**
specifies that keys are to be managed by the IBM Programmed Cryptographic Facility or the IBM Cryptographic Unit Support (5740-XY6) and supplies the 1- to 8-character key name of the external file key to be used to encipher the data encrypting key. The key is only known by the system deciphering it. The key name and its corresponding enciphered data encrypting key will only be listed in SYSPRINT if NOSTOREDATAKEY is specified.

Abbreviation: EKN

INTERNALKEYNAME(key name**)**
specifies that keys are to be managed by the IBM Programmed Cryptographic Facility or the IBM Cryptographic Unit Support, and supplies the 1- to 8-character key name of the internal file key to be used to encipher the data encrypting key. The key is retained by the system creating the key. 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 performed across logical record boundaries) and also improve your performance. However, there is a corresponding increase in main storage requirements. The remaining records in the data set, after the last

complete group of multiple records, will be enciphered as a group. (If number is 5 and there are 22 records in that data set, the last 2 records will be enciphered as a unit.)

The value for number may range from 1 to 255.

Abbreviation: CHPRUN

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 only valid 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 will be 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 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 may contain 1 to 8 EBCDIC characters or 1 to 16 hexidecimal 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

SHIPKEYNAME(key name [key name...])
supplies the 1- to 8-character key name of one or more external file key(s) to be used to encipher the data encrypting key. Each key name and its corresponding enciphered data encrypting key will be listed in SYSPRINT, but will not be 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 only valid 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 will be that identified by INTERNALKEYNAME or EXTERNALKEYNAME. This parameter is only valid 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.

Note: A data encrypting key enciphered under the keys identified by SHIPKEYNAME cannot be stored in the header. Therefore, you may want to avoid using STOREDATAKEY and SHIPKEYNAME together, because this could result in storing header information that is unusable at some of the locations involved.

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 key name and its corresponding enciphered data encrypting key will be listed in SYSPRINT.

Abbreviation: NSTRDK

STOREKEYNAME(key name**)**
specifies whether a key name for the key used to encipher the data encrypting key is to be stored in the target data set header. The key name specified must be the name by which the key is known on the system on which the REPRO DECIPHER is to be performed. This key name must be the same one that was specified in INTERNALKEYNAME if REPRO DECIPHER is to be run on the same system, but can be a different key name than that specified in INTERNALKEYNAME or EXTERNALKEYNAME if REPRO DECIPHER is to be run on a different system.

This parameter is only valid when INTERNALKEYNAME or EXTERNALKEYNAME is specified. If the key name is stored in the data set header, it does not have to be supplied by the user when the data is deciphered.

Note: key name values identified by the SHIPKEYNAME parameter cannot be stored in the header. Therefore, you may want to avoid using STOREKEYNAME and SHIPKEYNAME together, because this could result in storing header information that is unusable at some of the locations involved.

Abbreviation: STRKN

USERDATA(value**)**
specifies 1 to 32 characters of user data to be placed in the target data set header. This information could be used, for example, to identify the security classification of the data.

value may 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 target data set is to contain a plaintext (deciphered) copy of the enciphered source data set.

The DECIPHER parameter indicates 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 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 keys are to be managed privately by you, or managed by the IBM Programmed Cryptographic Facility or the IBM Cryptographic Unit Support.

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. For a discussion of how the method of supplying the data encrypting key relates to the security of the key, see VSAM Administration Guide.

value can contain 1 to 8 EBCDIC characters. value 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 fewer than 8 characters.

Abbreviation: DKV

SYSTEMKEY
specifies that keys are to be managed by the IBM Programmed Cryptographic Facility or the IBM Cryptographic Unit Support.

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 only valid if SYSTEMKEY is specified. If SYSTEMDATAKEY is not specifed, REPRO will obtain the enciphered system data encrypting key from the source data set header, in which case, STOREDATAKEY must have been specified when the data set was enciphered.

value may contain 1 to 8 EBCDIC characters. 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. 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(key name**)**
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 will obtain the key name of the internal key from the source data set header, in which case, STOREKEYNAME must have been specified when the data set was enciphered.

Abbreviation: SYSKN

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, D40.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(D40.EXAMPLE.KSDS1)
/*
```

Job control language statement:

- INDSET1 DD, which 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, D40.EXAMPLE.KSDS1. Its parameters are:

- INFILE, which points to the INDSET1 DD statement. The INDSET1 DD statement identifies the source data set: ISAMDSET. ISAMDSET is an indexed-sequential data set.
- OUTDATASET, which 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. Catalog D27UCAT1 will be searched for the data set, because D40 is an alias for D27UCAT1.

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.D27V, a sequential data set, into a key-sequenced VSAM data set, D40.MYDATA. Next, records are copied from the key-sequenced data set, D40.MYDATA, into an entry-sequenced data set, ENTRY.

```
//REPRO2   JOB  ...
//JOBCAT   DD   DSNAME=USERCAT3,DISP=SHR
//STEP1    EXEC PGM=IDCAMS
//INPUT    DD   DSNAME=SEQ.D27V,DISP=SHR,DCB=(BUFNO=6)
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
          INFILE(INPUT) -
          OUTDATASET(D40.MYDATA)
/*
//STEP2    EXEC PGM=IDCAMS
//INPUT    DD   DSNAME=D40.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.D27V, into a key-sequenced data set, D40.MYDATA. STEP1's job control language statements:

- JOBCAT DD, which makes the catalog USERCAT3 available for this job.
- INPUT DD, which identifies the sequential data set, SEQ.D27V, 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.D27V, to the target data set, D40.MYDATA. Its parameters are:

- INFILE, which points to the INPUT DD statement. The INPUT DD statement identifies the source data set.
- OUTDATASET, which 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.

STEP2: Access method services copies some of the records of the key-sequenced data set D40.MYDATA into another entry-sequenced data set, ENTRY. STEP2's job control language statements:

- JOBCAT DD, which applies to this job step as well as to STEP1.
- INPUT DD, which identifies the key-sequenced cluster, D40.MYDATA, that contains the source records.
- OUTPUT DD, which 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, D40.MYDATA, to the target data set, ENTRY. Only those records with key values from DEAN to, and including, JOHNSON are copied. The REPRO command's parameters are:

- INFILE, which points to the INPUT DD statement. The INPUT DD statement identifies the source key-sequenced data set.

- OUTFILE, which points to the OUTPUT DD statement. The OUTPUT DD statement identifies the entry-sequenced data set into which the source records are to be copied.
- FROMKEY and TOKEY, which specify the lower and upper key boundaries. Only those records with key values from DEAN to, and including, JOHNSON are copied.

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.

Copy a Catalog: Example 3

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) -
               MASTERPW(UCATMPW) -
               UDATEPW(UCATUPW) -
               FOR(365) -
               VOLUME(VSER06) ) -
               CYLINDERS(100 10) ) -
           DATA -
               (CYLINDERS(20 10) ) -
           INDEX -
               (CYLINDERS(10) ) -
           CATALOG -
                (AMASTCAT/MRCATPW2)
/*
//STEP2    EXEC  PGM=IDCAMS
//STEPCAT  DD    DSNAME=COPYUCAT,DISP=OLD
//         DD    DSNAME=MYCAT,DISP=OLD
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     REPRO -
          INDATASET(MYCAT) -
          OUTDATASET(COPYUCAT/UCATMPW)
     EXPORT -
          MYCAT -
          DISCONNECT
/*
//STEP3    EXEC  PGM=IDCAMS
//STEPCAT  DD    DSNAME=COPYUCAT,DISP=SHR
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     LISTCAT
/*
//STEP4    EXEC  PGM=IDCAMS
//STEPCAT  DD    DSNAME=COPYUCAT,DISP=OLD
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DELETE -
          MYCAT -
          CLUSTER -
          PURGE -
          CATALOG(COPYUCAT/UCATMPW)
/*
//STEP5    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//SYSIN    DD    *
     DEFINE ALIAS -
          (NAME(MYCAT) -
          RELATE(COPYUCAT) )
/*
```

STEP1: A user catalog, COPYUCAT, is defined on volume VSER06. The catalog entries in MYCAT will be copied into COPYUCAT.

The DEFINE USERCATALOG command defines the user catalog, COPYUCAT. Its parameters are:

- NAME, which specifies the name of the new catalog, COPYUCAT.
- MASTERPW and UPDATEPW, which specify the master and update level passwords for the catalog.
- FOR, which specifies that the catalog is to be retained for 365 days.

- VOLUME, which specifies that the catalog is to reside on volume VSER06.
- CYLINDERS, which specifies that the catalog's data space is initially to be 100 cylinders. When the data space is extended, it is to be extended in increments of 10 cylinders. Part of the data space contains the catalog, COPYUCAT. The rest of the data space is available for suballocated VSAM data sets.
- DATA(CYLINDERS) and INDEX(CYLINDERS) specify that the catalog itself is initially to occupy 30 cylinders. When the catalog's data component is extended, it is to be extended in increments of 10 cylinders.
- CATALOG, which specifies that a user catalog connector entry is to be built and written into the AMASTCAT catalog. The user catalog connector entry points to COPYUCAT.

STEP2: Access method services copies the contents of MYCAT into COPYUCAT.

STEP2's job control statement:

- STEPCAT DD, which makes two catalogs available for this job step: COPYUCAT and MYCAT. The DD statements that identify the catalogs are concatenated.

The REPRO command copies all records from MYCAT into COPYUCAT. Access method services treats each catalog as a key-sequenced data set and copies each catalog record. Consequently, the first 13 catalog records of MYCAT, which describe MYCAT as a key-sequenced data set, are also copied into COPYUCAT. Entries from MYCAT are written into COPYUCAT beginning with catalog record 14 (that is, after the 13 self-describing records of COPYUCAT). The REPRO command's parameters are:

- INDATASET, which identifies the source data set: MYCAT. VSAM assumes that MYCAT is cataloged in either MYCAT or COPYUCAT.
- OUTDATASET, which identifies the receiving data set: COPYUCAT. VSAM assumes that COPYUCAT is cataloged in either COPYUCAT or MYCAT.

The EXPORT command removes MYCAT's user catalog connector entry from the master catalog. MYCAT's cataloged objects are now not available to the system. (STEP5 builds an alias entry that relates MYCAT to COPYUCAT, making the cataloged objects available to the system again.)

STEP3: Access method services 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.

STEP4: Access method services removes the entries in COPYUCAT that describe MYCAT as a key-sequenced data set. (See STEP2 description above.) The DELETE command's parameters are:

- MYCAT, which identifies the object to be deleted.
- CLUSTER, which specifies that MYCAT is cataloged as a cluster in COPYUCAT. MYCAT's cluster, data, and index entry are removed from COPYUCAT.
- PURGE, which specifies that MYCAT is to be deleted regardless of a specified retention period or date.

- CATALOG, which specifies that MYCAT is cataloged in COPYUCAT. COPYUCAT's master password, UCATMPW, is required to delete its catalog entries.

STEP5: Access method services builds an alias entry that relates MYCAT to COPYUCAT. Because no CATALOG parameter or JOBCAT or STEPCAT DD statement identifies the catalog that is to contain the alias entry, VSAM assumes the entry is to be written into the master catalog.

When MYCAT was defined, it was defined on volume VSER04. Its catalog entries describe VSAM objects on that volume. COPYUCAT is defined on volume VSER06. Because MYCAT was copied into COPYUCAT (and MYCAT no longer exists as a user catalog), COPYUCAT entries now describe VSAM objects on volumes VSER04 and VSER06.

Unload a VSAM User Catalog: Example 4

This example shows how a VSAM user catalog can be unloaded to tape using the catalog unload/reload feature of the REPRO command.

```
//UNLOAD   JOB   ...
//JOBCAT   DD    DSNAME=D27UCAT2,DISP=OLD
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//CATIN    DD    DSNAME=D27UCAT2,DISP=OLD
//CATOUT   DD    DSNAME=PORTABLE.TAPE,LABEL=(1,SL),
//     DISP=NEW,UNIT=TAPE,VOL=SER=TAPE01,
//     DCB=(DEN=3,RECFM=VB,LRECL=516,BLKSIZE=5164)
//SYSIN    DD    *
     REPRO -
          INFILE(CATIN/USERMRPW) -
          OUTFILE(CATOUT)
/*
```

Job control language statements:

- JOBCAT DD, which is required, and which identifies and allocates the user catalog as the job catalog.
- CATIN DD, which describes and allocates the user catalog, D27UCAT2, as a VSAM data set to be opened and used by the REPRO command as the source data set for the unload operation.
- CATOUT DD, which describes and allocates a magnetic tape file to contain the copy of the catalog. The DCB parameters must be specified as shown. The BLKSIZE parameter can be any multiple of LRECL plus 4, where LRECL=516.

The REPRO command causes the VSAM user catalog D27UCAT2 to be unloaded (that is, copied) to a magnetic tape file. The REPRO command's parameters are:

- INFILE, which is required, and which identifies the user catalog as a source VSAM data set. The catalog's master password is required to open it as a data set.
- OUTFILE, which is required, and which describes the magnetic tape file, or target data set that is to contain the catalog's copy.

Reload an Unloaded VSAM User Catalog: Example 5

This example shows how a VSAM user catalog can be reloaded from the backup copy, created in the previous example, using the catalog unload/reload feature of the REPRO command.

```
//RELOAD   JOB   ...
//JOBCAT   DD    DSNAME=D27UCAT2,DISP=OLD
//STEP1    EXEC  PGM=IDCAMS
//SYSPRINT DD    SYSOUT=A
//CATIN    DD    DSNAME=PORTABLE.TAPE,LABEL=(1,SL),
//     DISP=OLD,UNIT=TAPE,VOL=SER=TAPE01,DCB=DEN=3
//CATOUT   DD    DSNAME=D27UCAT2,DISP=OLD
//SYSIN    DD    *
     REPRO -
          INFILE(CATIN) -
          OUTFILE(CATOUT/USERMRPW)
/*
```

Job control language statements:

- JOBCAT DD, which is required and which describes and allocates the user catalog to be reloaded. The JOBCAT statement specifies that D27UCAT2 is the job catalog.
- CATIN DD, which describes and allocates the magnetic tape file that contains the unloaded copy of the catalog. Because a standard label tape is used, DCB parameters of record format, record size, and block size need not be specified. For a nonlabeled tape, the same parameters as in the previous example must be specified.
- CATOUT DD, which describes the user catalog as a VSAM data set to be opened and used by REPRO as the target data set for the reload operation.

The REPRO command causes a VSAM user catalog to be reloaded from the backup copy created in the previous example. The REPRO command's parameters are:

- INFILE, which is required, and which identifies the magnetic tape file that contains the unloaded, or backup, copy of the user catalog.
- OUTFILE, which is required and which identifies the user catalog as a VSAM data set to be opened for record processing. The master password allows opening the catalog as a data set.

Encipher Using System Keys: Example 6

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 to be deciphered at a remote installation. Four source records at a time will be enciphered as a unit. The keys will be managed by the IBM Programmed Cryptographic Facility or the IBM Cryptographic Unit Support.

```
//ENSYS    JOB    ...
//JOBCAT   DD   DSN=D27UCAT,DISP=SHR
//STEP1    EXEC PGM=IDCAMS
//CLEAR    DD   DSN=RRDS1,DISP=SHR
//CRYPT    DD   DSN=RRDSEN,LABEL=(1,SL),DISP=NEW
                UNIT=TAPE,VOL=SER=TAPE01,
                DCB=(DEN=3,RECFM=FB,LRECL=516,BLKSIZE=5160)
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
          INFILE(CLEAR/RDPW) -
          OUTFILE(CRYPT) -
          COUNT(50) -
          ENCIPHER -
           (EXTERNALKEYNAME(AKEY27) -
            STOREDATAKEY -
            CIPHERUNIT(4) -
            USERDATA(CONF))
/*
```

Job control language statements:

- JOBCAT DD makes the catalog D27UCAT available.
- 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 will be enciphered in units of four records, except for the last two records, which will be enciphered together. The enciphered data encrypting key will be 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 REPRO command are:

- INFILE points to the CLEAR DD statement, identifying the source data set to be enciphered, RRDS1. RRDS1 is cataloged in either D27UCAT or the master catalog. The read password for the source data set is RDPW.
- OUTFILE points to the CRYPT DD statement, identifying the target data set on tape.
- COUNT indicates the copy operation is to end after 50 records have been 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 7

In this example, the enciphered data set produced by the job in Example 6 is deciphered at the remote location, using a VSAM relative record data set as the target for the plaintext (deciphered) data. The empty slots in the original data set will be reestablished. Keys will be managed by the IBM Programmed Cryptographic Facility or the IBM Cryptographic Support Unit.

```
//DESYS    JOB    ...
//JOBCAT   DD   DSN=D27UCATR,DISP=SHR
//STEP2    EXEC PGM=IDCAMS
//CRYPT    DD   DSN=RRDSEN,LABEL=(1,SL),DISP=OLD,
                UNIT=TAPE,VOL=SER=TAPE01,
                DCB=DEN=3
//CLEAR    DD   DSN=RRDS2,DISP=SHR
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
          INFILE(CRYPT) -
          OUTFILE(CLEAR/UPPW2) -
          DECIPHER -
           (SYSTEMKEY -
            SYSTEMKEYNAME(BKEY27))
/*
```

Job control language statements:

- JOBCAT DD makes the catalog D27UCATR available.
- 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. The internal file key (BKEY27) is used to decipher the enciphered data encrypting key, which 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. RRDS2 is cataloged in either D27UCATR or the master catalog. The defined record size must be the same as that of the original relative record data set. The update password for the target data set is UPPW2.
- 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 to be managed by the IBM Program Cryptographic Facility or the IBM Cryptographic Unit Support.
- 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 8

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 will be enciphered separately, using a data encrypting key supplied by the user via a data encrypting key data set. Keys will be managed privately by the user.

```
//ENPRI    JOB    ...
//JOBCAT   DD   DSN=D27UCAT,DISP=SHR
//STEP1    EXEC PGM=IDCAMS
//CLEAR    DD   DSN=SAMDS1,DISP=OLD,
                VOL=SER=VOL005,UNIT=DISK
//CRYPT    DD   DSN=ESDS1,DISP=OLD
//KEYIN    DD   *
          53467568503A7C29
/*
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     REPRO -
           INFILE(CLEAR) -
           OUTFILE(CRYPT/UPPW) -
           REUSE -
           ENCIPHER -
            (PRIVATEKEY -
             DATAKEYFILE(KEYIN))
/*
```

Job control language statements:

- JOBCAT DD makes the catalog D27UCAT available.
- 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 will not be 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. ESDS1 is cataloged in either D27UCAT or the master catalog. 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 terminate.
- 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, which supplies the plaintext data encrypting key, X'53467568503A7C29', to be used to encipher the data.

Decipher Using Private Keys: Example 9

In this example, the enciphered data set produced by the job in Example 8, is deciphered at the same location, using an entry-sequenced data set as the target for the plaintext (deciphered) data. Keys will be managed privately by the user.

```
//DEPRI    JOB    ...
//JOBCAT   DD   DSN=D27UCAT,DISP=SHR
//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/RDPW1) -
          OUTFILE(CLEAR/UPPW3) -
          DECIPHER -
           (DATAKEYVALUE(X'53467568503A7C29') )
/*
```

Job control language statements:

- JOBCAT DD makes the catalog D27UCAT available.
- 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 to be enciphered, ESDS1. ESDS1 is cataloged in either D27UCAT or the master catalog. The read password for the source data set is RDPW1.
- OUTFILE points to the CLEAR DD statement, identifying the target data set, ESDS3. ESDS3 is cataloged in either D27UCAT or the master catalog, and 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. The update password for the target data set is UPPW3.
- 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', which was used to encipher the data.

RESETCAT

The RESETCAT command compares catalog entries with catalog recovery area (CRA) entries in order to regain access to catalog data. This command is for use with recoverable catalogs only.

The format of the RESETCAT command is:

```
RESETCAT  CATALOG(catname[/password][ ddname])
          {CRAFILES((ddname1[ ALL| NONE])
            [(ddname1[ ALL| NONE])...])|
              CRAVOLUMES((volser[ devtype])
              [(volser[ devtype])...])}
          [IGNORE|NOIGNORE]
          [MASTERPW(password)]
          [WORKCAT(catname[/password])]
          [WORKFILE(ddname[/password])]
```

RESETCAT may be abbreviated: RCAT.

RESETCAT PARAMETERS

Required Parameters

CATALOG(catname[/password][ddname])
identifies the catalog to be reset. The catalog specified by catname must have the RECOVERABLE attribute.

Exclusive control of the catalog is required throughout the duration of the command. Also, no VSAM data sets cataloged in the catalog may be open. The master catalog may not be reset while it is in use as a master catalog.

password
specifies the catalog's master password. If the catalog is password protected, you must specify the master password of the catalog. RACF: The alter RACF authority to the catalog is required.

ddname
specifies the name of the DD statement for the catalog being reset. If ddname is not specified, the catalog will be dynamically allocated. ddname must also be specified in a JOBCAT or STEPCAT.

Abbreviation: CAT

CRAFILES((ddname1[ALL| NONE])...)
specifies a list of DD statements that identify the volumes and devices to be used to reset the catalog. Each volume contains a CRA that has a copy of the catalog records describing the VSAM objects on that volume. ALL specifies that the catalog records in the CRA on the volume specified by ddname1 are to be reset.

If a volume contains a multivolume data set whose catalog entry is being reset, other volumes of that multivolume data set may be needed during the reset operation; you should specify these via additional ddname1 parameters. Use the NONE subparameter if you do not want the entries in these additional volumes to be used to reset the catalog.

If a volume is needed and you do not specify the DD statements provided by CRAFILES, it will be allocated dynamically, and will assume the NONE attribute.

CRAVOLUMES((volser[devtype])...)
specifies the volume serial numbers and corresponding device types of the volumes used in resetting the catalog.

Each volume contains a catalog recovery area (CRA) that has a copy of the catalog records that describe the objects on the volume. The volume is allocated dynamically.

volser
specifies the volume serial number used in resetting the catalog.

devtype
specifies the device type for the associated volser. If you do not specify devtype, the volume must be mounted and on a unit marked permanently resident or reserved. If you specify a generic name for devtype, the volume need not be mounted.

If a volume contains a multivolume data set whose catalog entry is being reset, other volumes of that multivolume data set may be needed during the reset operation; if needed and not specified for reset, these volumes will be allocated dynamically. Dynamically allocated volumes assume the NONE attribute as described for the CRAFILES parameter.

Abbreviation: CRAVOL

Optional Parameters

IGNORE|NOIGNORE
specifies whether RESETCAT should continue processing and force the reset when certain errors are encountered. These errors may be:

- I/O error in the catalog.
- I/O error in the CRA.
- The volume record or its extensions in the CRA are in error.

IGNORE
specifies that the RESETCAT command will try to recover as much information as possible and reset the catalog's description of those data sets on the specified volumes.

For example, if IGNORE is specified and an I/O error occurs while processing a record from a CRA, that record will be ignored. In the process, data sets that could not be recovered on a particular volume would be marked unusable on that volume, and the space that was assigned to them would be freed.

Abbreviation: IGN

NOIGNORE
specifies that I/O errors result in immediate termination.

Abbreviation: NIGN

MASTERPW(password)
specifies the master catalog's master password when the master catalog is password protected.

WORKCAT(catname[/password])
specifies the name of the catalog in which RESETCAT defines the WORKFILE. This catalog must not be the same as the catalog being reset. The catalog must be specified in a JOBCAT or STEPCAT DD statement.

If you do not specify the WORKCAT parameter, the catalog will be chosen using existing rules for "Order of Catalog Selection for DEFINE" on page 16.

If you do not specify the WORKCAT parameter, the catalog must be the first DD statement in the JOBCAT or STEPCAT DD statement concatenation. However, if the work file catalog is the master catalog, then you must specify the WORKCAT parameter and it must be the last DD statement in the JOBCAT or STEPCAT DD statement concatenation. The resultant catalog must not be the same as the one being reset.

password
: If the catalog is password protected, the update- or higher-level password is required. RACF: The update or higher RACF authority to the catalog is required.

Abbreviation: WCAT

WORKFILE(ddname**[/**password**])**
: specifies the name of a DD statement that identifies a VSAM data set name and a list of volume serial numbers of volumes containing VSAM data spaces and units to be used by RESETCAT for a work file. These volumes must be owned by the catalog in which the work file is defined.

  This data set will be defined by RESETCAT and used as temporary storage while processing the command; it will be deleted at the end of the command. The data set must not be already defined.

  Space requirements for the data set are noted in "WORKFILE Space Requirements" in Catalog Administration Guide. If you do not specify WORKFILE, IDCUT1 is used as a default ddname.

  password
  : If you require one, you may specify a password to be used by RESETCAT to password protect the work file. Because the work file will contain a copy of the catalog records, it may contain security-sensitive information. The password you specify will become the work file's master password. You may specify the password only if ddname is explicitly specified.

  Abbreviation: WFILE

Resetting a Catalog: Example 1

This example illustrates the use of RESETCAT to reset a catalog (USERCAT1), one of whose owned volumes (VSER01) has been restored. The other volume owned by USERCAT1 is VSER02.

```
//RECOVER  JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//STEPCAT  DD   DSN=USERCAT1,DISP=OLD
//         DD   DSN=AMASTCAT,DISP=OLD
//IDCUT1   DD   DSN=WORKFILE,DISP=OLD,
//          VOL=SER=333333,UNIT=DISK,AMP='AMORG'
//CRAVOL1  DD   DISP=OLD,VOL=SER=VSER01,UNIT=DISK
//CRAVOL2  DD   DISP=OLD,VOL=SER=VSER02,UNIT=DISK
//DDCAT    DD   DSN=USERCAT1,DISP=OLD
//SYSIN    DD   *
    RESETCAT -
      CATALOG(USERCAT1/MASTER DDCAT) -
      CRAFILES((CRAVOL1 ALL) (CRAVOL2 NONE)) -
      WORKCAT(AMASTCAT/MCATMRPW) -
      MASTERPW(MCATMRPW) -
      IGNORE
/*
```

Job control language statements:

- STEPCAT DD, which makes the user catalog, USERCAT1, and the master catalog, AMASTCAT, available for the step. These catalogs must be available for the catalog to be reset and for the work file to be defined, respectively. Note that, when the work file is to be defined in the master catalog, the master catalog must be last in the concatenation and the catalog name is supplied via the WORKCAT parameter.
- IDCUT1 DD, which identifies the data set to be used for the work file.
- CRAVOL1 DD, which identifies and allocates a volume whose catalog recovery area is to be used to reset the catalog.
- CRAVOL2 DD, which identifies and allocates a volume whose catalog recovery area may change as a result of resetting the catalog recovery area on volume VSER01. This volume is not reset in the catalog.
- DDCAT DD, which allocates the catalog to be reset.

The RESETCAT command causes catalog USERCAT1 to be reset from the catalog recovery area on VSER01. The RESETCAT command's parameters are:

- CATALOG, which identifies the catalog to be reset. The master password of the catalog is required to update the catalog. The DD statement for the catalog is also identified.
- CRAFILES, which specifies a list of DD statements that identify the volumes to be used to reset the catalog. ALL specifies that the catalog records in the CRA on the volume specified by the CRAVOL1 DD statement are to be reset (in the catalog). NONE allows the CRAVOL2 DD statement to be specified for a CRA volume that is not to be used for reset, but that may be needed as a result of the reset (for example, a multivolume data set resides on a reset volume and on a nonreset volume).
- WORKCAT, which specifies the name of the catalog in which to define the work file. WORKCAT must be specified if the work file is to be defined in the master catalog.

- IGNORE, which specifies that RESETCAT should force the reset of the catalog despite certain errors that may be encountered during the RESETCAT processing.

Resetting a Catalog: Example 2

This example illustrates the use of RESETCAT to reset a catalog (USERCAT1) after the volume (VSER00) on which the catalog resides has been restored. The volumes owned by USERCAT1, VSER01, and VSER02 have been determined to be out of synchronization with the catalog by running a LISTCRA with the COMPARE option for all volumes owned by USERCAT1.

```
//RECOVER  JOB  ...
//JOBCAT   DD   DSN=USERCAT2,DISP=OLD
//         DD   DSN=USERCAT1,DISP=OLD
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//WORKF    DD   DSN=WORKFILE,DISP=OLD,UNIT=(DISK,2),
//          VOL=SER=(333333,333334),AMP='AMORG'
//SYSIN    DD   *
    RESETCAT -
     CATALOG(USERCAT1) -
     CRAVOLS((VSER01 3330-1)(VSER02 3330-1)) -
     WORKFILE(WORKF/WRKPSW) -
     NOIGNORE
/*
```

Job control language statements:

- JOBCAT DD, which makes the user catalogs USERCAT2 and USERCAT1 available for the job. These catalogs must be available for the work file to be defined and for the catalog to be reset, respectively.
- WORKF DD, which identifies the data set to be used for the work file.

  Note: All other required volumes will be dynamically allocated by RESETCAT.

The RESETCAT command causes catalog USERCAT1 to be reset from the catalog recovery areas on VSER01 and VSER02. The RESETCAT parameters are:

- CATALOG, which identifies the catalog to be reset.
- CRAVOLS, which specifies a list of volumes and their device types to be used to reset the catalog.
- WORKFILE, which specifies the DD statement for the work file and a password to be used as the work file's master password.
- NOIGNORE, which specifies that RESETCAT should not force the reset of the catalog if certain errors are encountered.

Resetting a Catalog: Example 3

This three part example illustrates the use of RESETCAT to reset a catalog (USERCAT1 on volume VSER00) that must be reallocated on a different volume (VSER03). Such a situation may occur when irreparable physical damage has occurred to the catalog. In this example, the catalog recovery area for the damaged catalog volume is assumed to be accessible and valid and thus included for reset in the newly defined catalog. Note that the newly defined catalog bears the same name as the damaged catalog. The other volumes owned by USERCAT1 are VSER01 and VSER02. The catalog is first disconnected, then redefined.

```
//DISC      JOB  ...
//STEP1     EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//STEPCAT  DD   DSN=USERCAT1,DISP=OLD
//SYSIN    DD   *
    EXPORT USERCAT1 DISCONNECT

//STEP2    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//UCATDD   DD   VOL=SER=VSER03,UNIT=DISK,DISP=OLD
//CRAVOL1  DD   VOL=SER=VSER01,UNIT=DISK,DISP=OLD
//CRAVOL2  DD   VOL=SER=VSER02,UNIT=DISK,DISP=OLD
//SYSIN    DD   *
    DEFINE UCAT -
      (NAME (USERCAT1) FILE(UCATDD) VOL(VSER03) -
      RECOVERABLE CYL(20 1)) -
      DATA(CYL(10  1)) INDEX(CYL(1))

//STEP3    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//STEPCAT  DD   DSN=USERCAT1,DISP=OLD
//         DD   DSN=AMASTCAT,DISP=OLD
//IDCUT1   DD   DSN=WORKFILE,DISP=OLD,UNIT=DISK,
//         VOL=SER=333333,AMP='AMORG'
//SYSIN    DD   *
    RESETCAT -
        CATALOG(USERCAT1) -
        WORKCAT(AMASTCAT) -
        CRAVOLS((VSER00 3330)(VSER01 3330-1)(VSER02 3330-1))
/*
```

STEP1: The user catalog, USERCAT1, is disconnected from the system by using the EXPORT command.

STEP2: The user catalog, USERCAT1, is redefined on the volume VSER03.

STEP3: The user catalog USERCAT1 is reset from the previous catalog's owned volumes.

Job control language statements:

- STEPCAT DD, which makes the user catalog, USERCAT1, and the master catalog, AMASTCAT, available for the step. These catalogs must be available for the catalog to be reset, and for the work file to be defined, respectively.
- IDCUT1 DD, which identifies the data set to be used for the work file.

The RESETCAT command causes the redefined catalog USERCAT1 to be reset from the catalog recovery areas on volumes VSER00, VSER01, and VSER02. The RESETCAT command's parameters are:

- CATALOG, which identifies the catalog to be reset.
- WORKCAT, which specifies the name of the catalog in which to define the work file. WORKCAT must be specified if the work file is to be defined in the master catalog.
- CRAVOLS, which specifies a list of volumes and their device types to be used to reset the catalog.

VERIFY

The VERIFY command causes a catalog to correctly reflect the end of a data set after an error has occurred while closing a VSAM data set. The error may have caused the catalog to be incorrect.

The format of the VERIFY command is:

VERIFY	{FILE(ddname[/password])| DATASET(entryname[/password])}

VERIFY can be abbreviated: VFY

VERIFY PARAMETERS

Required Parameters

FILE(ddname**[/**password**])**
ddname specifies the name of a DD statement that identifies the cluster or alternate index to be verified. For further information, see "Using VERIFY to Synchronize Values" in VSAM Administration Guide.

password
is the control or master password of a password-protected object (RACF: The control or alter RACF authority to the object is required), or is the master password of the object's password-protected catalog (RACF: The alter RACF authority to the catalog is required).

DATASET(entryname**[/**password**])**
specifies the name and password of the object to be verified. If DATASET is coded, the object is dynamically allocated.

password
is the control or master password of a password-protected object (RACF: The control or alter RACF authority to the object is required), or is the master password of the object's password-protected catalog (RACF: The alter RACF authority to the catalog is required).

Abbreviation: DS

The VERIFY command can be used following a system failure that caused a component opened for update processing to be improperly closed. It can also be used to verify an entry-sequenced data set defined with REUSE that was open in create mode when the system failure occurred.

Note: When sharing data sets between different processors, it is recommended that you run VERIFY as the first step of a job stream to prevent job termination caused by an open ACB error code, if the other processor already has the data set open.

VERIFY EXAMPLE

Upgrading a Data Set's End-of-File Information: Example

When a data set that was improperly closed (that is, closed as a result of system failure) is opened, the VSAM OPEN routines set a return code to indicate that the data set's cataloged information might not be accurate. The user 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[1] and issuing the VERIFY command:

```
//VERIFY   JOB  ...
//JOBCAT   DD   DSNAME=D27UCAT1,DISP=SHR
//FIXEOD   EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
             LISTCAT   ENTRIES(FAROUT) -
                       ALL
             VERIFY    DATASET(FAROUT)
             LISTCAT   ENTRIES(FAROUT) -
                       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 closed properly.

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 show the point at which processing stopped due to system failure. This information should help the user determine how much data was added correctly before the system failed.

Note: VERIFY will only update the high-used RBA fields for the data set, and not any record counts.

See Appendix A, "Interpreting LISTCAT Output Listings" on page 256, for details on the order in which catalog records are to be listed and the meanings of the listed records.

---

[1] When the data set is opened (first OPEN after system failure), 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 failure. Subsequently, when the data set is next opened, its EOD and EOKR information might still be erroneous (until VERIFY is issued to correct it), but VSAM OPEN sets the "data set opened correctly" return code.

APPENDIX A. 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 when you specify certain options. It also lists and describes fields that can be printed for each type of catalog entry.

Each listed entry is identified by its type (that is, cluster, non-VSAM, data, and so forth) and by its entryname. Entries are listed in alphabetic order of the entrynames, except when the ENTRIES parameter is used. The entries are then listed in the order in which 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. If type options (CLUSTER, DATA, SPACE, etc.) have been specified or a generic entryname list was specified, this excludes the associated entry.

This appendix is organized in three parts:

- "LISTCAT Output Keywords" lists all field names that can be listed for each type of entry.
- "Description of Keyword Fields" describes each field name within a group of related field names.
- "Examples of LISTCAT Output Listings" 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
DSP	data space group
GDG	generation data group base entry, special fields
NVS	non-VSAM entry, special field
HIS	history group

Abbreviations	Group Names
PRT	protection group
STA	statistics group
VLS	volumes group
VOL	volume entry, special fields

Alias Entry Keywords

```
ASSOCIATIONS (ASN)
entryname (HIS)
HISTORY (HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (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)
  EXPIRATION (HIS)
  OWNER-IDENT(HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (HIS)
  RELEASE (HIS)
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)
  EXPIRATION (HIS)
  OWNER-IDENT (HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (HIS)
  RELEASE (HIS)
INDEX (ASN)
MASTERPW (PRT)
PATH (ASN)
PROTECTION (PRT)
```

RACF (PRT)
READPW (PRT)
UPDATEPW (PRT)
USAR (PRT)
USVR (PRT)

Data Entry Keywords

ALLOCATION (ALC)
AIX (ASN)
ASSOCIATIONS (ASN)
ATTEMPTS (PRT)
ATTRIBUTES (ATT)
AVGLRECL (ATT)
AXRKP (ATT)
BIND (ATT)
BUFSPACE (ATT)
BYTES/TRACK (VLS)
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)
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-BYTES (STA)
HIGH-KEY (VLS)
HI-KEY-RBA (VLS)
HISTORY (HIS)
  CREATION (HIS)
  EXPIRATION (HIS)
  OWNER-IDENT (HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (HIS)
  RELEASE (HIS)
IMBED (ATT)
INDEX (ASN)
INH-UPDATE (ATT)
INDEXED (ATT)
KEYLEN (ATT)
LOW-KEY (VLS)
MASTERPW (PRT)
MAXLRECL (ATT)
MAXRECS (ATT)
NOERASE (ATT)
NOIMBED (ATT)
NONINDEXED (ATT)
NOREPLICAT (ATT)
NOREUSE (ATT)
NONSPANNED (ATT)
NOSWAP (ATT)
NOTRKOVFL (ATT)
NONUNIQKEY (ATT)
NOTUSABLE (ATT)
NOWRITECHK (ATT)
NUMBERED (ATT)

ORDERED (ATT)
PGSPC (ASN)
PHYRECS/TRK (VLS)
PHYREC/SIZE (VLS)
PROTECTION (PRT)
RACF (PRT)
HI-ALLOC-RBA (ALC)
HI-USED-RBA (ALC)
HI-ALLOC-RBA (VLS)
HI-USED-RBA (VLS)
READPW (PRT)
RECOVERY (ATT)
REC-DELETED (STA)
REC-INSERTED (STA)
REC-RETRIEVED (STA)
REC-TOTAL (STA)
REC-UPDATED (STA)
RECORDS/CI (ATT)
REPLICATE (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)
SUBALLOC (ATT)
SWAP (ATT)
SYSTEM-TIMESTAMP (STA)
TEMP-EXP (ATT)
TRACKS/CA (VLS)
TRKOVFL (ATT)
UNORDERED (ATT)
UNIQUE (ATT)
UNIQUEKEY (ATT)
UPDATEPW (PRT)
USAR (PRT)
USVR (PRT)
VOLFLAG (VLS)
VOLSER (VLS)
VOLUMES (VLS)
WRITECHECK (ATT)

Index Entry Keywords

ALLOCATION (ALC)
AIX (ASN)
ASSOCIATIONS (ASN)
ATTEMPTS (PRT)
ATTRIBUTES (ATT)
AVGLRECL (ATT)
BIND (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)
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-BYTES (STA)
HIGH-KEY (VLS)
HISTORY (HIS)
  CREATION (HIS)
  EXPIRATION (HIS)
  OWNER-IDENT (HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (HIS)
  RELEASE (HIS)
IMBED (ATT)
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)
NOIMBED (ATT)
NOREPLICAT (ATT)
NOREUSE (ATT)
EXCPS (STA)
EXTENTS (STA)
NOTUSABLE (ATT)
NOWRITECHK (ATT)
ORDERED (ATT)
PHYRECS/TRK (VLS)
PHYREC-SIZE (VLS)
PROTECTION (PRT)
RACF (PRT)
HI-ALLOC-RBA (ALC)
HI-USED-RBA (ALC)
HI-ALLOC-RBA (VLS)
HI-USED-RBA (VLS)
RECOVERY (ATT)
READPW (PRT)
REC-DELETED (STA)
REC-INSERTED (STA)
REC-RETRIEVED (STA)
REC-TOTAL (STA)
REC-UPDATED (STA)
REPLICATE (ATT)
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)
SUBALLOC (ATT)
SYSTEM-TIMESTAMP (STA)
TEMP-EXP (ATT)
TRACKS/CA (VLS)
UNORDERED (ATT)
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)
  EXPIRATION (HIS)
  OWNER-IDENT (HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (HIS)
  RELEASE (HIS)
NONVSAM (ASN)
```

Non-VSAM Entry Keywords

```
ALIAS (ASN)
ASSOCIATIONS (ASN)
DEVTYPE(VLS)
entryname (HIS)
FSEQN (NVS)
HISTORY (HIS)
  CREATION (HIS)
  EXPIRATION (HIS)
  OWNER-IDENT (HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (HIS)
  RELEASE (HIS)
VOLSER(VLS)
```

Page Space Entry Keywords

```
ASSOCIATIONS (ASN)
ATTEMPTS (PRT)
CODE (PRT)
CONTROLPW (PRT)
DATA (ASN)
entryname (HIS)
HISTORY (HIS)
  CREATION (HIS)
  EXPIRATION (HIS)
  OWNER-IDENT (HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (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)
  EXPIRATION (HIS)
  OWNER-IDENT (HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (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)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (HIS)
  RELEASE (HIS)
VOLFLAG (VLS)
VOLSER (VLS)

Volume Entry Keywords

ATTRIBUTES (DSP)
  AUTOMATIC (DSP)
  EXPLICIT (DSP)
  MASTERCAT (DSP)
  SUBALLOC (DSP)
  UNIQUE (DSP)
  USERCAT (DSP)
BYTES/TRK (VOL)
CHARACTERISTICS (VOL)
CYLS/VOL (VOL)
DATASET-DIRECTORY (DSP)
  ATTRIBUTES (DSP)
  DSN (DSP)
  EXTENTS (DSP)
DATASETS (DSP)
DATASETS-ON-VOL (VOL)
DATASPACE (DSP)
DATASPCS-ON-VOL (VOL)
DEVTYPE (VOL)
EXTENT-DESCRIPTOR (DSP)
  BEG-CCHH (DSP)
  SPACE-MAP (DSP)
  TRACKS-TOTAL (DSP)
  TRACKS-USED (DSP)
EXTENTS (DSP)
FORMAT-1-DSCB (DSP)

```
  CCHHR (DSP)
  TIMESTAMP (DSP)
MAX-PHYREC-SZ (VOL)
MAX-EXT/ALLOC (VOL)
HISTORY (HIS)
  RCVY-CI (HIS)
  RCVY-DEVT (HIS)
  RCVY-VOL (HIS)
  RELEASE (HIS)
SEC ALLOC (DSP)
TRKS/CYL (VOL)
TYPE (DSP)
  CYLINDER (DSP)
  TRACK (DSP)
volume serial number (HIS)
VOLUME-TIMESTAMP (VOL)
```

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
DSP	data space group
GDG	generation data group base entry, special fields
NVS	non-VSAM entry, special field
HIS	history group
PRT	protection group
STA	statistics group
VLS	volumes group
VOL	volume entry, special fields

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-ALLOC-RBA—The highest RBA (plus 1) available within allocated space to store data.

HI-USED-RBA—The highest RBA (plus 1) within allocated space that actually contains data.

SPACE-PRI—Gives the number of units (indicated under TYPE) of space allocated to the data or index component when the cluster (that is, its data or index component) was defined. This amount of space is to be allocated whenever a data component (or key range within it, and its 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

TRACK—Tracks

ASN: ASSOCIATIONS GROUP

This group lists the type, such as cluster or data, and entrynames 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.
  - 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.

AVGLRECL—The average length of data records. AVGLRECL equals MAXLRECL when the records are fixed length.

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.

BIND—The component is staged from mass storage to a direct access storage staging drive when it is opened, and it is retained (bound) in direct access storage until it is closed.

BUFSPACE—The minimum buffer space in virtual storage to be provided by a processing program.

CI/CA—The number of control intervals per control area.

CISIZE—The size of a control interval.

CYLFAULT—The component is not staged from mass storage to a direct access storage staging drive when it is opened, but data from it is staged as the data is needed.

DSTGWAIT—The component is destaged from direct access storage to mass storage before VSAM returns control to the program that is closing it.

ERASE—Records are to be erased (set to binary zeros) when deleted.

EXCPEXIT—The name of the object's exception exit routine.

IMBED—The sequence-set index record is stored along with its associated data control area.

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.

INDEXED—The data component has an index; it is key sequenced.

KEYLEN—The length of the key field in a data record.

MAXLRECL—The maximum length of data or index records. AVGLRECL equals MAXLRECL when the records are fixed length.

MAXRECS—Identifies the highest possible valid relative record number, for a relative record data set. This value is calculated as follows:

2^{32}/CISIZE x number of records slots per control interval.

NOERASE—Records are not to be erased (set to binary 0's) when deleted.

NOIMBED—The sequence-set index record is not stored along with its associated data control area.

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.

NOREPLICAT—Index records are not replicated.

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(s) in the entry's volume list.

NOWRITECHK—Write operations are not checked for correctness.

NUMBERED—The cluster is a relative record data set.

ORDERED—Volumes are used for space allocation in the order they were specified when the cluster was defined.

RECOVERY—A temporary CLOSE is issued as each control area of the data set is loaded, so the whole data set won't have to be reloaded if a serious error occurs during loading.

RECORDS/CI—Specifies the number of records, or slots, in each control interval of a relative record data set.

RECVABLE—This attribute is set in the data entry of a recoverable catalog, and each of the catalog's volumes contains a catalog recovery area.

REPLICATE—Index records are replicated (that is, each is duplicated around a track of the index's direct access device).

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.

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.

SUBALLOC—More than one VSAM cluster can share the data space. A VSAM catalog might also occupy the data space.

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.

UNIQUE—Only one VSAM cluster or catalog can occupy the data space—the cluster or catalog is unique.

UNIQUEKEY—Indicates, for an alternate index, that the alternate-key value identifies one, and only one, data record in the base cluster.

UNORDERED—Volumes specified when the cluster was defined can be used for space allocation in any order.

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 will be updated to reflect any changes to the base cluster's contents.

WRITECHECK—Write operations are checked for correctness.

DSP: DATA SPACE GROUP

The fields in this group are included by LISTCAT, as part of a volume entry, for each data space on the volume. If a volume contains no data spaces, it is a candidate volume.

ATTRIBUTES—Describes the attributes of the data space.

AUTOMATIC—The data space was created automatically by a secondary allocation operation. If a VSAM cluster must be given additional space on a volume and all existing data spaces are full, an existing data space will be extended, or a new one allocated on the volume, using catalog DADSM. A new data space so allocated is known as an implicit or automatic data space.

EXPLICIT—The data space was created explicitly by a DEFINE MASTERCATALOG or USERCATALOG command, or a DEFINE ALTERNATEINDEX or DEFINE CLUSTER command with the UNIQUE attribute specified or by a DEFINE SPACE command.

MASTERCAT—The data space contains the master catalog.

SUBALLOC—The data space might contain several VSAM clusters.

USERCAT—The data space contains a user catalog.

UNIQUE—The data space contains a single (unique) VSAM cluster or catalog.

DATASET DIRECTORY—Lists the VSAM data sets that can be stored (see CANDIDATE below) or actually are stored (in whole or in part) in the data space.

ATTRIBUTES—Describes the relation between the named data set and the data space.

CANDIDATE—The volume is a candidate for storing the data set

(NULL)—The data set is stored (in whole or in part) in the data space

DSN—The data set name of the object that can be stored or is stored on the volume.

EXTENTS—The number of data set extents for the data set within the data space.

DATASETS—The number of VSAM data sets stored (in whole or in part) in the data space. (The number includes data sets for which the volume is a candidate.)

EXTENT-DESCRIPTOR—Describes the data space extent.

BEG-CCHH—The device address (that is, CC = cylinder and HH = track) of the extent.

SPACE-MAP—A hexadecimal number that tells what tracks are used and what tracks are free in the extent. The number consists of one or more run length codes (RLCs). The first RLC gives the number of contiguous used tracks, starting at the beginning of the extent; if all the tracks in the extent are used, there is only one RLC. The second RLC gives the number of contiguous free tracks, beyond the used tracks. A third RLC gives used tracks again, a fourth gives free tracks, and so on.

A 1-byte RLC gives the number of tracks less than or equal to 250; a 3-byte RLC gives the number of tracks greater than 250. That is, if the first byte of an RLC is X'FA' (250) or less, it is the only byte of the RLC and gives the number of tracks. If the first byte of an RLC is X'FB' (251) or more,

the byte is followed by two more bytes that give the number of tracks (the first byte indicates merely to look at the next two bytes).

TRACKS-TOTAL—Allocated to the data space altogether.

TRACKS-USED—Used to store data.

EXTENTS—The number of data space extents in the data space.

FORMAT-1-DSCB—Identifies the format-1 DSCB that describes the data space.

CCHHR—The device address (that is, CC = cylinder, HH = track, and R = record number) of the DSCB in the VTOC.

TIMESTAMP—The time the data space was allocated (system time-of-day clock value). The DSCB contains the timestamp as part of the VSAM-generated name.

SEC-ALLOC—Gives the number of units (indicated under TYPE) of space to be allocated whenever the data space is extended.

TYPE—Indicates the unit of space allocation:

CYLINDERS—Cylinders

TRACK—Tracks

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 will be uncataloged when the maximum number (given under LIMIT) is reached and one more data set is to be 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 will be 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 that describes each one is removed from the VTOC of the volume on which 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, identify its catalog recovery volume and release level, 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 (yy.ddd) on which the entry was created. This field will always contain 00.000 for a non-VSAM data set entry in an OS CVOL.

EXPIRATION—The Julian date (yy.ddd) after which the entry can be deleted without specifying the PURGE parameter in the DELETE command. Julian date 99.999 indicates that PURGE is always required to delete the object. This field will always contain 00.000 for a non-VSAM data set entry in an OS CVOL.

OWNER-IDENT—The identity of the owner of the object described by the entry.

RCVY-CI—If the entry is in a recoverable catalog, this field contains the control interval number in the catalog recovery area where a duplicate of the entry can be found.

RCVY-DEVT—If the entry is in a recoverable catalog, this field identifies the recovery volume's device type. (See the Device Type Translate Table at the end of this section for a translation of the hexadecimal device type to its equivalent EBCDIC value.)

RCVY-VOL—If the entry is in a recoverable catalog, this field contains the recovery volume's serial number.

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

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 whether or not 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 attempt to enter a correct password.

CODE—Gives the code used to tell the console operator what alternate index, catalog, cluster, path, data component, or index component requires him to enter a password. 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) which 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 indicate 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 a failure occurs during CLOSE, the statistics may not be valid. Because a VSAM catalog remains open from job to job, its statistics are not updated.

Statistic values can be any value from a minus to a large number, if a failure occurs before or during close processing. VERIFY will not correct these statistics.

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-BYTES—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 i this statistic.

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 relative byte address (RBA) 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, in decimal, of the first sequence-set record. The sequence set might be separated from the index set by some quantity 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 (execute 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.

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).

VLS: VOLUMES GROUP

The fields in this group identify the volume(s) on which a data component, index component, user catalog, or non-VSAM data set is stored. It also identifies candidate volume(s) 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 describe 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 imbedded sequence set, and one for the high-level index. The extents for the imbedded sequence set will be 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. (See the Device Type Translate Table at the end of this section for a translation of the hexadecimal device type to its equivalent EBCDIC value.)

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.

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[2]—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[2]—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[2]—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.

---

[2] Multiple key ranges may reside on a single volume; the volumes group is repeated for each such key range field.

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-ALLOC-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-USED-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.

TRACKS/CA—The number of tracks in a control area in the data component. (This value is computed when the entry is defined, and 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 whether the volume is a candidate volume, the first, or a subsequent volume on which data in a given key range is stored.

CANDIDATE—The volume is a candidate for storing the data or index component.

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.

VOL: VOLUME ENTRY, SPECIAL FIELDS FOR

The special fields for a volume entry describe the characteristics of the space that VSAM uses on the volume.

volume serial number—The name of the cataloged volume entry. The volume serial number can be specified with the ENTRIES parameter of LISTCAT to identify the volume entry.

BYTES/TRK—The number of bytes that VSAM can use on each track on the volume.

CYLS/VOL—The number of cylinders that VSAM can use on the volume, including alternate track cylinders.

DATASETS-ON-VOL—The number of VSAM clusters that reside, in whole or in part, on the volume. This includes candidate volumes.

DATASPCS-ON-VOL—The number of VSAM data spaces on the volume. Should the number of data spaces be zero, then this is a candidate volume only, and the data space group is omitted.

DEVTYPE—The type of device to which the volume belongs. (See the Device Type Translate Table at the end of this section for a translation of the hexadecimal device type to its equivalent EBCDIC value.)

MAX-PHYREC-SZ—The size of the largest physical record that VSAM can write on the volume.

MAX-EXT/ALLOC—The maximum number of extents that can be allocated per each allocation request for a single data set on the volume.

TRKS/CYL—The number of tracks in each cylinder on the volume.

VOLUME-TIMESTAMP—The time (system time-of-day clock value) VSAM last changed the contents of the volume. The format-4 DSCB contains the timestamp at offset 76 (X'4C').

DEVICE TYPE TRANSLATE TABLE

LISTCAT Code	Device Type
3000 8001	9 TRK TAPE
3010 200C	3375
3010 200E	3380[3]
3040 200A	3340 (35M/70M)
3050 2006	2305-1
3050 2007	2305-2
3050 2009	3330
3050 200B	3350
3050 200D	3330-1
3058 2009	3330V
3080 8001	7 TRK TAPE
30C0 2008	2314/2319
3200 8003	9 TRK, 6250 BPI TAPE
3400 8003	9 TRK, 1600 BPI TAPE

Refer to "Data Area Description of the Unit Control Block (UCB)" in System Programming Library: Debugging Handbook, Volume 3 for additional information on device types.

---

[3] 3380 models A04, AA4, B04, AD4, BD4, AE4, and BE4.

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 receive when the LISTCAT procedure executes 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
//STEPCAT  DD    DSN=YOURCAT,DISP=SHR
//OUTDD    DD    DSN=LISTCAT.OUTPUT,UNIT=TAPE,
//         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/PASSWORD) -
        OUTFILE(OUTDD) -
        ...
/*
```

The JOB statement contains user and accounting information required for your installation.

The EXEC statement identifies the program to be executed, IDCAMS (that is, the access method services program).

- STEPCAT DD, which allocates your catalog. The catalog can be allocated dynamically if the STEPCAT and JOBCAT are omitted. If the catalog is dynamically allocated, its volume must be mounted as permanently RESIDENT or RESERVED.
- 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=2400-3 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, and so on.)
  - 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, DISP=(MOD,KEEP) or DISP=(MOD,PASS) can be used 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 'OUTDD' with '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 '/*' terminates the input data statements.

The LISTCAT command parameters shown above 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 the catalog, YOURCAT, whose entries are to be listed. If the catalog is password protected, its read- or higher-level password, PASSWORD, is required. If the passwords and protection attributes of each entry are to be listed, the catalog's master password is required.

- 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
/*
```

LISTCAT AND ACCESS METHOD SERVICES OUTPUT MESSAGES

When the LISTCAT job completes, access method services provides messages and diagnostic information. An analysis of any error messages received can be found in Message Library: System Messages. When your LISTCAT job completes successfully, access method services provides messages that follow the entry listing (see Figure 10):

```
                        LISTING FROM CATALOG -- MJKCAT

     THE NUMBER OF ENTRIES PROCESSED WAS:
               AIX ----------------------1
               ALIAS --------------------2
               CLUSTER ------------------5
               DATA ---------------------6
               GDG ----------------------1
               INDEX --------------------3
               NONVSAM ------------------2
               PAGESPACE ----------------0
               PATH ---------------------2
               SPACE --------------------0
               USERCATALOG --------------0
               TOTAL --------------------22

     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. 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

When you specify LISTCAT with no parameters, the entryname and type of each entry are listed (see Figure 11 on page 280). The same listing would result if the NAMES parameter were specified.

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.

```
  /*    LIST ENTRYNAMES OF THE CATALOG */
     LISTCAT -
          CATALOG (MJKCAT)

                              LISTING FROM CATALOG -- MJKCAT

AIX ----------- MJK.ALT.INDEX1
   DATA ------- MJK.ALT.INDEX1.DATA
   INDEX ------ MJK.ALT.INDEX1.INDEX
   PATH ------- MJK.AIX1.PATH
CLUSTER ------- MJK.CLUSTER1
   DATA ------- MJK.CLUSTER.DATA
   INDEX ------ MJK.CLUSTER1.INDEX
GDG BASE ------ MJK.GDG1
   NONVSAM ---- MJK.GDG1.G0001V00
ALIAS --------- MJK.GDG1.ALIAS
NONVSAM ------- MJK.NONVSAM1
CLUSTER ------- MJKCAT
   DATA ------- VSAM.CATALOG.BASE.DATA.RECORD
   INDEX ------ VSAM.CATALOG.BASE.INDEX.RECORD
VOLUME -------- 333001

                              LISTING FROM CATALOG -- MJKCAT

         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX --------------------1
                   ALIAS ------------------1
                   CLUSTER ----------------2
                   DATA -------------------3
                   GDG --------------------1
                   INDEX ------------------3
                   NONVSAM ----------------2
                   PAGESPACE --------------0
                   PATH -------------------1
                   SPACE ------------------0
                   USERCATALOG ------------0
                   TOTAL ------------------14

         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 Output When No Parameters Are Specified

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 12).

```
  /*    LIST VOLUMES FOR EACH ENTRY  */
     LISTCAT -
          VOLUME -
          CATALOG (MJKCAT)


                              LISTING FROM CATALOG -- MJKCAT

AIX ----------- MJK.ALT.INDEX1
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000010'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'

   DATA ------- MJK.ALT.INDEX1.DATA
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000011'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     VOLUMES
       VOLSER------------333001       DEVTYPE------X'30502009'

   INDEX ------ MJK.ALT.INDEX1.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000012'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     VOLUMES
       VOLSER------------333001       DEVTYPE------X'30502009'

   PATH ------- MJK.AIX1.PATH
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000014'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'

CLUSTER ------- MJK.CLUSTER1
     HISTORY
       OWNER-IDENT-----OWNCLUST       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'00000E'
       RELEASE----------------2       EXPIRATION--------77.140      RCVY-DEVT----X'30502009'

   DATA ------- MJK.CLUSTER.DATA
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'00000D'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     VOLUMES
       VOLSER------------333001       DEVTYPE------X'30502009'

   INDEX ------ MJK.CLUSTER1.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'00000F'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     VOLUMES
       VOLSER------------333001       DEVTYPE------X'30502009'

GDG BASE ------ MJK.GDG1
     HISTORY
       OWNER-IDENT-------OWNGDG       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000015'
       RELEASE----------------2       EXPIRATION--------99.365      RCVY-DEVT----X'30502009'

   NONVSAM ---- MJK.GDG1.G0001V00
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000016'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
    VOLUMES
       VOLSER------------333001       DEVTYPE------X'30502009'
       VOLSER------------333002       DEVTYPE------X'30502009'
```

Figure 12 (Part 1 of 2). Example of LISTCAT VOLUME Output

```
NONVSAM ---- MJK.NONVSAM1
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000018'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     VOLUMES
       VOLSER------------333001       DEVTYPE------X'30502009'
CLUSTER ------- MJKCAT
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040
       RELEASE----------------2     EXPIRATION--------99.999

   DATA ------- VSAM.CATALOG.BASE.DATA.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------00.000
       RELEASE----------------2     EXPIRATION--------00.000
     VOLUMES
       VOLSER------------333001     DEVTYPE------X'30502009'

   INDEX ------ VSAM.CATALOG.BASE.INDEX.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------00.000
       RELEASE----------------2     EXPIRATION--------00.000
     VOLUMES
       VOLSER------------333001     DEVTYPE------X'30502009'

VOLUME -------- 333001
     HISTORY
       RELEASE----------------2     RCVY-VOL----------333001     RCVY-DEVT----X'30502009'     RCVY-CI--------X'000009'
     VOLUMES
       VOLSER------------333001     DEVTYPE------X'30502009'


                          LISTING FROM CATALOG -- MJKCAT


         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX --------------------1
                   ALIAS ------------------0
                   CLUSTER ----------------2
                   DATA -------------------3
                   GDG --------------------1
                   INDEX ------------------3
                   NONVSAM ----------------2
                   PAGESPACE -------------0
                   PATH -------------------1
                   SPACE ------------------1
                   USERCATALOG ------------0
                   TOTAL ------------------14

         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 (Part 2 of 2). Example of LISTCAT VOLUME Output

LISTCAT SPACE ALL OUTPUT LISTING

When the LISTCAT command is specified with the SPACE and ALL parameters, all the information for each volume entry in the catalog is listed (see Figure 13). You can use this type of listing to determine how space on each cataloged volume is allocated to VSAM data spaces. You may have to list the volume's table of contents (VTOC) to determine how all of the volume's space is allocated.

```
  /*   LIST VOLUME INFORMATION FOR THE VOLUMES -
         CONTROLLED BY THE CATALOG  */
LISTCAT -
   SPACE -
   ALL -
   CATALOG(MJKCAT)


                          LISTING FROM CATALOG -- MJKCAT


VOLUME -------- 333001
     HISTORY
       RELEASE----------------2       RCVY-VOL----------333001      RCVY-DEVT----X'30502009'      RCVY-CI--------X'000009'
     CHARACTERISTICS
       BYTES/TRK----------13165       DEVTYPE------X'30502009'      MAX-PHYREC-SZ------13030      DATASETS-ON-VOL--------5
       TRKS/CYL-----------19          VOLUME-TIMESTAMP:             MAX-EXT/ALLOC----------5      DATASPCS-ON-VOL--------1
       CYLS/VOL-----------411              X'8A51B6A76850E000'
     DATASPACE
       DATASETS-----------5           FORMAT-1-DSCB:                ATTRIBUTES:
       EXTENTS------------1           CCHHR------X'0000000103'      SUBALLOC
       SEC-ALLOC----------40          TIMESTAMP                     EXPLICIT
       TYPE---------------TRACK            X'8A51B61D97B04000'     USERCAT
       EXTENT-DESCRIPTOR:
       TRACKS-TOTAL---------160       BEG-CCHH-----X'00000002'      SPACE-MAP----------5947
       TRACKS-USED----------89
       DATASET-DIRECTORY:
         DSN----MJKCAT                                              ATTRIBUTES--------(NULL)      EXTENTS----------------3
         DSN----MJK.CLUSTER.DATA                                    ATTRIBUTES--------(NULL)      EXTENTS----------------1
         DSN----MJK.CLUSTER1.INDEX                                  ATTRIBUTES--------(NULL)      EXTENTS----------------1
         DSN----MJK.ALT.INDEX1.DATA                                 ATTRIBUTES--------(NULL)      EXTENTS----------------1
         DSN----MJK.ALT.INDEX1.INDEX                                ATTRIBUTES--------(NULL)      EXTENTS----------------1


                          LISTING FROM CATALOG -- MJKCAT


       THE NUMBER OF ENTRIES PROCESSED WAS:
                AIX ------------------0
                ALIAS ----------------0
                CLUSTER --------------0
                DATA -----------------0
                GDG ------------------0
                INDEX ----------------0
                NONVSAM --------------0
                PAGESPACE ------------0
                PATH -----------------0
                SPACE ----------------1
                USERCATALOG ----------0
                TOTAL ----------------1

       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 13. Example of LISTCAT SPACE ALL Output

LISTCAT ALL OUTPUT LISTING

When you specify the LISTCAT command and include the ALL parameter, all the information for each catalog entry is listed. The example in Figure 14 on page 285 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 that are not volume entries. If a volume serial number is specified with the ENTRIES parameter, 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, and so forth), all entries in the catalog, including volume entries, are listed.

```
  /*    LIST ALL CATALOGED INFORMATION FOR EACH ENTRY  */
     LISTCAT -
          ALL -
          CATALOG (MJKCAT)


                              LISTING FROM CATALOG -- MJKCAT


AIX ----------- MJK.ALT.INDEX1
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000010'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     PROTECTION-PSWD-----(NULL)       RACF----------------(NO)
     ASSOCIATIONS
       DATA-----MJK.ALT.INDEX1.DATA
       INDEX----MJK.ALT.INDEX1.INDEX
       CLUSTER--MJK.CLUSTER1
       PATH-----MJK.AIX1.PATH
     ATTRIBUTES
       UPGRADE

   DATA ------- MJK.ALT.INDEX1.DATA
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000011'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     PROTECTION-PSWD-----(NULL)       RACF----------------(NO)
     ASSOCIATIONS
       AIX------MJK.ALT.INDEX1
     ATTRIBUTES
       KEYLEN-----------------5       AVGLRECL------------4086      BUFSPACE------------8704      CISIZE--------------4096
       RKP--------------------5       MAXLRECL-----------32600      EXCPEXIT----------(NULL)      CI/CA-----------------18
       AXRKP-----------------45
       SHROPTNS(1,3)   RECOVERY       SUBALLOC          NOERASE     INDEXED        NOWRITECHK     NOIMBED        NOREPLICAT
       UNORDERED         NOREUSE      SPANNED        NONUNIQKEY
     STATISTICS
       REC-TOTAL--------------4       SPLITS-CI--------------0      EXCPS------------------7
       REC-DELETED------------0       SPLITS-CA--------------0      EXTENTS----------------1
       REC-INSERTED-----------0       FREESPACE-%CI----------0      SYSTEM-TIMESTAMP:
       REC-UPDATED------------0       FREESPACE-%CA----------0           X'8A51B6C0C87CF000'
       REC-RETRIEVED----------0       FREESPC-BYTES------69632
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------6       HI-ALLOC-RBA-------73728
       SPACE-SEC--------------6       HI-USED-RBA--------73728
     VOLUME
       VOLSER-----------333001        PHYREC-SIZE---------4096      HI-ALLOC-RBA-------73728      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'       PHYRECS/TRK------------3      HI-USED-RBA--------73728      EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME       TRACKS/CA--------------6
       EXTENTS:
       LOW-CCHH-----X'00040008'       LOW-RBA----------------0      TRACKS-----------------6
       HIGH-CCHH----X'0004000D'       HIGH-RBA-----------73727
```

Figure 14 (Part 1 of 7). Example of LISTCAT ALL Output

```
   INDEX ------ MJK.ALT.INDEX1.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)        CREATION----------77.040     RCVY-VOL----------333001      RCVY-CI--------X'000012'
       RELEASE----------------2        EXPIRATION--------00.000     RCVY-DEVT----X'30502009'
     PROTECTION-PSWD-----(NULL)        RACF----------------(NO)
     ASSOCIATIONS
       AIX------MJK.ALT.INDEX1
     ATTRIBUTES
       KEYLEN-----------------5        AVGLRECL---------------0     BUFSPACE---------------0      CISIZE---------------512
       RKP--------------------5        MAXLRECL-------------505     EXCPEXIT----------(NULL)      CI/CA-----------------20
       SHROPTNS(1,3)   RECOVERY        SUBALLOC          NOERASE    NOWRITECHK        NOIMBED     NOREPLICAT      UNORDERED
       NOREUSE
     STATISTICS
       REC-TOTAL--------------1        SPLITS-CI--------------0     EXCPS------------------3      INDEX:
       REC-DELETED------------0        SPLITS-CA--------------0     EXTENTS----------------1      LEVELS-----------------1
       REC-INSERTED-----------0        FREESPACE-%CI----------0     SYSTEM-TIMESTAMP:             ENTRIES/SECT-----------4
       REC-UPDATED------------0        FREESPACE-%CA----------0          X'8A51B6C0C87CF000'      SEQ-SET-RBA------------0
       REC-RETRIEVED----------0        FREESPC-BYTES-------9728                                   HI-LEVEL-RBA-----------0
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1        HI-ALLOC-RBA-------10240
       SPACE-SEC--------------1        HI-USED-RBA----------512
     VOLUME
       VOLSER-----------333001         PHYREC-SIZE----------512     HI-ALLOC-RBA-------10240      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'        PHYRECS/TRK----------20      HI-USED-RBA----------512      EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME        TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0004000E'        LOW-RBA----------------0     TRACKS-----------------1
       HIGH-CCHH----X'0004000E'        HIGH-RBA-----------10239

   PATH ------- MJK.AIX1.PATH
     HISTORY
       OWNER-IDENT-------(NULL)        CREATION----------77.040     RCVY-VOL----------333001      RCVY-CI--------X'000014'
       RELEASE----------------2        EXPIRATION--------00.000     RCVY-DEVT----X'30502009'
     PROTECTION-PSWD-----(NULL)        RACF----------------(NO)
     ASSOCIATIONS
       AIX------MJK.ALT.INDEX1
       DATA-----MJK.ALT.INDEX1.DATA
       INDEX----MJK.ALT.INDEX1.INDEX
       DATA-----MJK.CLUSTER.DATA
       INDEX----MJK.CLUSTER1.INDEX
     ATTRIBUTES
       UPDATE

CLUSTER ------- MJK.CLUSTER1
     HISTORY
       OWNER-IDENT-----OWNCLUST        CREATION----------77.040     RCVY-VOL----------333001      RCVY-CI--------X'00000E'
       RELEASE----------------2        EXPIRATION--------77.140     RCVY-DEVT----X'30502009'
     PROTECTION
       MASTERPW----------MASTCL        UPDATEPW----------UPDCL      CODE-------------CODECL       RACF----------------(NO)
       CONTROLPW---------CNTLCL        READPW------------READCL     ATTEMPTS---------------3      USVR--------------(NULL)
       USAR--------------(NONE)
     ASSOCIATIONS
       DATA-----MJK.CLUSTER1.DATA
       INDEX----MJK.CLUSTER1.INDEX
       AIX------MJK.ALT.INDEX1
```

Figure 14 (Part 2 of 7). Example of LISTCAT ALL Output

```
   DATA ------- MJK.CLUSTER.DATA
     HISTORY
       OWNER-IDENT-------(NULL)        CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'00000D'
       RELEASE----------------2        EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     PROTECTION-PSWD-----(NULL)        RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER--MJK.CLUSTER1
     ATTRIBUTES
       KEYLEN-----------------5        AVGLRECL--------------80      BUFSPACE-----------25088      CISIZE-------------12288
       RKP--------------------5        MAXLRECL-------------100      EXCPEXIT--------EXITCLUS      CI/CA------------------2
       SHROPTNS(1,3)   RECOVERY        SUBALLOC          NOERASE     INDEXED         NOWRITECHK    NOIMBED         NOREPLICAT
       UNORDERED        NOREUSE        NONSPANNED
     STATISTICS
       REC-TOTAL-------------20        SPLITS-CI--------------0      EXCPS------------------4
       REC-DELETED-----------0         SPLITS-CA--------------0      EXTENTS----------------1
       REC-INSERTED----------0         FREESPACE-%CI---------15      SYSTEM-TIMESTAMP:
       REC-UPDATED-----------0         FREESPACE-%CA---------20           X'8A51B6ACD5B92000'
       REC-RETRIEVED---------20        FREESPC-BYTES------12288
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------2        HI-ALLOC-RBA-------24576
       SPACE-SEC--------------2        HI-USED-RBA--------24576
     VOLUME
       VOLSER-----------333001         PHYREC-SIZE---------4096      HI-ALLOC-RBA-------24576      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'        PHYRECS/TRK------------3      HI-USED-RBA--------24576      EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME        TRACKS/CA--------------2
       EXTENTS:
       LOW-CCHH-----X'00040005'        LOW-RBA----------------0      TRACKS-----------------2
       HIGH-CCHH----X'00040006'        HIGH-RBA-----------24575

   INDEX ------ MJK.CLUSTER1.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)        CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'00000F'
       RELEASE----------------2        EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     PROTECTION-PSWD-----(NULL)        RACF----------------(NO)
     ASSOCIATIONS
       CLUSTER--MJK.CLUSTER1
     ATTRIBUTES
       KEYLEN-----------------5        AVGLRECL--------------0       BUFSPACE---------------0      CISIZE---------------512.
       RKP--------------------5        MAXLRECL-------------505      EXCPEXIT--------EXITCLUS      CI/CA------------------20
       SHROPTNS(1,3)   RECOVERY        SUBALLOC          NOERASE     NOWRITECHK      NOIMBED       NOREPLICAT      UNORDERED
       NOREUSE
     STATISTICS
       REC-TOTAL-------------1         SPLITS-CI--------------0      EXCPS------------------4      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'8A51B6ACD5B92000'      SEQ-SET-RBA------------0
       REC-RETRIEVED---------0         FREESPC-BYTES-------9728                                    HI-LEVEL-RBA-----------0
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI--------------1        HI-ALLOC-RBA-------10240
       SPACE-SEC--------------1        HI-USED-RBA----------512
     VOLUME
       VOLSER-----------333001         PHYREC-SIZE----------512      HI-ALLOC-RBA-------10240      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'        PHYRECS/TRK----------20       HI-USED-RBA----------512      EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME        TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00040007'        LOW-RBA----------------0      TRACKS-----------------1
       HIGH-CCHH----X'00040007'        HIGH-RBA-----------10239

GDG BASE ------ MJK.GDG1
     HISTORY
       OWNER-IDENT-------OWNGDG        CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000015'
       RELEASE----------------2        EXPIRATION--------99.365      RCVY-DEVT----X'30502009'
     ATTRIBUTES
       LIMIT----------------250        SCRATCH             EMPTY
     ASSOCIATIONS
       NONVSAM--MJK.GDG1.G0001V00
```

Figure 14 (Part 3 of 7). Example of LISTCAT ALL Output

```
   NONVSAM ---- MJK.GDG1.G0001V00
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000016'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     VOLUMES
       VOLSER------------333001       DEVTYPE------X'30502009'      FSEQN------------------0
       VOLSER------------333002       DEVTYPE------X'30502009'      FSEQN------------------0
     ASSCCIATIONS
       GDG------MJK.GDG1
       ALIAS----MJK.GDG1.ALIAS

ALIAS --------- MJK.GDG1.ALIAS
     HISTORY
       RELEASE----------------2       RCVY-VOL----------333001      RCVY-DEVT----X'30502009'      RCVY-CI--------X'000017'
     ASSOCIATIONS
       NONVSAM--MJK.GDG1.G0001V00

NONVSAM ------- MJK.NONVSAM1
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000018'
       RELEASE----------------2       EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     VOLUMES
       VOLSER------------333001       DEVTYPE------X'30502009'      FSEQN------------------0
     ASSOCIATIONS--------(NULL)

CLUSTER ------- MJKCAT
     HISTORY
       OWNER-IDENT-------(NULL)       CREATION----------77.040
       RELEASE----------------2       EXPIRATION--------99.999
     PROTECTION-PSWD-----(NULL)       RACF----------------(NO)
     ASSOCIATIONS
       DATA-----VSAM.CATALOG.BASE.DATA.RECORD
       INDEX----VSAM.CATALOG.BASE.INDEX.RECORD
```

Figure 14 (Part 4 of 7). Example of LISTCAT ALL Output

```
DATA ------- VSAM.CATALOG.BASE.DATA.RECORD
  HISTORY
    OWNER-IDENT-------(NULL)        CREATION----------00.000
    RELEASE----------------2        EXPIRATION--------00.000
  PROTECTION-PSWD-----(NULL)        RACF----------------(NO)
  ASSOCIATIONS
    CLUSTER--MJKCAT
  ATTRIBUTES
    KEYLEN-----------------44       AVGLRECL------------505     BUFSPACE-----------3072     CISIZE---------------512
    RKP--------------------0        MAXLRECL------------505     EXCPEXIT----------(NULL)    CI/CA-----------------40
    SHROPTNS(3,3)   RECOVERY        SUBALLOC          NOERASE   INDEXED        NOWRITECHK   IMBED          NOREPLICAT
    UNORDERED        NOREUSE        NONSPANNED          BIND    RECVABLE
  STATISTICS
    REC-TOTAL-------------15        SPLITS-CI--------------0    EXCPS-----------------24
    REC-DELETED------------0        SPLITS-CA--------------0    EXTENTS----------------2
    REC-INSERTED-----------0        FREESPACE-%CI----------0    SYSTEM-TIMESTAMP:
    REC-UPDATED------------0        FREESPACE-%CA----------0         X'8A51B6244C1B1000'
    REC-RETRIEVED----------0        FREESPC-BYTES-----381952
  ALLOCATION
    SPACE-TYPE---------TRACK
    SPACE-PRI-------------57        HI-ALLOC-RBA------389120
    SPACE-SEC-------------18        HI-USED-RBA-------389120
  VOLUME
    VOLSER------------333001        PHYREC-SIZE----------512    HI-ALLOC-RBA------368640    EXTENT-NUMBER----------1
    DEVTYPE------X'30502009'        PHYRECS/TRK-----------20    HI-USED-RBA--------20480    EXTENT-TYPE--------X'00'
    VOLFLAG------------PRIME        TRACKS/CA--------------3
    LOW-KEY---------------00
    HIGH-KEY--------------3F
    HI-KEY-RBA----------6144
    EXTENTS:
    LOW-CCHH-----X'00000002'        LOW-RBA----------------0    TRACKS----------------54
    HIGH-CCHH----X'00020011'        HIGH-RBA----------368639
  VOLUME
    VOLSER------------333001        PHYREC-SIZE----------512    HI-ALLOC-RBA------389120    EXTENT-NUMBER----------1
    DEVTYPE------X'30502009'        PHYRECS/TRK-----------20    HI-USED-RBA-------389120    EXTENT-TYPE--------X'00'
    VOLFLAG------------PRIME        TRACKS/CA--------------3
    LOW-KEY---------------40
    HIGH-KEY--------------FF
    HI-KEY-RBA--------368640
    EXTENTS:
    LOW-CCHH-----X'00030002'        LOW-RBA-----------368640    TRACKS-----------------3
    HIGH-CCHH----X'00030004'        HIGH-RBA----------389119
```

Figure 14 (Part 5 of 7). Example of LISTCAT ALL Output

```
INDEX ------ VSAM.CATALOG.BASE.INDEX.RECORD
  HISTORY
    OWNER-IDENT-------(NULL)        CREATION----------00.000
    RELEASE----------------2        EXPIRATION--------00.000
  PROTECTION-PSWD-----(NULL)        RACF----------------(NO)
  ASSOCIATIONS
    CLUSTER--MJKCAT
  ATTRIBUTES
    KEYLEN-----------------44       AVGLRECL--------------0       BUFSPACE--------------0       CISIZE---------------512
    RKP--------------------0        MAXLRECL-------------505      EXCPEXIT----------(NULL)      CI/CA-----------------20
    SHROPTNS(3,3)   RECOVERY        SUBALLOC          NOERASE     NOWRITECHK          IMBED     NOREPLICAT      UNORDERED
    NOREUSE               BIND
  STATISTICS
    REC-TOTAL--------------3        SPLITS-CI-------------0       EXCPS-----------------13      INDEX:
    REC-DELETED------------0        SPLITS-CA-------------0       EXTENTS----------------3      LEVELS----------------2
    REC-INSERTED-----------0        FREESPACE-%CI---------0       SYSTEM-TIMESTAMP:             ENTRIES/SECT----------7
    REC-UPDATED------------0        FREESPACE-%CA---------0            X'8A51B6244C1B1000'      SEQ-SET-RBA--------30720
    REC-RETRIEVED----------0        FREESPC-BYTES------38912                                    HI-LEVEL-RBA----------0
  ALLOCATION
    SPACE-TYPE---------TRACK
    SPACE-PRI--------------3        HI-ALLOC-RBA-------40448
    SPACE-SEC--------------3        HI-USED-RBA--------40448
  VOLUME
    VOLSER-----------333001         PHYREC-SIZE----------512      HI-ALLOC-RBA-------30720      EXTENT-NUMBER----------1
    DEVTYPE------X'30502009'        PHYRECS/TRK----------20       HI-USED-RBA----------512      EXTENT-TYPE--------X'00'
    VOLFLAG------------PRIME        TRACKS/CA--------------1
    EXTENTS:
    LOW-CCHH-----X'00020012'        LOW-RBA----------------0      TRACKS----------------3
    HIGH-CCHH----X'00030001'        HIGH-RBA----------30719
  VOLUME
    VOLSER-----------333001         PHYPEC-SIZE----------512      HI-ALLOC-RBA-------39936      EXTENT-NUMBER----------1
    DEVTYPE------X'30502009'        FHYRECS/TRK----------20       HI-USED-RBA--------31232      EXTENT-TYPE--------X'80'
    VOLFLAG------------PRIME        TRACKS/CA--------------3
    LOW-KEY--------------00
    HIGH-KEY-------------3F
    EXTENTS:
    LOW-CCHH-----X'00000002'        LOW-RBA----------30720        TRACKS----------------54
    HIGH-CCHH----X'00020011'        HIGH-RBA----------39935
  VOLUME
    VOLSER-----------333001         PHYREC-SIZE----------512      HI-ALLOC-RBA-------40448      EXTENT-NUMBER----------1
    DEVTYPE------X'30502009'        PHYRECS/TRK----------20       HI-USED-RBA--------40448      EXTENT-TYPE--------X'80'
    VOLFLAG------------PRIME        TRACKS/CA--------------3
    LOW-KEY--------------40
    HIGH-KEY-------------FF
    EXTENTS:
    LOW-CCHH-----X'00030002'        LOW-RBA----------39936        TRACKS----------------3
    HIGH-CCHH----X'00030004'        HIGH-RBA----------40447
```

Figure 14 (Part 6 of 7). Example of LISTCAT ALL Output

```
VOLUME -------- 333001
     HISTORY
       RELEASE----------------2        RCVY-VOL----------333001      RCVY-DEVT----X'30502009'      RCVY-CI--------X'000C00'
     CHARACTERISTICS
       BYTES/TRK----------13165        DEVTYPE------X'30502009'      MAX-PHYREC-SZ------13030      DATASETS-ON-VOL--------5
       TRKS/CYL-----------19           VOLUME-TIMESTAMP:             MAX-EXT/ALLOC----------5      DATASPCS-ON-VOL--------1
       CYLS/VOL-----------411               X'8A51B6A76850E000'
     DATASPACE
       DATASETS-----------5            FORMAT-1-DSCB:                ATTRIBUTES:
       EXTENTS------------1            CCHHR------X'0000000103'      SUBALLOC
       SEC-ALLOC----------40           TIMESTAMP                     EXPLICIT
       TYPE---------------TRACK             X'8A51B61D97B04000'      USERCAT
       EXTENT-DESCRIPTOR:
       TRACKS-TOTAL---------160        BEG-CCHH-----X'00000002'      SPACE-MAP----------5947
       TRACKS-USED----------89
       DATASET-DIRECTORY:
         DSN----MJKCAT                                               ATTRIBUTES--------(NULL)      EXTENTS----------------3
         DSN----MJK.CLUSTER.DATA                                     ATTRIBUTES--------(NULL)      EXTENTS----------------1
         DSN----MJK.CLUSTER1.INDEX                                   ATTRIBUTES--------(NULL)      EXTENTS----------------1
         DSN----MJK.ALT.INDEX1.DATA                                  ATTRIBUTES--------(NULL)      EXTENTS----------------1
         DSN----MJK.ALT.INDEX1.INDEX                                 ATTRIBUTES--------(NULL)      EXTENTS----------------1

                              LISTING FROM CATALOG -- MJKCAT

         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX -------------------1
                   ALIAS -----------------1
                   CLUSTER ---------------2
                   DATA ------------------3
                   GDG -------------------1
                   INDEX -----------------3
                   NONVSAM ---------------2
                   PAGESPACE -------------0
                   PATH ------------------1
                   SPACE -----------------1
                   USERCATALOG -----------0
                   TOTAL -----------------15

         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 (Part 7 of 7). Example of LISTCAT ALL Output

LISTCAT ALLOCATION OUTPUT LISTING

When you specify the LISTCAT command and include the ALLOCATION parameter, each cataloged object with space allocated to it from a VSAM data space is listed (see Figure 15). 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.

```
  /*    LIST SPACE ALLOCATION INFORMATION  */

       LISTCAT -
           ALLOCATION -
           CATALOG(MJKCAT)


AIX ----------- MJK.ALT.INDEX1
     HISTORY
       OWNER-IDENT-------(NULL)      CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000010'
       RELEASE----------------2      EXPIRATION--------00.000      RCVY-DEVT----X'30502009'

   DATA ------- MJK.ALT.INDEX1.DATA
     HISTORY
       OWNER-IDENT-------(NULL)      CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000011'
       RELEASE----------------2      EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI----------------6    HI-ALLOC-RBA-------73728
       SPACE-SEC----------------6    HI-USED-RBA--------73728
     VOLUME
       VOLSER------------333001      PHYREC-SIZE---------4096      HI-ALLOC-RBA-------73728      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'      PHYRECS/TRK------------3      HI-USED-RBA--------73728      EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME      TRACKS/CA--------------6
       EXTENTS:
       LOW-CCHH-----X'00040008'      LOW-RBA----------------0      TRACKS-----------------6
       HIGH-CCHH----X'0004000D'      HIGH-RBA----------73727

   INDEX ------ MJK.ALT.INDEX1.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)      CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000012'
       RELEASE----------------2      EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI----------------1    HI-ALLOC-RBA-------10240
       SPACE-SEC----------------1    HI-USED-RBA----------512
     VOLUME
       VOLSER------------333001      PHYREC-SIZE----------512      HI-ALLOC-RBA-------10240      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'      PHYRECS/TRK-----------20      HI-USED-RBA----------512      EXTENT-TYPE--------X'40'
       VOLFLAG------------PRIME      TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'0004000E'      LOW-RBA----------------0      TRACKS-----------------1
       HIGH-CCHH----X'0004000E'      HIGH-RBA----------10239

   PATH ------- MJK.AIX1.PATH
     HISTORY
       OWNER-IDENT-------(NULL)      CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'000014'
       RELEASE----------------2      EXPIRATION--------00.000      RCVY-DEVT----X'30502009'

CLUSTER ------- MJK.CLUSTER1
     HISTORY
       OWNER-IDENT-----OWNCLUST      CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'00000E'
       RELEASE----------------2      EXPIRATION--------77.040      RCVY-DEVT----X'30502009'
```

Figure 15 (Part 1 of 3). Example of LISTCAT ALLOCATION Output

```
   DATA ------- MJK.CLUSTER.DATA
     HISTORY
       OWNER-IDENT-------(NULL)      CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'00000D'
       RELEASE----------------2      EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI----------------2    HI-ALLOC-RBA-------24576
       SPACE-SEC----------------2    HI-USED-RBA--------24576
     VOLUME
       VOLSER------------333001      PHYREC-SIZE---------6144      HI-ALLOC-RBA-------24576      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'      PHYRECS/TRK------------2      HI-USED-RBA--------24576      EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME      TRACKS/CA--------------2
       EXTENTS:
       LOW-CCHH-----X'00040005'      LOW-RBA----------------0      TRACKS-----------------2
       HIGH-CCHH----X'00040006'      HIGH-RBA----------24575

   INDEX ------ MJK.CLUSTER1.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)      CREATION----------77.040      RCVY-VOL----------333001      RCVY-CI--------X'00000F'
       RELEASE----------------2      EXPIRATION--------00.000      RCVY-DEVT----X'30502009'
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI----------------1    HI-ALLOC-RBA-------10240
       SPACE-SEC----------------1    HI-USED-RBA----------512
     VOLUME
       VOLSER------------333001      PHYREC-SIZE----------512      HI-ALLOC-RBA-------10240      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'      PHYRECS/TRK-----------20      HI-USED-RBA----------512      EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME      TRACKS/CA--------------1
       EXTENTS:
       LOW-CCHH-----X'00040007'      LOW-RBA----------------0      TRACKS-----------------1
       HIGH-CCHH----X'00040007'      HIGH-RBA----------10239

CLUSTER ------- MJKCAT
     HISTORY
       OWNER-IDENT-------(NULL)      CREATION----------77.040
       RELEASE----------------2      EXPIRATION--------99.999

   DATA ------- VSAM.CATALOG.BASE.DATA.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)      CREATION----------00.000
       RELEASE----------------2      EXPIRATION--------00.000
     ALLOCATION
       SPACE-TYPE---------TRACK
       SPACE-PRI---------------57    HI-ALLOC-RBA------389120
       SPACE-SEC---------------18    HI-USED-RBA-------389120
     VOLUME
       VOLSER------------333001      PHYREC-SIZE----------512      HI-ALLOC-RBA------368640      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'      PHYRECS/TRK-----------20      HI-USED-RBA--------20480      EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME      TRACKS/CA--------------3
       LOW-KEY----------------00
       HIGH-KEY---------------3F
       HI-KEY-RBA----------6144
       EXTENTS:
       LOW-CCHH-----X'00000002'      LOW-RBA----------------0      TRACKS-----------------54
       HIGH-CCHH----X'00020011'      HIGH-RBA----------368639
     VOLUME
       VOLSER------------333001      PHYREC-SIZE----------512      HI-ALLOC-RBA------389120      EXTENT-NUMBER----------1
       DEVTYPE------X'30502009'      PHYRECS/TRK-----------20      HI-USED-RBA-------389120      EXTENT-TYPE--------X'00'
       VOLFLAG------------PRIME      TRACKS/CA--------------3
       LOW-KEY----------------40
       HIGH-KEY---------------FF
       HI-KEY-RBA--------368640
       EXTENTS:
       LOW-CCHH-----X'00030002'      LOW-RBA----------368640      TRACKS-----------------3
       HIGH-CCHH----X'00030004'      HIGH-RBA----------389119
```

Figure 15 (Part 2 of 3). Example of LISTCAT ALLOCATION Output

```
INDEX ------ VSAM.CATALOG.BASE.INDEX.RECORD
  HISTORY
    OWNER-IDENT-------(NULL)     CREATION----------00.000
    RELEASE----------------2     EXPIRATION--------00.000
  ALLOCATION
    SPACE-TYPE---------TRACK
    SPACE-PRI--------------3     HI-ALLOC-RBA-------40448
    SPACE-SEC--------------3     HI-USED-RBA--------40448
  VOLUME
    VOLSER------------333001     PHYREC-SIZE----------512     HI-ALLOC-RBA-------30720     EXTENT-NUMBER----------1
    DEVTYPE------X'30502009'     PHYRECS/TRK-----------20     HI-USED-RBA--------512       EXTENT-TYPE--------X'00'
    VOLFLAG------------PRIME     TRACKS/CA--------------1
    EXTENTS:
    LOW-CCHH-----X'00020012'     LOW-RBA----------------0     TRACKS-----------------3
    HIGH-CCHH----X'00030001'     HIGH-RBA-----------30719
  VOLUME
    VOLSER------------333001     PHYREC-SIZE----------512     HI-ALLOC-PBA-------39936     EXTENT-NUMBER----------1
    DEVTYPE------X'30502009'     PHYRECS/TRK-----------20     HI-USED-PBA--------31232     EXTENT-TYPE--------X'80'
    VOLFLAG------------PRIME     TRACKS/CA--------------3
    LOW-KEY---------------00
    HIGH-KEY--------------3F
    EXTENTS:
    LOW-CCHH-----X'00000002'     LOW-RBA------------30720     TRACKS----------------54
    HIGH-CCHH----X'00020011'     HIGH-RBA-----------39935
  VOLUME
    VOLSER------------333001     PHYREC-SIZE----------512     HI-ALLOC-RBA-------40448     EXTENT-NUMBER----------1
    DEVTYPE------X'30502009'     PHYRECS/TRK-----------20     HI-USED-RBA--------40448     EXTENT-TYPE--------X'80'
    VOLFLAG------------PRIME     TRACKS/CA--------------3
    LOW-KEY---------------40
    HIGH-KEY--------------FF
    EXTENTS:
    LOW-CCHH-----X'00030002'     LOW-RBA------------39936     TRACKS-----------------3
    HIGH-CCHH----X'00030004'     HIGH-RBA-----------40447

                          LISTING FROM CATALOG -- MJKCAT

       THE NUMBER OF ENTRIES PROCESSED WAS:
                 AIX -------------------1
                 ALIAS -----------------0
                 CLUSTER ---------------2
                 DATA ------------------3
                 GDG -------------------0
                 INDEX -----------------3
                 NONVSAM ---------------0
                 PAGESPACE -------------0
                 PATH ------------------1
                 SPACE -----------------0
                 USERCATALOG -----------0
                 TOTAL ----------------10

       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 (Part 3 of 3). Example of LISTCAT ALLOCATION Output

LISTCAT HISTORY OUTPUT LISTING

When you specify the LISTCAT command and include the HISTORY parameter, only the name, ownerid, creation date, and expiration date are listed for each entry that is selected (see Figure 16). Only these types of entries have HISTORY information: ALTERNATEINDEX, CLUSTER, DATA, INDEX, NONVSAM, and PATH.

```
/*  LIST HISTORY INFORMATION FOR EACH ENTRY */

LISTCAT -
  HISTORY -
  CATALOG(MJKCAT)


                          LISTING FROM CATALOG -- MJKCAT

AIX ----------- MJK.ALT.INDEX1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040     RCVY-VOL----------333001     RCVY-CI--------X'000010'
       RELEASE----------------2     EXPIRATION--------00.000     RCVY-DEVT----X'30502009'

   DATA ------- MJK.ALT.INDEX1.DATA
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040     RCVY-VOL----------333001     RCVY-CI--------X'000011'
       RELEASE----------------2     EXPIRATION--------00.000     RCVY-DEVT----X'30502609'

   INDEX ------ MJK.ALT.INDEX1.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040     RCVY-VOL----------333001     RCVY-CI--------X'000012'
       RELEASE----------------2     EXPIRATION--------00.000     RCVY-DEVT----X'30502009'

   PATH ------- MJK.AIX1.PATH
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040     RCVY-VOL----------333001     RCVY-CI--------X'000014'
       RELEASE----------------2     EXPIRATION--------00.000     RCVY-DEVT----X'30502009'

CLUSTER ------- MJK.CLUSTER1
     HISTORY
       OWNER-IDENT-----OWNCLUST     CREATION----------77.040     RCVY-VOL----------333001     RCVY-CI--------X'00000E'
       RELEASE----------------2     EXPIRATION--------77.140     RCVY-DEVT----X'30502009'

   DATA ------- MJK.CLUSTER.DATA
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040     RCVY-VOL----------333001     RCVY-CI--------X'00000D'
       RELEASE----------------2     EXPIRATION--------00.000     RCVY-DEVT----X'30502009'

   INDEX ------ MJK.CLUSTER1.INDEX
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040     RCVY-VOL----------333001     RCVY-CI--------X'00000F'
       RELEASE----------------2     EXPIRATION--------00.000     RCVY-DEVT----X'30502009'

GDG BASE ------ MJK.GDG1
     HISTORY
       OWNER-IDENT-------OWNGDG     CREATION----------75.223     RCVY-VOL----------333001     RCVY-CI--------X'000015'
       RELEASE----------------2     EXPIRATION--------99.365     RCVY-DEVT----X'30502009'

   NONVSAM ---- MJK.GDG1.G0001V00
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040     RCVY-VOL----------333001     RCVY-CI--------X'000016'
       RELEASE----------------2     EXPIRATION--------00.000     RCVY-DEVT----X'30502009'

ALIAS --------- MJK.GDG1.ALIAS
     HISTORY
       RELEASE----------------2     RCVY-VOL----------333001     RCVY-DEVT----X'30502009'     RCVY-CI--------X'000017'

NONVSAM ------- MJK.NONVSAM1
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040     RCVY-VOL----------333001     RCVY-CI--------X'000018'
       RELEASE----------------2     EXPIRATION--------00.000     RCVY-DEVT----X'30502009'
```

Figure 16 (Part 1 of 2). Example of LISTCAT HISTORY Output

```
CLUSTER ------- MJKCAT
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------77.040
       RELEASE----------------2     EXPIRATION--------99.999

   DATA ------- VSAM.CATALOG.BASE.DATA.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------00.000
       RELEASE----------------2     EXPIRATION--------00.000

   INDEX ------ VSAM.CATALOG.BASE.INDEX.RECORD
     HISTORY
       OWNER-IDENT-------(NULL)     CREATION----------00.000
       RELEASE----------------2     EXPIRATION--------00.000

VOLUME -------- 333001
     HISTORY
       RELEASE----------------2     RCVY-VOL----------333001     RCVY-DEVT----X'30502009'      RCVY-CI--------X'000009'


                            LISTING FROM CATALOG -- MJKCAT


         THE NUMBER OF ENTRIES PROCESSED WAS:
                   AIX ------------------1
                   ALIAS ----------------1
                   CLUSTER --------------2
                   DATA -----------------3
                   GDG ------------------1
                   INDEX ----------------3
                   NONVSAM --------------2
                   PAGESPACE ------------0
                   PATH -----------------1
                   SPACE ----------------1
                   USERCATALOG ----------0
                   TOTAL ----------------15

         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 (Part 2 of 2). Example of LISTCAT HISTORY 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.

In Figure 17, for example, because all entries in the listed catalog, USERCAT3, were created on the same day as the LISTCAT, no entries are listed as a result of the LISTCAT CREATION(5) job. When that job is run on an older catalog, each entry whose creation date is prior to the number of days specified with the CREATION parameter 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. The creation date of the data and index object of a catalog is always set to 0.

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.

When the LISTCAT EXPIRATION(20) job is run, each entry whose expiration date occurs within 20 days of today's date is listed. Because the expiration date of a cluster or alternate index is controlled by the cluster or alternate index object entry in the catalog, the expiration date for the data and index objects of a cluster or alternate index is always 0.

When you list all entries of a catalog and you specify the EXPIRATION parameter, each volume entry will be listed, because volume entries have no expiration date.

These types of entries can have a creation or expiration date: ALTERNATEINDEX, PATH, CLUSTER, DATA, INDEX, NONVSAM, GDG, and PAGESPACE.

```
/*     LIST EACH CATALOG ENTRY WHOSE CREATION DATE   */
/*     IS 5 DAYS AGO OR GREATER (THAT IS, THE OBJECT */
/*     IS AT LEAST 5 DAYS OLD).                      */

LISTCAT -
   CREATION(5) -
   CATALOG(MJKCAT)

                          LISTING FROM CATALOG -- MJKCAT

ALIAS --------- MJK.GDG1.ALIAS
```

Figure 17 (Part 1 of 2). Example of LISTCAT CREATION/EXPIRATION Output

```
                              LISTING FROM CATALOG -- USERCAT3

          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

/*      LIST EACH CATALOG ENTRY WHOSE EXPIRATION DATE */
/*      IS WITHIN 20 DAYS*/

LISTCAT -
   EXPIRATION(20) -
   CATALOG(MJKCAT)

                              LISTING FROM CATALOG -- USERCAT3

AIX ---------- MJK.ALT.INDEX1

   DATA ------- MJK.ALT.INDEX1.DATA

   INDEX ------ MJK.ALT.INDEX1.INDEX

   PATH ------- MJK.AIX1.PATH

ALIAS --------- MJK.GDG1.ALIAS

NONVSAM ------- MJK.NONVSAM1

VOLUME -------- 333001

                              LISTING FROM CATALOG -- USERCAT3

          THE NUMBER OF ENTRIES PROCESSED WAS:
                    AIX ------------------1
                    ALIAS ----------------1
                    CLUSTER --------------0
                    DATA -----------------1
                    GDG ------------------0
                    INDEX ----------------1
                    NONVSAM --------------1
                    PAGESPACE ------------0
                    PATH -----------------1
                    SPACE ----------------1
                    USERCATALOG ----------0
                    TOTAL ----------------7

          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 17 (Part 2 of 2). Example of LISTCAT CREATION/EXPIRATION 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: AMAST1
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 B. INTERPRETING LISTCRA OUTPUT LISTINGS

When you code the LISTCRA command, you can specify options that allow you to tailor the contents of the LISTCRA output. This appendix illustrates the various types of LISTCRA output, the order in which entries are listed, and the meanings of the listed fields.

There are five kinds of LISTCRA listings. Four are illustrated in this appendix; the fifth, SEQUENTIALDUMP, is the same as that for DUMP NOCOMPARE, except that the entries are not sorted into groups.

Each listed entry is identified by type (that is, by cluster, non-VSAM, and so on) and by name. Entries are usually listed in alphabetic order within groups of entries, according to entryname (except for SEQUENTIALDUMP). However, if insufficient virtual storage is available for the sorting operation, the records are listed as they appear in the catalog recovery area.

On the listing, entries are sorted into three groups (except for a SEQUENTIALDUMP listing):

- VSAM entries (which are also sorted according to entryname). Cluster entries, alternate index entries, and their associated data, index, and path entries are within this group.
- Other entries that are sorted according to entryname. User catalog connector entries and non-VSAM entries are in this group.
- Unsorted entries: other entries that are not sorted according to entryname. Entries that are listed within this group depend on the type of LISTCRA listing requested.

The following list contains the abbreviation, type, and description for each kind of entry that can be listed as a result of the LISTCRA command:

Abbreviation	Type	Description
AIX	G	Alternate index entry
ALIA	X	Alias entry
CLUS	C	Cluster entry
DATA	D	Data component entry
FRSP	F	Freespace entry
GDGB	B	Generation data group entry
INDX	I	Index component entry
NONV	A	Non-VSAM data set entry
OEXT	E	Extension record for an entry other than a volume entry
PATH	P	Path entry
UCAT	U	User catalog connector entry (in the master catalog only)
UPGD	Y	Upgrade set entry

Abbreviation	Type	Description
VEXT	W	Extension record for a volume entry
VOL	V	Volume entry

EXAMPLES OF LISTCRA LISTINGS

The five kinds of LISTCRA listings are:

- NAME NOCOMPARE—This is the default option. Each entryname, its volumes, and the name and volumes of each related entry are listed. Within the "unsorted entries" group, all catalog records not yet printed are printed. See Figure 18 on page 306.

- DUMP NOCOMPARE—When DUMP is specified (NOCOMPARE is a default), each record is listed in the dump format. In addition, the name and volumes of each indirectly related entry are also listed. Within the "unsorted entries" group, the CRA's self-describing records (control intervals through 8) are printed, and any free space records and records not yet printed are printed. See Figure 19 on page 309.

- NAME COMPARE—When COMPARE is specified (NAME is a default), only the name of each miscompared catalog entry is listed. A miscompared catalog entry is one whose information is not identical to the information contained in the entry's copy in the catalog recovery area. A MISCOMPARE message is printed that identifies the most serious level of miscomparison. Within the "unsorted entries" group, miscompared records not yet printed out are printed. See Figure 20 on page 312.

- DUMP COMPARE—When DUMP and COMPARE are specified, only the records (in dump format) of each miscompared catalog entry are listed. A miscompared catalog entry is one whose information is not identical to the information contained in the entry's copy in the catalog area. For each entry, the catalog recovery area copy is listed first, followed by the catalog entry, followed by a line that contains asterisks to identify each miscompared byte. Following each entry, a MISCOMPARE message is printed that identifies the most serious level of miscomparison. Within the "unsorted entries" group, miscompared records not yet printed out are printed in the dump format. See Figure 21 on page 313.

  Note: As explained above, all MISCOMPARE messages result from a comparison between the catalog and the CRA. If the comparison shows that the catalog and CRA records are not identical, no inference is given as to which is correct. You must make this determination yourself by looking at other mismatches in the same listing and by examining related records in the catalog or CRA.

- SEQUENTIALDUMP—When SEQUENTIALDUMP is specified, each record in the catalog recovery area is printed. The format of the output is the same as that of the DUMP NOCOMPARE, except that the entries are not sorted into groups or alphabetic sequence.

  See Catalog Diagnosis Reference for a complete description of catalog recovery area record formats, catalog record formats, and relationships between catalog records.

JOB CONTROL LANGUAGE (JCL) FOR LISTCRA JOBS

The job control language (JCL) statements that can be used to list catalog recovery areas are:

```
//LISTCAT   JOB    ...
//STEP1     EXEC   PGM=IDCAMS
//STEPCAT   DD     DSN=YOURCAT,DISP=SHR
//CATDD     DD     DSN=YOURCAT,DISP=OLD
//CRADD1    DD     VOL=SER=333001,UNIT=DISK,DISP=OLD
//CRADD2    DD     VOL=SER=333002,UNIT=DISK,DISP=OLD
//OUTDD     DD     DSN=LISTCRA.OUTPUT,UNIT=TAPE,
//           VOL=SER=TAPE10,LABEL=(1,NL),DISP=(NEW,KEEP),
//           DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)
//SYSPRINT  DD     SYSOUT=A
//SYSIN     DD     *
     LISTCRA -
        COMPARE -
        DUMP -
        INFILE(CRADD1,CRADD2) -
        CATALOG(YOURCAT/PASSWORD CATDD) -
        MASTERPW(SECRET) -
        OUTFILE(OUTDD)
/*
```

The JOB statement contains user and accounting information required for your installation.

The EXEC statement identifies the program to be executed, IDCAMS (that is, the access method services program).

- STEPCAT DD, which allocates your catalog. This is only required if the COMPARE option is specified. The catalog must be open as a catalog before it can be opened as a data set through the DD statement that so identifies it for the comparisons.
- CATDD, which specifies the catalog to be opened as a data set and compared to the catalog recovery areas. This is only required if the COMPARE option is specified.
- CRADD1, which specifies a volume whose catalog recovery area (CRA) is to be listed.
- CRADD2, which specifies another volume whose CRA is to be listed.
- OUTDD, which specifies an alternate output file, so that the LISTCRA output can be written onto an auxiliary storage device. The LISTCRA command's OUTFILE parameter points to the OUTDD DD statement. Only the LISTCRA output is written to the alternate output device. JCL statements, system messages, and job statistics are written to the SYSPRINT output device.
  - DSN=LISTCRA.OUTPUT specifies the name for the magnetic tape file.
  - UNIT=TAPE and VOL=SER=TAPE10 specify 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 LISTCRA 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, and so on).
  - 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 LISTCRA command, DISP=(MOD,KEEP) or DISP=(MOD,PASS) can be used to concatenate all of the LISTCRA output in one sequential file.

- DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629) specifies that the LISTCRA 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 LISTCRA 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 'OUTDD' with '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 '/*' terminates the input data statements.

The LISTCRA command parameters shown above are common to the LISTCRA DUMP COMPARE example that follows. Other LISTCRA parameters may be coded and the output that results is illustrated.

COMPARE, which specifies that the CRA entries are to be compared with the catalog entries identified by the CATALOG parameter. Only those that miscompare will be listed.

DUMP, which specifies that each listed entry is to be printed in its entirety in both hexadecimal and character form.

INFILE, which specifies the two DD statements, CRADD1 and CRADD2, which identify the two volumes whose CRAs are to be compared with the catalog entries and the miscomparing entries listed.

CATALOG, which identifies the DD statement, CATDD, which identifies the catalog, YOURCAT, whose entries are to be compared with those in the CRAs. If the catalog is password protected, its master password, PASSWORD, is required.

MASTERPW, which specifies the master password of the master catalog, SECRET, to enable the CRAs to be OPENed.

OUTFILE, which points to the OUTDD DD statement. The OUTDD DD statement allocates an alternate output file for the LISTCRA output.

If you want to print the LISTCRA 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, LISTCRA.OUTPUT, that was allocated previously:

```
//PRINTOUT   JOB    ...
//STEP1      EXEC   PGM=IEBGENER
//SYSUT1     DD     DSN=LISTCRA.OUTPUT,UNIT=TAPE,
               DCB=(RECFM=VBA,LRECL=125,BLKSIZE=629)
//SYSUT2     DD     SYSOUT=A
//SYSPRINT   DD     SYSOUT=A
//SYSIN      DD     DUMMY
/*
//
```

LISTCRA OUTPUT LISTING

When you specify LISTCRA with no parameters, each entryname, its volume(s), and the name and volume(s) of each related entry are listed. The same listing would result if the NAME and NOCOMPARE parameters were specified.

You can use this type of listing to list the name of each catalog entry whose copy is in the catalog recovery area, and to determine the number of entries in each catalog recovery area. The total number of entries is an approximate size, in records, of the catalog recovery area.

```
  LISTING OF CATALOG RECOVERY AREA FOR VOLUME -- 333301 -- VSAM ENTRIES

VOL  - 333301
    CRAVOLRCD  -  10/07/74   23:35:00
    F4DSCBVSAM -  10/07/74   23:35:00
    F4DSCBDUMP -  10/07/74   23:35:00

CLUS - AA.LISTCRA.ESDS

  DATA - TF41C9A0.VSAMDSET.DFD74280.T861DAD4.TF41C9A0
    DATA VOL -
      333301

  AIX  - AA2.LR.ESDS
    DATA VOL -
      333301
    INDX VOL -
      333301

  AIX  - AA1.LR.ESDS
    DATA VOL -
      333301
    INDX VOL -
      333301

  PATH - AAU.LR.ESDS

  UPGD -

CLUS - AA.LISTCRA.KSDS

  DATA - T5C55DD0.VSAMDSET.DFD74280.T861DB0A.T5C55DD0
    DATA VOL -
      333301

  INDX - T5C56E70.VSAMDSET.DFD74280.T861DB0A.T5C56E70
    INDX VOL -
      333301

AIX  - AA1.LR.ESDS

  DATA - TCD41020.VSAMDSET.DFD74280.T861DB23.TCD41020
    DATA VOL -
      333301

  INDX - TCD41F00.VSAMDSET.DFD74280.T861DB23.TCD41F00
    INDX VOL -
      333301

  CLUS - AA.LISTCRA.ESDS

  PATH - AA1U.LR.ESDS

  PATH - AA1N.LR.ESDS

CLUS - LR.MCKEYRNG.KSDS

  DATA - TF081480.VSAMDSET.DFD74280.T861DB0E.TF081480
    DATA VOL - HIGH KEY
      333301 - C1C1C1C1C1C1C1C1C5F9
      333301 - C1C1C1C1C1C1C1C1E9E9

  INDX - TF082370.VSAMDSET.DFD74280.T861DB0E.TF082370
    INDX VOL -
      333301
      333301
```

Figure 18 (Part 1 of 2). Example of LISTCRA NAME NOCOMPARE Output

```
  LISTING OF CATALOG RECOVERY AREA FOR VOLUME -- 333301 -- OTHER ENTRIES

UCAT - USERCAT2
  DATA VOL -
    333304

NUMBER OF ENTRIES PROCESSED
 CLUS -  3
 DATA -  4
 AIX  -  1
 INDX -  2
 PATH -  3
 VOL  -  1
 UPGD -  1
 SUM  - 15
```

Figure 18 (Part 2 of 2). Example of LISTCRA NAME NOCOMPARE Output

LISTCRA NAME NOCOMPARE Output Notes:

- The title line of the output contains the volume serial number and identifies the group that the following entries are within (in this case, VSAM Entries).
- The time stamps:

  CRAVOLRCD is the timestamp of the CRA volume record. It is updated only when the first catalog record and first CRA volume record are updated, after the CRA is opened when VSAM is running. (**Note:** If the volume is moved to a DOS/VS system and used there, the CRAVOLRCD timestamp is updated each time the catalog and CRA volume records are updated for a space allocation change).

  F4DSCBVSAM is a timestamp in the format-4 DSCB, and is updated in the same manner as the CRAVOLRCD timestamp.

  F4DSCBDUMP is also a timestamp in the format-4 DSCB, and is updated in the same manner as the CRAVOLRCD timestamp. It is also updated whenever the operating system utility program IEHDASDR dumps the volume.
- VSAM entries are listed along with the entryname and volume(s) of each related entry.
- Paths are shown as related only to the entry to which the path provides access. A path that serves as an alias for a VSAM data set is listed with the data-set's entry. A path that shows the relationship of an alternate index to a base cluster is listed with the alternate-index's entry only.
- UPGD indicates that there are alternate indexes in the base-cluster's upgrade set. If you want to identify the indexes in the upgrade set, use the DUMP NOCOMPARE option.
- The high-key value of each keyrange is shown.
- SUM is the total number of catalog recovery area entries printed.

LISTCRA DUMP OUTPUT LISTING

When you specify LISTCRA with the DUMP parameter, each record in the catalog recovery area is listed in dump format. In addition, the name and volumes of each indirectly related entry are also listed. The same listing would result if the DUMP and NOCOMPARE parameters were specified.

When you specify LISTCRA SEQUENTIALDUMP, the same listing results, except that the records are listed in their entry sequence within the catalog recovery area.

You can use this type of listing to list the complete contents of each catalog entry whose copy is in the catalog recovery area, and to determine the exact number of entries and records in each catalog recovery area.

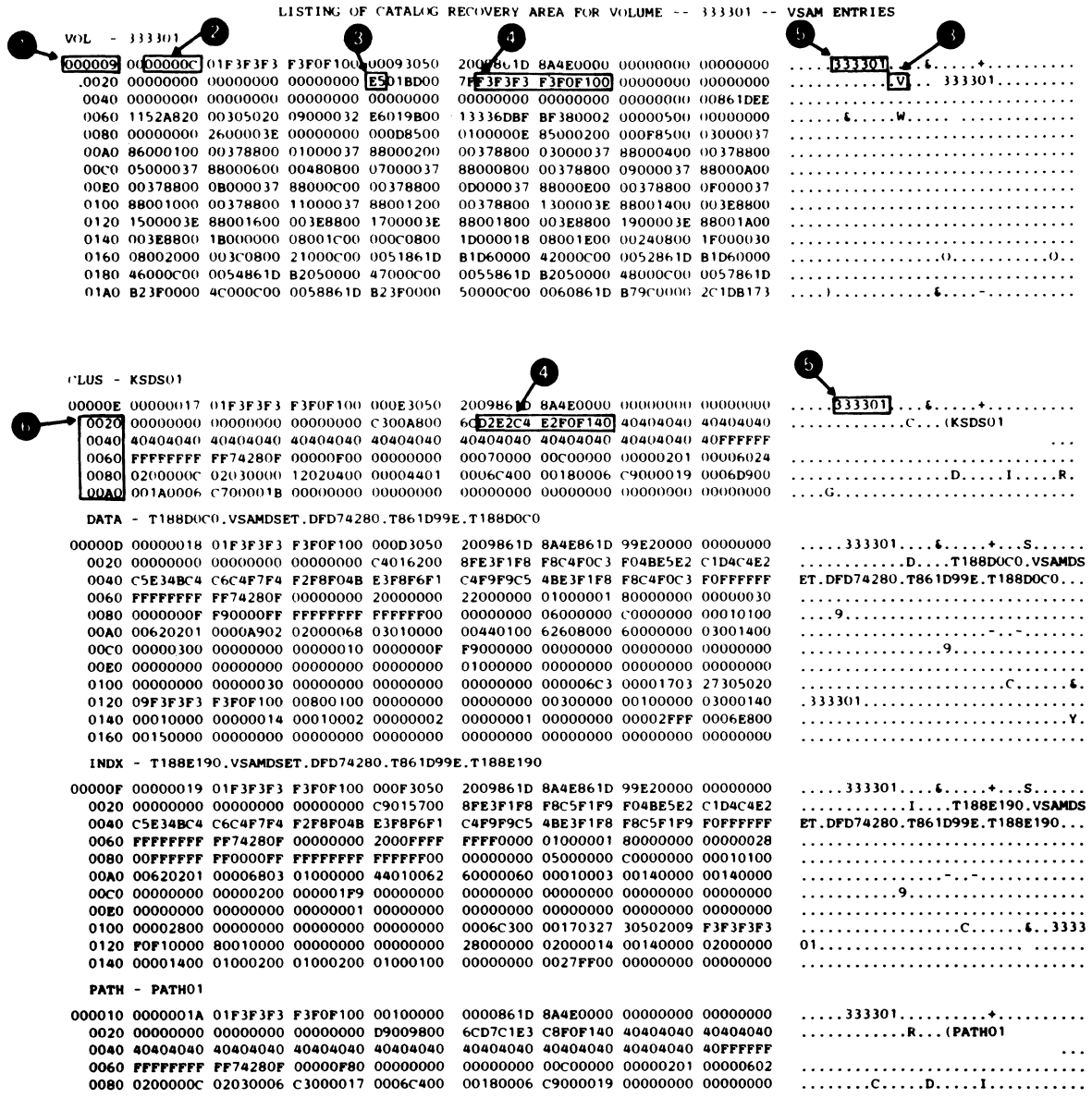


Figure 19 (Part 1 of 2). Example of LISTCRA DUMP NOCOMPARE Output

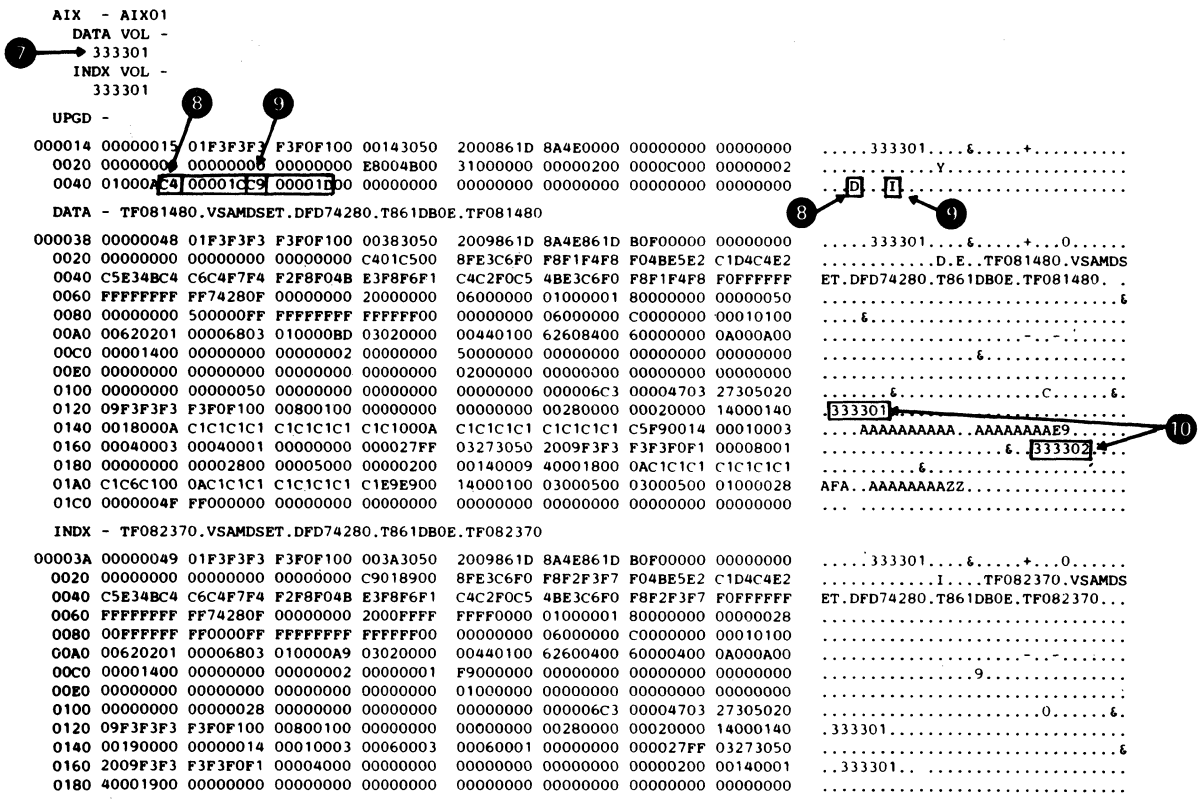


Figure 19 (Part 2 of 2). Example of LISTCRA DUMP NOCOMPARE Output

LISTCRA DUMP NOCOMPARE Output Notes:

Note: When you specify SEQUENTIALDUMP, the same information is listed. The entries are not sorted according to type of entry, or according to alphanumeric-sequence of the entryname field. See also Note 7, below.

1. The 3-byte CRA control interval number, in hexadecimal form.

2. The 3-byte control interval number of the catalog record for which this is a copy, in hexadecimal form.

3. The entrytype at offset X'2C' (decimal 44), a character that identifies the type of catalog entry being listed.

4. A 44-byte field at offset X'31' (decimal 49) that contains the entry's entryname (padded with binary zeros), or an 8-byte field followed by 36 bytes of binary zeros that contain the volume entry's volume serial number.

5. The volume serial number of the recovery volume.

6. A 2-byte displacement value. Printing is suppressed after the last line containing data is listed, even though all catalog records are 512 bytes long.

7. As in the NAME NOCOMPARE output, the alternate index name and the fact that there is an upgrade set is given, along with the volume serial numbers of the alternate index's data and index components. (**Note:** If SEQUENTIALDUMP is specified, the information described with this note is not listed.)

8. The first byte identifies the alternate index's data component and the next three bytes contain its catalog control interval number.

9. The first byte identifies the alternate index's index component and the next three bytes contain its catalog control interval number.

10. Because there are two key ranges for this cluster, there are two volume information sets of fields in the cluster's data and index component entries.

LISTCRA COMPARE OUTPUT LISTING

When you specify LISTCRA with the COMPARE parameter, each record in the catalog recovery area is compared with its original in the catalog. The name of each catalog entry that miscompares is listed. A miscompared catalog entry is one whose information is not identical to the information contained in the entry's copy in the catalog recovery area. The same listing would result if the NAME and COMPARE parameters were specified.

You can use this type of listing to determine the number of damaged entries in your catalog, and you can get an idea of the type of damage that occurred to each entry. You can issue the EXPORTRA and IMPORTRA commands to recover the catalog and make its damaged entries usable.

```
LISTING OF CATALOG RECOVERY AREA FOR VOLUME -- 333301 -- VSAM ENTRIES

    CATVOLRCD  -  09/17/42   23:53:47
    CRAVOLRCD  -  10/07/74   23:35:00
    F4DSCBVSAM -  10/07/74   23:35:00
    F4DSCBDUMP -  10/07/74   23:35:00

CLUS - LR.DELETED.ESDS

* MISCOMPARES - CATALOG ENTRY HAS DIFFERENT NAME

CLUS - LR.MCHURBA.ESDS

  DATA - T73FEDA0.VSAMDSET.DFD74280.T861DAE1.T73FEDA0
    DATA VOL -
      333301

* MISCOMPARES - HIGH USED RBA

CLUS - LR.MCKEYRNG.KSDS

  DATA - TF081480.VSAMDSET.DFD74280.T861DB0E.TF081480
    DATA VOL - HIGH KEY
      333301 - C1C1C1C1C1C1C1C1C5F9
      333301 - C1C1C1C1C1C1C1C1E9E9

* MISCOMPARES - HIGH USED RBA

  INDX - TF082370.VSAMDSET.DFD74280.T861DB0E.TF082370
    INDX VOL -
      333301
      333301

* MISCOMPARES - STATISTICS

NUMBER OF ENTRIES PROCESSED
 CLUS - 23
 DATA - 18
 AIX  -  8
 INDX - 11
 PATH -  5
 VOL  -  1
 UPGD -  3
 SUM  - 69

IDC0665I NUMBER OF ENTRIES THAT MISCOMPARED IN THIS CRA - 3

IDC0877I NUMBER OF RECORDS THAT MISCOMPARED IN THIS CRA - 4
```

Figure 20. Example of LISTCRA NAME COMPARE Output

LISTCRA NAME COMPARE Output Notes:

- In addition to the timestamps described for the NAME NOCOMPARE output (noted previously), the timestamp in the catalog volume record, CATVOLRCD, is listed. CATVOLRCD is updated in the same manner as the CRAVOLRCD timestamp.
- The MISCOMPARES message always refers to the record listed above it. See "Regaining Access to Data" in the Catalog Administration Guide for the cause and seriousness of this MISCOMPARES message.
- The number of records is sometimes larger than the number of entries, because an entry might consist of a catalog record and one or more extension records that failed to compare correctly.

LISTCRA DUMP COMPARE OUTPUT LISTING

When you specify LISTCRA with the DUMP and COMPARE parameters, each record in the catalog recovery area is compared to its original in the catalog. Each catalog entry's copy in the catalog recovery area that miscompares is listed, followed by its original catalog entry (which is damaged), followed by asterisks to indicate each miscompared byte.

You can use this type of listing to determine the exact damage that occurred to each entry. You might be able to correct some or all of the damage by using the ALTER command. You can issue the EXPORTRA and IMPORTRA commands to recover the catalog and make its damaged entries usable.

```
                    LISTING OF CATALOG RECOVERY AREA FOR VOLUME -- 333301 -- VSAM ENTRIES

   CLUS - LR.MCHURBA.ESDS

000020 0000002C 01F3F3F3 F3F0F100 00203050  2009861D 8A4E0000 00000000 00000000   .....333301....&.....+.........
CATRCD 0000002C 01F3F3F3 F3F0F100 00203050  2009861D 8A4E0000 00000000 00000000   .....333301....&.....+.........

  0020 00000000 00000000 00000000 C3008200  6CD3D94B D4C3C8E4 D9C2C14B C5E2C4E2   ............C....(LR.MCHURBA.ESDS
       00000000 00000000 00000000 C3008200  6CD3D94B D4C3C8E4 D9C2C14B C5E2C4E2   ............C....(LR.MCHURBA.ESDS

  0040 40404040 40404040 40404040 40404040  40404040 40404040 40404040 40FFFFFF                              ...
       40404040 40404040 40404040 40404040  40404040 40404040 40404040 40FFFFFF                              ...

  0060 FFFFFFFF FF74280F 00000F00 00000000  00000000 00020100 00004401 0006C400   ..............................D.
       FFFFFFFF FF74280F 00000F00 00000000  00000000 00020100 00004401 0006C400   ..............................D.

  0080 002D0000 00000000 00000000 00000000  00000000 00000000 00000000 00000000   ................................
       002D0000 00000000 00000000 00000000  00000000 00000000 00000000 00000000   ................................

   DATA - T73FEDA0.VSAMDSET.DFD74280.T861DAE1.T73FEDA0

000021 0000002D 01F3F3F3 F3F0F100 00213050  2009861D 8A4E861D AE2E0000 00000000   .....333301....&.....+.........
CATRCD 0000002D 01F3F3F3 F3F0F100 00213050  2009861D 8A4E861D AE2E0000 00000000   .....333301....&.....+.........

  0020 00000000 00000000 00000000 C4015700  8FE4F7F3 C6C5C4C1 F04BE5E2 C1D4C4E2   ............D....T73FEDA0.VSAMDS
       00000000 00000000 00000000 C4015700  8FE4F7F3 C6C5C4C1 F04BE5E2 C1D4C4E2   ............D....T73FEDA0.VSAMDS

  0040 C5E34BC4 C6C4F7F4 F2F8F04B E3F8F6F1  C4C1C5F1 4BE3F7F3 C6C5C4C1 F0FFFFFF   ET.DFD74280.T861DAE1.T73FEDA0...
       C5E34BC4 C6C4F7F4 F2F8F04B E3F8F6F1  C4C1C5F1 4BE3F7F3 C6C5C4C1 F0FFFFFF   ET.DFD74280.T861DAE1.T73FEDA0...

  0060 FFFFFFFF FF74280F 00000000 20000000  04000000 01000001 80000000 00000028   ................................
       FFFFFFFF FF74280F 00000000 20800000  04000000 01000001 80000006 00000028   ................................
*                                  **                                    **
  0080 00000000 500000FF FFFFFFFF FFFFFF00  00000000 05000000 00000000 00010100   ....&...........................
       00000000 500000FF FFFFFFFF FFFFFF00  00000000 05000000 00000000 00010100   ....&...........................

  00A0 00620201 00006803 01000000 44010062  60000060 00000000 00000000 00140000   ..............-..-..............
       00620201 00006803 01000000 44010062  60000060 00000000 00000000 00140000   ..............-..-..............

* MISCOMPARES - HIGH USED RBA
```

Figure 21. Example of LISTCRA DUMP COMPARE Output

LISTCRA DUMP COMPARE Output Notes:

1. The 3-byte CRA control interval number in hexadecimal form.

2. Each pair of hexadecimal-data lines is the information in the catalog recovery area's copy of the catalog record, followed by the information in the catalog record, for the same displacement.

3. Asterisks are printed below the bytes in which a miscompare exists and, in the left margin, the miscompared lines are flagged with a single asterisk.

4. See "Regaining Access to Data" in the Catalog Administration Guide for the cause and seriousness of this MISCOMPARES message.

APPENDIX C. SAMPLE OUTPUT FROM CHKLIST

Figure 22 shows the format of output from the CHKLIST command.

For each checkpoint entry listed under "CHECKID", the other columns give dsname, ddname, unit, and volume information for each tape data set that was open at the time of the checkpoint. Asterisks under "UNIT" indicate an unrecognizable unit type. "VOLUME X OF Y IS CURRENT" indicates the volume sequence number of the volume mounted at the time of the checkpoint and the total number of volumes in the data set. each volume's volume serial number is listed, and an asterisk indicates the volume mounted at the time of the checkpoint.

```
CHKLIST INFILE(CHKPT) CHECKID(C0000001 C0000002)
IDCAMS  SYSTEM SERVICES          TIME: 21:34:43       01/16/75
OPEN TAPE DATA SET LIST FROM CHECKPOINT DATA SET - CHKPT DATASET
CHECKID         DSNAME                DDNAME   UNIT       VOLUMES - * INDICATES CURRENT VOLUME
C0000001
                USER.TAPE.DATASET0    DDN2VS11 2400-7TRK  VOLUME    1 OF   1 IS CURRENT
                                                          120001*
                USER.TAPE.DATASET1    DDN3VS11 2400-9TRK  VOLUME    3 OF   4 IS CURRENT
                                                          130001   130002  130003* 130004
                USER.TAPE.DATASET2    DDN4VS11 3400-7TRK  VOLUME    1 OF   1 IS CURRENT
                                                          140001*
C0000002
                USER.TAPE.DATASET3    DDNAME32 2400-9TRK  VOLUME    8 OF   8 IS CURRENT
                                                          230001   230002  230003  230004  230005
                                                          230006   230007  230008*
                USER.TAPE.DATASET4    DDNAME42 2400-9TRK  VOLUME   20 OF  24 IS CURRENT
                                                          240001   240002  240003  240004  240005
                                                          240006   240007  240008  240009  240010
                                                          240011   240012  240013  240014  240015
                                                          240016   240017  240018  240019  240020*
                                                          240021   240022  240023  240024
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

Figure 22. Example of CHKLIST Output

APPENDIX D. CAUTION FOR JCL DD PARAMETERS

You should be aware that some specifications do nothing in a VSAM environment or might cause problems for you. Figure 23 shows the DD statement parameters and subparameters to be avoided with VSAM. For more information, see JCL.

Parameter	Subparameter	Comment
AFF	ddname	You must use this parameter carefully. If the cluster components (the data and its index) reside on unlike devices, the results of an AFF specification are unpredictable.
CHKPT	**EOV**	Because checkpoints at end of volume are executed only for BSAM and QSAM data sets, this parameter does not apply to VSAM data sets and need not be coded.
DATA		Because there is no way to get VSAM data into the input stream, this parameter is not applicable to VSAM.
DCB	All	The access method control block (ACB), not the DCB, describes VSAM data sets; therefore, the DCB parameter is not applicable to VSAM. An access method control block is generated by an ACB or GENCB macro, and can be modified by a MODCB macro.
DISP	**CATLG**	CATLGVSAM data sets are cataloged and uncataloged as a result of an access method services command; if CATLG is coded, a message is issued, but the data set is not cataloged.
	DELETE	VSAM data sets are deleted as a result of an access method services command; if DELETE is coded, a message is issued, but the data set is not deleted.
	MOD	For VSAM data sets, MOD is treated as if OLD were specified, except for processing with an ISAM program, in which case MOD indicates resume load.
	KEEP	Because KEEP is implied for VSAM data sets, it need not be coded.
	NEW	VSAM data sets are initially allocated as a result of the access method services DEFINE command. If NEW is specified, space is allocated and it is never used by VSAM. Moreover, an access method services request for space may fail if the DISP=NEW acquisition of space causes too little space to remain available.

Figure 23 (Part 1 of 3). JCL DD Parameters

Parameter	Subparameter	Comment
	UNCATLG	VSAM data sets are cataloged and uncataloged as a result of access method services commands; if UNCATLG is coded, a message is issued, but the data set is not uncataloged.
DSNAME	dsname(areaname)	The name is used; areaname is ignored.
	dsname(generation)	A generation data group entry must exist in the catalog for this to apply.
	dsname(member)	The name is used; member is ignored.
	All temporary dsnames	Because VSAM data sets are built by access method services, which uses the data set name supplied in the DEFINE command; temporary names cannot be used with VSAM.
	All backward DD references of the form *.ddname	If the object referred to is a cluster and the data set and index reside on unlike devices, the results of a backward DD reference are unpredictable.
LABEL	**BLP, NL, NSL**	Because these subparameters have no meaning for direct access devices, they do not apply for VSAM data sets, which all reside on direct access storage.
IN		Because IN is used to override DCB subparameters and the DCB parameter does not apply to VSAM data sets, IN does not apply.
OUT		Because OUT is used to override DCB subparameters and the DCB parameter does not apply to VSAM data sets, OUT does not apply.
LABEL	**NOPWREAD PASSWORD**	The password protection bit is set for all VSAM data sets, regardless of the PASSWORD/NOPWREAD specification in the LABEL parameter.
SL, SUL		Although these parameters apply to direct access storage devices, SL is always used for VSAM, whether you specify SL, SUL, or neither.
SEP	ddname	You must use this parameter carefully. If the cluster components, the data and its index, reside on unlike devices, the results of a SEP specification are unpredictable.
SPACE		VSAM data sets are initially allocated as a result of the access method services DEFINE command. Therefore, if SPACE is specified on the DD statement, an extent is allocated that is never used by VSAM. Moreover, an access method services request for space may fail as a result of the SPACE acquisition of space.

Figure 23 (Part 2 of 3). JCL DD Parameters

Parameter	Subparameter	Comment
SYSOUT		If SYSOUT is coded with a mutually exclusive parameter (for example, DISP), the job step is terminated with an error message.
UCS	All	Because this parameter applies only to unit record devices, it does not apply to VSAM.
UNIT AFF		You must use this subparameter carefully. If the cluster components, the data, and its index reside on unlike devices, the results of UNIT=AFF are unpredictable.
UNIT SEP		You must use this subparameter carefully. If the cluster components, the data, and its index reside on unlike devices, the results of UNIT=SEP are unpredictable.
VOLUME	**REF**	You must use this subparameter carefully. If the referenced volumes are not a subset of those contained in the catalog record for the data set, the results are unpredictable.
	volseq#	Results are unpredictable.
	volcount	This subparameter is used to request some number of nonspecific volumes. Because all VSAM volumes must be specifically defined before processing, volcount is not applicable to VSAM data sets.

Figure 23 (Part 3 of 3). JCL DD Parameters

APPENDIX E. INVOKING ACCESS METHOD SERVICES FROM A PROBLEM PROGRAM

Access method services can be invoked by a problem 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.

AUTHORIZED PROGRAM FACILITY (APF)

For information on APF authorization, see Catalog Administration Guide or VSAM Administration Guide.

INVOKING MACRO INSTRUCTIONS

The following descriptions of the invoking macro instructions are related to Figure 24 on page 322, which describes the argument lists referenced by the invoking macros.

LINK OR ATTACH MACRO INSTRUCTION

Access method services may be invoked through either the LINK or the ATTACH macro instruction.

The format of the LINK or ATTACH macro instruction is:

[name]	LINK|ATTACH	EP=IDCAMS, PARAM=(optionaddr [,dnameaddr] [,pgnoaddr] [,iolistaddr]), 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 24 on page 322 shows the format of the options list.

dnameaddr
specifies the address of a list of alternate dd names 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 may be omitted. Figure 24 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 may be omitted. If omitted, the default page number is 1. Figure 24 on page 322 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 may be omitted. Figure 24 shows the format of the I/O 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 may also be invoked via a LOAD of the module IDCAMS, followed by a CALL to that module. The format of the LOAD macro instruction is:

[*name*]	**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 'IDCAMS bb'.

After loading IDCAMS, register 15 must be loaded with the address returned from the LOAD macro. Then CALL may be used to pass control to IDCAMS. The format of the CALL macro instruction is:

[*name*]	**LR** **CALL**	**15,0** **(15),** (*optionaddr* [,*dnameaddr*] [,*pgnoaddr*] [,*iolistaddr*]), **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 24 on page 322 shows the format of the options list.

dnameaddr
specifies the address of a list of alternate dd names 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 may be omitted. Figure 24 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 may be omitted. If omitted, the default page number is 1. Figure 24 on page 322 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 may be omitted. Figure 24 shows the format of the I/O 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 may 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 branched to via a CALL statement, and deleted via a RELEASE IDCAMS statement. The format of the CALL statement is:

CALL	IDCAMS	(options[,dnames][,pageno][,iolist]);

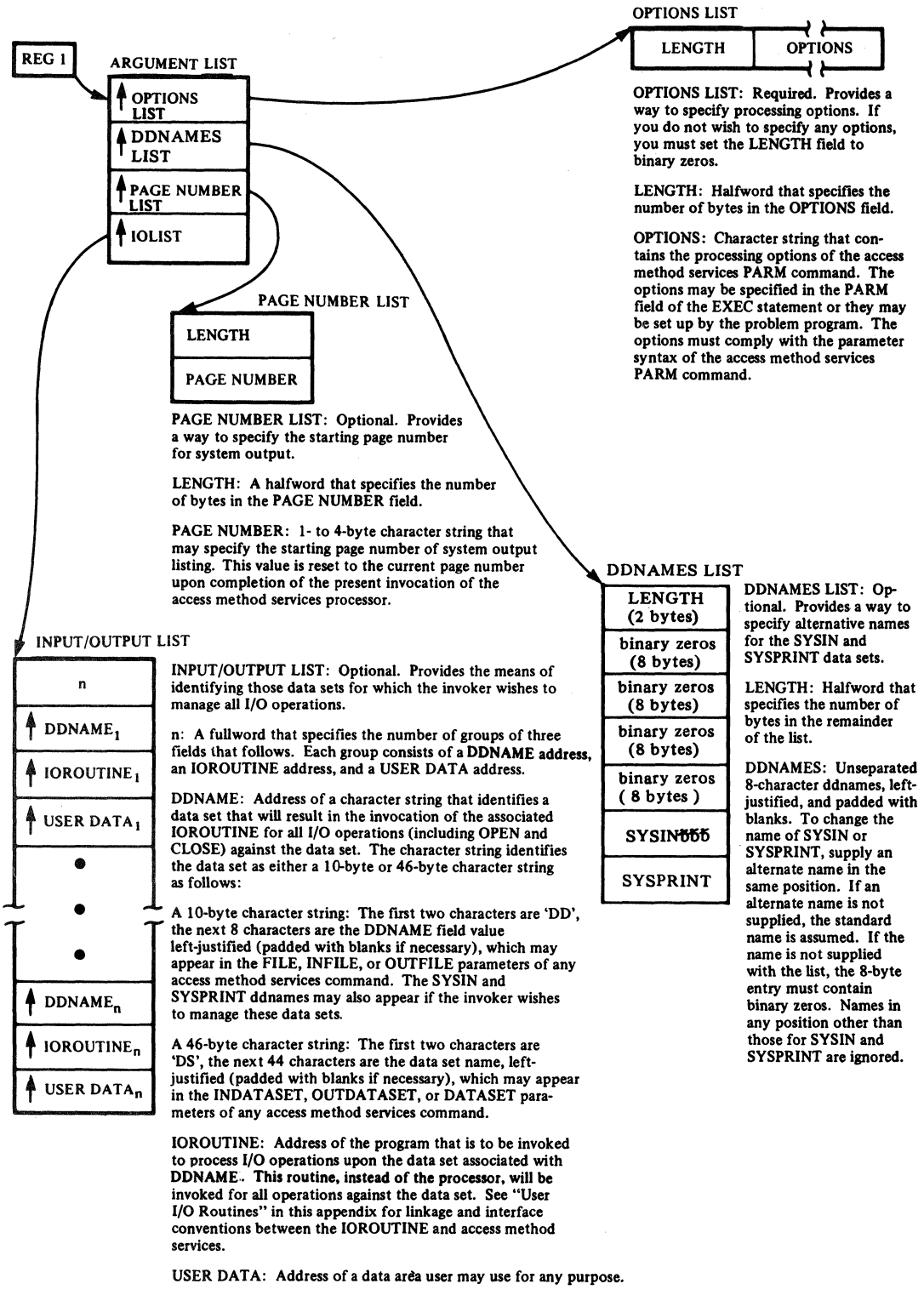
where:

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 24 on page 322 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 may be omitted. Figure 24 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 may be omitted. If not specified, the default page number is 1. Figure 24 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 may be omitted. Figure 24 shows the format of the I/O list.



OPTIONS LIST: Required. Provides a way to specify processing options. If you do not wish to specify any options, you must set the LENGTH field to binary zeros.

LENGTH: Halfword that specifies the number of bytes in the OPTIONS field.

OPTIONS: Character string that contains the processing options of the access method services PARM command. The options may be specified in the PARM field of the EXEC statement or they may be set up by the problem program. The options must comply with the parameter syntax of the access method services PARM command.

PAGE NUMBER LIST: Optional. Provides a way to specify the starting page number for system output.

LENGTH: A halfword that specifies the number of bytes in the PAGE NUMBER field.

PAGE NUMBER: 1- to 4-byte character string that may specify the starting page number of system output listing. This value is reset to the current page number upon completion of the present invocation of the access method services processor.

DDNAMES LIST: Optional. Provides a way to specify alternative names for the SYSIN and SYSPRINT data sets.

LENGTH: Halfword that specifies the number of bytes in the remainder of the list.

DDNAMES: Unseparated 8-character ddnames, left-justified, and padded with blanks. To change the name of SYSIN or SYSPRINT, supply an alternate name in the same position. If an alternate name is not supplied, the standard name is assumed. If the name is not supplied with the list, the 8-byte entry must contain binary zeros. Names in any position other than those for SYSIN and SYSPRINT are ignored.

INPUT/OUTPUT LIST: Optional. Provides the means of identifying those data sets for which the invoker wishes to manage all I/O operations.

n: A fullword that specifies the number of groups of three fields that follows. Each group consists of a DDNAME address, an IOROUTINE address, and a USER DATA address.

DDNAME: Address of a character string that identifies a data set that will result in the invocation of the associated IOROUTINE for all I/O operations (including OPEN and CLOSE) against the data set. The character string identifies the data set as either a 10-byte or 46-byte character string as follows:

A 10-byte character string: The first two characters are 'DD', the next 8 characters are the DDNAME field value left-justified (padded with blanks if necessary), which may appear in the FILE, INFILE, or OUTFILE parameters of any access method services command. The SYSIN and SYSPRINT ddnames may also appear if the invoker wishes to manage these data sets.

A 46-byte character string: The first two characters are 'DS', the next 44 characters are the data set name, left-justified (padded with blanks if necessary), which may appear in the INDATASET, OUTDATASET, or DATASET parameters of any access method services command.

IOROUTINE: Address of the program that is to be invoked to process I/O operations upon the data set associated with DDNAME. This routine, instead of the processor, will be invoked for all operations against the data set. See "User I/O Routines" in this appendix for linkage and interface conventions between the IOROUTINE and access method services.

USER DATA: Address of a data area user may use for any purpose.

Figure 24. Processor Invocation Argument List from a Problem Program

PROCESSOR INVOCATION

Figure 24 on page 322 shows the processor invocation argument list as it exists in the user's area.

Entry and exit to the access method services processor occurs through IDCSA01, 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 for IDCSA01. On exit from the access method services processor, register 15 contains the value of MAXCC (see "Processor Condition Codes" later in this appendix).

The argument list, as shown in Figure 24, can be a maximum of four 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 executed successfully. Informational messages may have been issued.
4(4)	Some minor problems in executing the complete function were encountered, but it was possible to continue. The results may 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 performed.
16(10)	Severe error or problem encountered. Remainder of command stream is flushed 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 methods services processor. This makes it possible, for instance, to control the command input stream by providing an I/O routine for SYSIN. As pointed out above, standard linkage is used.

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 via the input/output list of the processor invocation parameter list (see Figure 24 on page 322).

When writing a user I/O routine, the user must be aware of three things: First, 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. Second, the user must know the data format so that the routine can be coded to handle the correct type of input and format the correct type of output. Third, 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 operation.
8(8)	Error encountered during a GET/PUT operation, but continue processing.
12(C)	Do not allow any further calls (except CLOSE) to this routine.

Figure 25 on page 325 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.

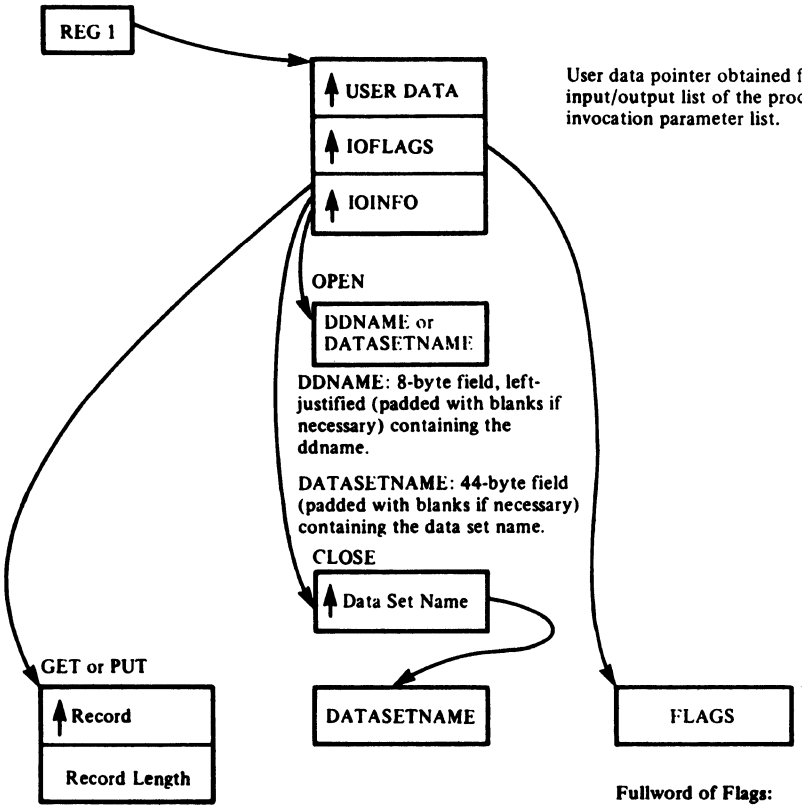
The type of operation to be performed is indicated via IOFLAGS. The IOINFO field indicates, for OPEN and CLOSE operations, the data set name or ddname of the data set; for GET and PUT operations, the IOINFO field is used to communicate the record length and address.

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 via IOFLAGS (see Figure 25 on page 325). 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 via IOFLAGS is the nnn or nnnn portion of the message converted to binary.



For a GET, this information is returned to the processor by the user's I/O routine in the 8-byte area passed 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): Fullword length of retrieved record.

Record (PUT): Address of record to be written.

Record Length (PUT): Fullword length of record to be written.

DATASETNAME: 44-byte field (padded with blanks if necessary) containing the data set name.

Fullword 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 25. Arguments Passed to and from a User I/O Routine

APPENDIX F. COMMAND REFERENCE SUMMARY

This appendix shows the format of each access method services functional command, parameter abbreviation, and default. The commands are grouped together to allow you to remove these pages and use them for a quick reference.

ALTER—MODIFY ATTRIBUTES OF PREVIOUSLY DEFINED CATALOG ENTRIES

Parameters	Abbrev.
ALTER	—
entryname[/password]	—
[ADDVOLUMES(volser[volser...])]	AVOL
[ATTEMPTS(number)]	ATT
[AUTHORIZATION(entrypoint[string])]	AUTH
[BUFFERSPACE(size)]	BUFSP, BUFSPC
[CODE(code)]	—
[CONTROLPW(password)]	CTLPW
[DESTAGEWAIT|NODESTAGEWAIT]	DSTGW NDSTGW
[EMPTY|NOEMPTY]	EMP NEMP
[ERASE|NOERASE]	ERAS NERAS
[EXCEPTIONEXIT(entrypoint)]	EEXT
[FILE(ddname)]	—
[FREESPACE(CI-percent[CA-percent])]	FSPC
[INHIBIT|UNINHIBIT]	INH UNINH
[KEYS(length offset)]	—
[MASTERPW(password)]	MRPW
[NEWNAME(newname)]	NEWNM
[NULLIFY(	NULL
[AUTHORIZATION(MODULE|STRING)]	AUTH MDLE STRG
[CODE]	—
[CONTROLPW]	CTLPW
[EXCEPTIONEXIT]	EEXT
[MASTERPW]	MRPW
[OWNER]	—
[READPW]	RDPW
[RETENTION]	RETN
[UPDATEPW])]	UPDPW
[OWNER(ownerid)]	—
[READPW(password)]	RDPW
[RECORDSIZE(average maximum)]	RECSZ
[REMOVEVOLUMES(volser[volser...])]	RVOL
[SCRATCH|NOSCRATCH]	SCR NSCR
[SHAREOPTIONS(crossregion[crosssystem])]	SHR
[STAGE|BIND|CYLINDERFAULT]	— — CYLF
[TO(date)|FOR(days)]	— —
[UNIQUEKEY|NONUNIQUEKEY]	UNQK NUNQK
[UPDATE|NOUPDATE]	UPD NUPD
[UPDATEPW(password)]	UPDPW
[UPGRADE|NOUPGRADE]	UPG NUPG
[WRITECHECK|NOWRITECHECK]	WCK NWCK
[CATALOG(catname[/password])]	CAT

BLDINDEX—BUILD ALTERNATE INDEXES FOR EXISTING VSAM CLUSTERS

Parameters	Abbrev.
BLDINDEX	BIX
{INFILE(ddname[/password])|	IFILE
INDATASET(entryname[/password])}	IDS
{OUTFILE(ddname[/password]	OFILE
[ddname[/password]...])|	
OUTDATASET(entryname[/password]	ODS
[entryname[/password]...])}	
[CATALOG(catname[/password])]	CAT
[EXTERNALSORT|INTERNALSORT]	ESORT ISORT
[WORKFILES(ddname ddname)]	WFILE

CHKLIST—IDENTIFY TAPE VOLUMES MOUNTED WHEN A CHECKPOINT WAS TAKEN

Parameters	Abbrev.
CHKLIST	CKLST
INFILE(ddname)	IFILE
[CHECKID(checkid...)]	CHKID
[OUTFILE(ddname)]	OFILE

CNVTCAT—CONVERT OS CVOL ENTRIES INTO VSAM CATALOG ENTRIES

Parameters	Abbrev.
CNVTCAT	CNVTC
{INFILE(ddname[/password])|	IFILE
[INDATASET(entryname[/password])}	IDS
{OUTFILE(ddname[/password])	OFILE
[OUTDATASET(entryname[/password])}	ODS
[CATALOG(catname[/password])]	CAT
[CVOLEQUATES((catname (volser	CVEQU
[volser...]))[(catname...)...])]	
[LIST|NOLIST]	— NLIST
[MASTERCATALOG(catname[/password])]	MCAT

DEFINE ALIAS—DEFINE AN ALTERNATE NAME FOR A USER CATALOG OR A NON-VSAM DATA SET

Parameters	Abbrev.
DEFINE	DEF
ALIAS	—
(NAME(aliasname)	—
RELATE(entryname))	REL
[CATALOG(catname[/password])]	CAT

DEFINE ALTERNATEINDEX—DEFINE A CATALOG ENTRY FOR AN ALTERNATE INDEX

Parameters	Abbrev.
DEFINE ALTERNATEINDEX	DEF AIX
(NAME(entryname)	—
RELATE(entryname[/password])	REL
{CYLINDERS[1](primary[secondary])|	CYL
RECORDS[1](primary[secondary])|	REC
TRACKS[1](primary[secondary])}	TRK
VOLUMES[2](volser[volser...])	VOL
[ATTEMPTS(number|2)]	ATT
[AUTHORIZATION(entrypoint[string])]	AUTH
[BUFFERSPACE(size)]	BUFSP, BUFSPC
[CODE(code)]	—
[CONTROLINTERVALSIZE(size)]	CISZ, CNVSZ
[CONTROLPW(password)]	CTLPW
[DESTAGEWAIT|NODESTAGEWAIT]	DSTGW NDSTGW
[ERASE|NOERASE]	ERAS NERAS
[EXCEPTIONEXIT(entrypoint)]	EEXT
[FILE(ddname)]	—
[FREESPACE(CI-percent[CA-percent]|0 0)	FSPC
[IMBED|NOIMBED]	IMBD NIMBD
[KEYRANGES((lowkey highkey)	KRNG
[(lowkey highkey)...])]	
[KEYS(length offset|64 0)]	—
[MASTERPW(password)]	MRPW
[MODEL(entryname[/password]	—
[catname[/password]])]	
[ORDERED|UNORDERED]	ORD UNORD
[OWNER(ownerid)]	—
[READPW(password)]	RDPW
[RECORDSIZE(average maximum|	RECSZ
4086 32600)]	
[REPLICATE|NOREPLICATE]	REPL NREPL
[REUSE|NOREUSE]	RUS NRUS
[SHAREOPTIONS(crossregion	SHR
[crosssystem]|1 3)]	
[SPEED|RECOVERY]	— RCVY
[STAGE|BIND|CYLINDERFAULT]	— — CYLF
[TO(date)|FOR(days)]	— —
[UNIQUE|SUBALLOCATION]	UNQ SUBAL
[UNIQUEKEY|NONUNIQUEKEY]	UNQK NUNQK
[UPDATEPW(password)]	UPDPW
[UPGRADE|NOUPGRADE]	UPG NUPG
[WRITECHECK|NOWRITECHECK])	WCK NWCK

(Continued; notes follow)

Parameters	Abbrev.
[DATA	—
([ATTEMPTS(number)]	ATT
[AUTHORIZATION(entrypoint[string])]	AUTH
[BUFFERSPACE(size)]	BUFSP, BUFSPC
[CODE(code)]	—
[CONTROLINTERVALSIZE(size)]	CISZ, CNVSZ
[CONTROLPW(password)]	CTLPW
[CYLINDERS(primary[secondary])|	CYL
RECORDS(primary[secondary])|	REC
TRACKS(primary[secondary])]	TRK
[DESTAGEWAIT|NODESTAGEWAIT]	DSTGW NDSTGW
[ERASE|NOERASE]	ERAS NERAS
[EXCEPTIONEXIT(entrypoint)]	EEXT
[FILE(ddname * *)]	—
[FREESPACE(CI-percent[CA-percent])]	FSPC
[KEYRANGES((lowkey highkey)	KRNG
[(lowkey highkey)...])]	
[KEYS(length offset)]	—
[MASTERPW(password)]	MRPW
[MODEL(entryname[/password]	—
[catname[/password]])]	
[NAME(entryname)]	—
[ORDERED|UNORDERED]	ORD UNORD
[OWNER(ownerid)]	—
[READPW(password)]	RDPW
[RECORDSIZE(average maximum)]	RECSZ
[REUSE|NOREUSE]	RUS NRUS
[SHAREOPTIONS(crossregion[crosssystem])]	SHR
[SPEED|RECOVERY]	— RCVY
[STAGE|BIND|CYLINDERFAULT]	— — CYLF
[UNIQUE|SUBALLOCATION]	UNQ SUBAL
[UNIQUEKEY|NONUNIQUEKEY]	UNQK NUNQK
[UPDATEPW(password)]	UPDPW
[VOLUMES(volser[volser...])]	VOL
[WRITECHECK|NOWRITECHECK])]	WCK NWCK

(Continued; notes follow)

Parameters	Abbrev.
[INDEX	IX
[[ATTEMPTS(number)]	ATT
[AUTHORIZATION(entrypoint[string])]	AUTH
[CODE(code)]	—
[CONTROLINTERVALSIZE(size)]	CISZ, CNVSZ
[CONTROLPW(password)]	CTLPW
[CYLINDERS(primary[secondary])|	CYL
RECORDS(primary[secondary])|	REC
TRACKS(primary[secondary])}	TRK
[DESTAGEWAIT|NODESTAGEWAIT]	DSTGW NDSTGW
[EXCEPTIONEXIT(entrypoint)]	EEXT
[FILE(ddname)]	—
[IMBED|NOIMBED]	IMBD NIMBD
[MASTERPW(password)]	MRPW
[MODEL(entryname[/password]	—
[catname[/password]])]	
[NAME(entryname)]	—
[ORDERED|UNORDERED]	ORD UNORD
[OWNER(ownerid)]	—
[READPW(password)]	RDPW
[REPLICATE|NOREPLICATE]	REPL NREPL
[REUSE|NOREUSE]	RUS NRUS
[SHAREOPTIONS(crossregion[crosssystem])]	SHR
[STAGE|BIND|CYLINDERFAULT]	— — CYLF
[UNIQUE|SUBALLOCATION]	UNQ SUBAL
[UPDATEPW(password)]	UPDPW
[VOLUMES(volser[volser...])]	VOL
[WRITECHECK|NOWRITECHECK])]	WCK NWCK
[CATALOG(catname[/password])]	CAT

Notes

1 CYLINDERS, RECORDS, or TRACKS must be specified either as a parameter of ALTERNATEINDEX, as a parameter of DATA, or as a parameter of both DATA and INDEX if MODEL is not specified.

2 VOLUMES must be specified as a parameter of ALTERNATEINDEX or as a parameter of both DATA and INDEX if MODEL is not specified.

DEFINE CLUSTER—DEFINE A CATALOG ENTRY FOR A VSAM CLUSTER

Parameters	Abbrev.
DEFINE CLUSTER	DEF CL
(NAME(entryname**)**	—
{CYLINDERS[1]**(**primary**[** secondary**])|**	CYL
RECORDS[1]**(**primary**[** secondary**])|**	REC
TRACKS[1]**(**primary**[** secondary**])}**	TRK
VOLUMES[2]**(**volser**[** volser...**])**	VOL
[ATTEMPTS(number**|2)]**	ATT
[AUTHORIZATION(entrypoint**[** string**])]**	AUTH
[BUFFERSPACE(size**)]**	BUFSP, BUFSPC
[CODE(code**)]**	—
[CONTROLINTERVALSIZE(size**)]**	CISZ, CNVSZ
[CONTROLPW(password**)]**	CTLPW
[DESTAGEWAIT|NODESTAGEWAIT]	DSTGW NDSTGW
[ERASE|NOERASE]	ERAS NERAS
[EXCEPTIONEXIT(entrypoint**)]**	EEXT
[FILE(ddname**)]**	—
[FREESPACE[3]**(**CI-percent **[** CA-percent**]|0 0)]**	FSPC
[IMBED|NOIMBED][3]	IMBD NIMBD
[INDEXED|NONINDEXED|NUMBERED]	IXD NIXD NUMD
[KEYRANGES[3]**((**lowkey highkey**) [(**lowkey highkey**)...])]**	KRNG
[KEYS[3]**(**length offset**|64 0)]**	—
[MASTERPW(password**)]**	MRPW
[MODEL(entryname**[/**password**]**	—
[catname**[/**password**]])]**	
[ORDERED|UNORDERED]	ORD UNORD
[OWNER(ownerid**)]**	—
[READPW(password**)]**	RDPW
[RECORDSIZE[4]**(**average maximum**)]**	RECSZ
[REPLICATE|NOREPLICATE][3]	REPL NREPL
[REUSE|NOREUSE]	RUS NRUS
[SHAREOPTIONS(crossregion **[** crosssystem**]|1 3)]**	SHR
[SPANNED|NONSPANNED]	SPND NSPND
[SPEED|RECOVERY]	— RCVY
[STAGE|BIND|CYLINDERFAULT]	— — — CYLF
[TO(date**)|FOR(**days**)]**	— —
[UNIQUE|SUBALLOCATION]	UNQ SUBAL
[UPDATEPW(password**)]**	UPDPW
[WRITECHECK|NOWRITECHECK])	WCK NWCK

(Continued; notes follow)

Parameters	Abbrev.
[DATA	—
([ATTEMPTS(number)]	ATT
[AUTHORIZATION(entrypoint[string])]	AUTH
[BUFFERSPACE(size)]	BUFSP, BUFSPC
[CODE(code)]	—
[CONTROLINTERVALSIZE(size)]	CISZ, CNVSZ
[CONTROLPW(password)]	CTLPW
[CYLINDERS(primary[secondary])|	CYL
RECORDS(primary[secondary])|	REC
TRACKS(primary[secondary])]	TRK
[DESTAGEWAIT|NODESTAGEWAIT]	DSTGW NDSTGW
[ERASE|NOERASE]	ERAS NERAS
[EXCEPTIONEXIT(entrypoint)]	EEXT
[FILE(ddname)]	—
[FREESPACE[3](CI-percent[CA-percent])]	FSPC
[KEYRANGES[3]((lowkey highkey)	KRNG
[(lowkey highkey)...])]	
[KEYS[3](length offset)]	—
[MASTERPW(password)]	MRPW
[MODEL(entryname[/password]	—
[catname[/password]])]	
[NAME(entryname)]	—
[ORDERED|UNORDERED]	ORD UNORD
[OWNER(ownerid)]	—
[READPW(password)]	RDPW
[RECORDSIZE(average maximum)]	RECSZ
[REUSE|NOREUSE]	RUS NRUS
[SHAREOPTIONS(crossregion[crosssystem])]	SHR
[SPANNED|NONSPANNED]	SPND NSPND
[SPEED|RECOVERY]	— RCVY
[STAGE|BIND|CYLINDERFAULT]	— — CYLF
[UNIQUE|SUBALLOCATION]	UNQ SUBAL
[UPDATEPW(password)]	UPDPW
[VOLUMES(volser[volser...])]	VOL
[WRITECHECK|NOWRITECHECK])]	WCK NWCK

(Continued; notes follow)

Parameters	Abbrev.
[INDEX[3] ([ATTEMPTS(number)] [AUTHORIZATION(entrypoint[string])] [CODE(code)] [CONTROLINTERVALSIZE(size)] [CONTROLPW(password)] [CYLINDERS(primary[secondary])| RECORDS(primary[secondary])| TRACKS(primary[secondary])] [DESTAGEWAIT|NODESTAGEWAIT] [EXCEPTIONEXIT(entrypoint)] [FILE(ddname)] [IMBED|NOIMBED][3] [MASTERPW(password)] [MODEL(entryname[/password] [catname[/password]])] [NAME(entryname)] [ORDERED|UNORDERED] [OWNER(ownerid)] [READPW(password)] [REPLICATE|NOREPLICATE][3] [REUSE|NOREUSE] [SHAREOPTIONS(crossregion[crosssystem])] [STAGE|BIND|CYLINDERFAULT] [UNIQUE|SUBALLOCATION] [UPDATEPW(password)] [VOLUMES(volser[volser...])] [WRITECHECK|NOWRITECHECK])] [CATALOG(catname[/password])]	IX ATT AUTH — CISZ,CNVSZ CTLPW CYL REC TRK DSTGW NDSTGW EEXT — IMBD NIMBD MRPW — — ORD UNORD — RDPW REPL NREPL RUS NRUS SHR — — CYLF UNQ SUBAL UPDPW VOL WCK NWCK CAT

Notes

1 CYLINDERS, RECORDS, or TRACKS must be specified either as a parameter of CLUSTER, as a parameter of DATA, or as a parameter of both DATA and INDEX if MODEL is not specified.

2 VOLUMES must be specified as a parameter of CLUSTER or as a parameter of both DATA and INDEX if MODEL is not specified.

3 Can only be specified for a key-sequenced cluster.

4 When SPANNED is specified, the default is RECORDSIZE (4086 32600). Otherwise, the default is RECORDSIZE (4089 4089).

DEFINE GENERATIONDATAGROUP—CREATE A CATALOG ENTRY FOR A GENERATION DATA GROUP

Parameters	Abbrev.
DEFINE GENERATIONDATAGROUP	DEF GDG
(NAME(entryname)	—
LIMIT(limit)	LIM
[EMPTY|NOEMPTY]	EMP NEMP
[OWNER(ownerid)]	—
[SCRATCH|NOSCRATCH]	SCR NSCR
[TO(date)|FOR(days)])	—
[CATALOG(catname[/password])]	CAT

DEFINE NONVSAM—DEFINE A CATALOG ENTRY FOR A NON-VSAM DATA SET

Parameters	Abbrev.
DEFINE NONVSAM	DEF NVSAM
(NAME(entryname)	—
DEVICETYPES(devtype[devtype...])	DEVT
VOLUMES(volser[volser...])	VOL
[FILESEQUENCENUMBERS(number	FSEQN
[number...])]	
[OWNER(ownerid)]	—
[TO(date)|FOR(days)])	— —
[CATALOG(catname[/password])]	CAT

DEFINE PAGESPACE—DEFINE A CATALOG ENTRY FOR A PAGE SPACE

Parameters	Abbrev.
DEFINE PAGESPACE	DEF PGSPC
(NAME(entryname)	—
{CYLINDERS(primary)|	CYL
RECORDS(primary)|	REC
TRACKS(primary)}	TRK
VOLUME(volser)	VOL
[ATTEMPTS(number|2)]	ATT
[AUTHORIZATION(entrypoint[string])]	AUTH
[CODE(code)]	—
[CONTROLPW(password)]	CTLPW
[FILE(ddname)]	—
[MASTERPW(password)]	MRPW
[MODEL(entryname[/password]	—
[catname[/password]])]	
[OWNER(ownerid)]	—
[READPW(password)]	RDPW
[SWAP|NOSWAP]	— NSWAP
[TO(date)|FOR(days)]	— —
[UNIQUE|SUBALLOCATION]	UNQ SUBAL
[UPDATEPW(password)])	UPDPW
[CATALOG(catname[/password])]	CAT

DEFINE PATH—DEFINE A RELATIONSHIP BETWEEN AN ALTERNATE INDEX AND ITS BASE CLUSTER

Parameters	Abbrev.
DEFINE PATH	DEF —
(NAME(entryname)	—
PATHENTRY(entryname[/password])	PENT
[ATTEMPTS(number|2)]	ATT
[AUTHORIZATION(entrypoint[string])]	AUTH
[CODE(code)]	—
[CONTROLPW(password)]	CTLPW
[FILE(ddname)]	—
[MASTERPW(password)]	MRPW
[MODEL(entryname[/password]	—
[catname[/password]])]	
[OWNER(ownerid)]	—
[READPW(password)]	RDPW
[TO(date)|FOR(days)]	— —
[UPDATE|NOUPDATE]	UPD NUPD
[UPDATEPW(password)])	UPDPW
[CATALOG(catname[/password])]	CAT

DEFINE SPACE—DEFINE A VSAM DATA SPACE IN A VSAM CATALOG

Parameters	Abbrev.
DEFINE SPACE	DEF SPC
({CANDIDATE|	CAN
CYLINDERS(primary[secondary])|	CYL
RECORDS(primary[secondary])	REC
RECORDSIZE(average maximum)|	RECSZ
TRACKS(primary[secondary])}	TRK
VOLUMES(volser[volser...])	VOL
[FILE(ddname)])	—
[CATALOG(catname[/password])]	CAT

Parameters	Abbrev.
DEFINE USERCATALOG|MASTERCATALOG	DEF UCAT MCAT
[NAME(entryname)]	—
{CYLINDERS(primary[secondary])|	CYL
RECORDS(primary[secondary])|	REC
TRACKS(primary[secondary])}	TRK
VOLUME(volser)	VOL
[ATTEMPTS(number|2)]	ATT
[AUTHORIZATION(entrypoint[string])]	AUTH
[BUFFERSPACE(size|3072)]	BUFSP, BUFSPC
[CODE(code)]	—
[CONTROLPW(password)]	CTLPW
[DESTAGEWAIT|NODESTAGEWAIT]	DSTGW NDSTGW
[FILE(ddname)]	—
[MASTERPW(password)]	MRPW
[MODEL(entryname[/password]	—
[catname[/password]])]	
[OWNER(ownerid)]	—
[READPW(password)]	RDPW
[RECORDSIZE(average maximum)]	RECSZ
[RECOVERABLE|NOTRECOVERABLE]	RVBL NRVBL
[TO(date)|FOR(days)]	— —
[UPDATEPW(password)]	UPDPW
[WRITECHECK|NOWRITECHECK])	WCK NWCK
[DATA	—
([BUFFERSPACE(size)]	BUFSP, BUFSPC
[CYLINDERS(primary[secondary])|	CYL
RECORDS(primary[secondary])|	REC
TRACKS(primary[secondary])]	TRK
[DESTAGEWAIT|NODESTAGEWAIT]	DSTGW NDSTGW
[RECORDSIZE(average maximum)]	RECSZ
[RECOVERABLE|NOTRECOVERABLE]	RVBL NRVBL
[WRITECHECK|NOWRITECHECK])]	WCK NWCK
[INDEX	IX
([CYLINDERS(primary)|	CYL
RECORDS(primary)|	REC
TRACKS(primary)]	TRK
[DESTAGEWAIT|NODESTAGEWAIT]	DSTGW NDSTGW
[WRITECHECK|NOWRITECHECK])]	WCK NWCK
[CATALOG(mastercatname[/password])]	CAT

DELETE—DELETE ENTRIES FROM A CATALOG

Parameters	Abbrev.
DELETE	DEL
(entryname[/password]	—
[entryname[/password]...])	
[ALIAS|	—
ALTERNATEINDEX|	AIX
CLUSTER|	CL
GENERATIONDATAGROUP|	GDG
NONVSAM|	NVSAM
PAGESPACE|	PGSPC
PATH|	—
SPACE[1]|	SPC
USERCATALOG[1]	UCAT
[ERASE|NOERASE]	ERAS NERAS
[FILE(ddname)]	—
[FORCE|NOFORCE]	FRC NFRC
[PURGE|NOPURGE]	PRG NPRG
[SCRATCH|NOSCRATCH]	SCR NSCR
[CATALOG(catname[/password])]	CAT

Notes

1 When you delete a data space or catalog, you cannot delete any other type of entry. You must identify the type of entry to be deleted, by specifying SPACE or USERCATALOG.

EXAMINE—EXAMINING KEY-SEQUENCED DATA SET CLUSTERS

Parameters	Abbrev.
EXAMINE	—
NAME(clustername[/password])	— — —
[INDEXTEST|NOINDEXTEST]	ITEST NOITEST
[DATATEST|NODATATEST]	DTEST NODTEST
[ERRORLIMIT(value)]	ELIMIT

EXPORT DISCONNECT—DISCONNECT A USER CATALOG

Parameters	Abbrev.
EXPORT	EXP
usercatname[/password]	—
DISCONNECT	DCON

EXPORT—CREATE A BACKUP OR PORTABLE COPY OF A VSAM CLUSTER OR ALTERNATE INDEX

Parameters	Abbrev.
EXPORT	EXP
entryname[/password]	—
{OUTFILE(ddname)|OUTDATASET(entryname)}	OFILE ODS
[INFILE(ddname)]	IFILE
[ERASE|NOERASE]	ERAS NERAS
[INHIBITSOURCE|NOINHIBITSOURCE]	INHS NINHS
[INHIBITTARGET|NOINHIBITTARGET]	INHT NINHT
[PURGE|NOPURGE]	PRG NPRG
[TEMPORARY|PERMANENT]	TEMP PERM

EXPORTRA—COPY CATALOG ENTRIES FROM THE CATALOG RECOVERY AREA (CRA) FOR VSAM RECOVERABLE CATALOGS

Parameters	Abbrev.
EXPORTRA	XPRA
OUTFILE(ddname)	OFILE
CRA(	—
[ddname1	
{ALL[INFILE(ddname2)]|	— IFILE
ENTRIES(	ENT
(entryname[ddname3])	
[(entryname[ddname3])...])|	
NONE})	—
[(ddname1...)...])	
[FORCE|NOFORCE]	FRC NFRC
[MASTERPW(password)]	MRPW

IMPORT CONNECT—CONNECT A USER CATALOG

Parameters	Abbrev.
IMPORT	IMP
CONNECT	CON
OBJECTS((usercatname	OBJ
DEVICETYPE(devtype)	DEVT
VOLUMES(volser)))	VOL
[CATALOG(mastercatname[/password])]	CAT

IMPORT—RESTORE A VSAM CLUSTER OR ALTERNATE INDEX FROM A PORTABLE DATA SET CREATED BY THE EXPORT COMMAND

Parameters	Abbrev.
IMPORT	IMP
{INFILE(ddname)|INDATASET(entryname)}	IFILE IDS
{OUTFILE(ddname[/password])|	OFILE
OUTDATASET(entryname[/password])}	ODS
[ERASE|NOERASE]	ERAS NERAS
[INTOEMPTY]	IEMPTY
[OBJECTS((entryname	OBJ
[FILE(ddname)]	—
[KEYRANGES((lowkey highkey)	KRNG
[(lowkey highkey)...])]	
[NEWNAME(newname)]	NEWNM
[ORDERED|UNORDERED]	ORD UNORD
[VOLUMES(volser[volser...])])	VOL
[(entryname...)...])]	
[PURGE|NOPURGE]	PRG NPRG
[SAVRAC|NOSAVRAC]	— —
[CATALOG(catname[/password])]	CAT

IMPORTRA—RESTORE CATALOG ENTRIES FROM A PORTABLE DATA SET CREATED BY THE EXPORTRA COMMAND FOR VSAM RECOVERABLE CATALOGS

Parameters	Abbrev.
IMPORTRA	MPRA
{INFILE(ddname)|INDATASET(entryname)}	IFILE IDS
[OUTFILE(ddname)]	OFILE
[OBJECTS(	OBJ
(entryname	
[DEVICETYPES(devtype[devtype...])]	DEVT
[FILE(ddname)]	—
[VOLUMES(volser[volser...])])	VOL
[(entryname...)...])]	
[SAVRAC|NOSAVRAC]	— —
[CATALOG(catname[/password])]	CAT

LISTCAT—LIST ENTRIES FROM A CATALOG

Parameters	Abbrev.
LISTCAT	LISTC
[ENTRIES(entryname[/password]	ENT
[entryname[/password]...])|	
LEVEL(level)]	LVL
[ALIAS]	—
[ALTERNATEINDEX]	AIX
[CLUSTER]	CL
[DATA]	—
[GENERATIONDATAGROUP]	GDG
[INDEX]	IX
[NONVSAM]	NVSAM
[PAGESPACE]	PGSPC
[PATH]	—
[SPACE]	SPC
[USERCATALOG]	UCAT
[CREATION(days)]	CREAT
[EXPIRATION(days)]	EXPIR
[NAME|	—
HISTORY|	HIST
VOLUME|	VOL
ALLOCATION|	ALLOC
ALL]	—
[NOTUSABLE]	NUS
[OUTFILE(ddname)]	OFILE
[CATALOG(catname[/password])]	CAT

LISTCRA—LIST THE CONTENTS OF THE CATALOG RECOVERY AREA (CRA) FOR VSAM RECOVERABLE CATALOGS

Parameters	Abbrev.
LISTCRA	LISTR
INFILE(ddname[ddname...])	IFILE
[CATALOG(catname[/password] ddname)]	CAT
[COMPARE|NOCOMPARE]	CMPR NCMPR
[DUMP|NAME|SEQUENTIALDUMP]	— — SDUMP
[MASTERPW(password)]	MRPW
[OUTFILE(ddname)]	OFILE

PRINT—PRINT THE CONTENTS OF A DATA SET

Parameters	Abbrev.
PRINT	
{INFILE(ddname[/password])|	IFILE
INDATASET(entryname[/password])}	IDS
[OUTFILE(ddname)]	OFILE
[CHARACTER|DUMP|HEX]	CHAR — —
[FROMKEY(key)|FROMADDRESS(address)|	FKEY FADDR
FROMNUMBER(number)|SKIP(count)]	FNUM —
[TOKEY(key)|TOADDRESS(address)|	— TADDR
TONUMBER(number)|COUNT(count)]	TNUM —

REPRO—COPY DATA SETS, COPY INTEGRATED CATALOG FACILITY

catalogs and split or merge integrated catalog facility catalogs

Parameters	Abbrev.
REPRO	—
{INFILE(ddname[/password]	IFILE
[ENVIRONMENT(DUMMY)])|	ENV DUM
INDATASET(entryname[/password]	IDS
[ENVIRONMENT(DUMMY)])}	ENV DUM
{OUTFILE(ddname[/password])|	OFILE
OUTDATASET(entryname[/password])}	ODS
[DECIPHER	DECPHR
({DATAKEYFILE(ddname)|	DKFILE
DATAKEYVALUE(value)|SYSTEMKEY]	DKV SYSKEY
[SYSTEMDATAKEY(value)]	SYSDK
[SYSTEMKEYNAME(keyname)])]	SYSKN
[ENCIPHER	ENCPHR
({EXTERNALKEYNAME(keyname)|	EKN
INTERNALKEYNAME(keyname)|PRIVATEKEY}	IKN PRIKEY
[CIPHERUNIT(number|1)]	CPHRUN
[DATAKEYFILE(ddname)|DATAKEYVALUE(value)]	DKFILE DKV
[SHIPKEYNAMES(keyname[keyname...])]	SHIPKN
[STOREDATAKEY|NOSTOREDATAKEY]	STRDK NSTRDK
[STOREKEYNAME(keyname)]	STRKN
[USERDATA(value)])]	UDATA
[FROMKEY(key)|FROMADDRESS(address)|	FKEY FADDR
FROMNUMBER(number)|SKIP(count)]	FNUM —
[REPLACE|NOREPLACE]	REP NREP
[REUSE|NOREUSE]	RUS NRUS
[TOKEY(key)|TOADDRESS(address)|	— TADDR
TONUMBER(number)|COUNT(count)]	TNUM —

Notes: The ENCIPHER and DECIPHER parameters apply only with the Programmed Cryptographic Facility (5740-XY5) or the IBM Cryptographic Unit Support (5740-XY6). (For more information, see VSAM Administration Guide.)

RESETCAT—SYNCHRONIZE A VSAM RECOVERABLE CATALOG TO THE LEVEL OF ITS OWNED VOLUMES

Parameters	Abbrev.
RESETCAT	RCAT
CATALOG(catname[/password][ddname])	CAT
{CRAFILES((ddname1[ALL| NONE])	— — —
[(ddname1[ALL| NONE])...])|	— —
CRAVOLUMES((volser[devtype])	CRAVOL
[(volser[devtype])...])}	
[IGNORE|NOIGNORE]	IGN NIGN
[MASTERPW(password)]	MRPW
[WORKCAT(catname[/password])]	WCAT
[WORKFILE(ddname[/password])]	WFILE

VERIFY—RESTORE A VSAM CLUSTER'S END-OF-FILE VALUES

Parameters	Abbrev.
VERIFY	VFY
{FILE(ddname[/password])|	—
DATASET(entryname[/password])}	DS

IF-THEN-ELSE COMMAND SEQUENCE

Parameters	
IF {LASTCC|MAXCC} comparand number THEN[command| DO command set END] [ELSE[command| DO command set END]]	

PARM COMMAND

Parameters	
PARM [TEST({[TRACE] [AREAS(areaid[areaid...])] [FULL((dumpid[count1[count2]]) [(dumpid...)...])]| OFF})] [GRAPHICS(CHAIN(chain)|TABLE(mname))] [MARGINS(leftmargin rightmargin)]	

SET COMMAND

Parameters	
SET {MAXCC|LASTCC} = number	

GLOSSARY

The following terms are defined as they are used in this book. If you do not find the term you are looking for, see the index or the IBM Dictionary of Computing, SC20-1699.

access method services. A multifunction service program that defines VSAM data sets and allocates space for them, converts indexed-sequential data sets to key-sequenced data sets with indexes, modifies data-set attributes in the catalog, reorganizes data sets, facilitates data portability between operating systems, creates backup copies of data sets and indexes, helps make inaccessible data sets accessible, and lists the records of data sets and catalogs.

alias. An alternative name for an entry.

alias entry. An entry that relates an alias (alternate entryname) to the real entry name of a user catalog or non-VSAM data set.

alternate-index entry. A catalog entry that contains information about an alternate index. An alternate index is conceptually a key-sequenced cluster, and is cataloged in the same way. 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. 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, each data record's prime key value is the pointer. When the alternate index supports an entry-sequenced data set, the data record's RBA value 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. (See also key and key field.)

application. As used in this publication, 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.

backup data set. A copy that can be used to replace or reconstruct a damaged data set.

base cluster. The VSAM cluster whose data records are to be accessed through a path. Usually, a base cluster is the key-sequenced or entry-sequenced data set which an alternate index supports (that is, an alternate index is used by VSAM to sequence and locate the data records of a base cluster). (See also alternate-index entry and path.)

catalog. (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. 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.

cell. An occurrence of information such as passwords, volume information, or associations.

cluster. A data component and an index component when data is key sequenced; a data component alone when data is entry sequenced.

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 a data entry and an index entry. An entry-sequenced cluster entry points to a data entry.

component. The data portion or, for a key-sequenced cluster, alternate index, or VSAM catalog, the index portion of a VSAM object. 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.

control area. 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. A fixed-length area of auxiliary-storage space in which VSAM stores records and distributes free space. It is the unit of information transmitted to or from auxiliary storage by VSAM.

CRA. (See catalog recovery area.)

data component. That 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 integrity. Preservation of data or programs for their intended purpose. As used in this publication, the safety of data from inadvertent destruction or alteration.

data record. A collection of items of information from the standpoint of its use in an application, as a user supplies it to VSAM for storage.

data security. Prevention of access to or use of data or programs without authorization. As used in this publication, the safety of data from unauthorized use, theft, or purposeful destruction.

data set. The major unit of data storage and retrieval in the operating system, consisting of data in a prescribed arrangement and described by control information to which the system has access. As used in this publication, a collection of fixed- or variable-length records in auxiliary storage, arranged by VSAM in key sequence or in entry sequence. (See also key-sequenced data set and entry-sequenced data set.)

data set control block. A data set label for a data set in direct access storage.

data space. A storage area defined in the volume table of contents of a direct-access volume for the exclusive use of VSAM to store data sets, indexes, and catalogs.

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.)

DSCB. (See data set control block.)

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.

entry. A collection of information about a cataloged object in a VSAM master or user catalog. Each entry resides in one or more 512-byte record.

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 with key.)

entry-sequenced data set. 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.

esoteric name. An arbitrary name provided by the user to identify a group of devices having similar characteristics (for example, DISK, TAPE, SYSDA).

extent. A continuous space allocated on a direct-access storage volume, reserved for a particular data space or data set. An extent of a data set contains a whole number of control areas.

field. In a record or a control block, a specified area used for a particular category of data or control information.

GDG. (See generation data group base entry.)

GDS. (See generation data set.)

generation data group base entry. An entry that permits non-VSAM data sets to be associated with other non-VSAM data sets as generation data sets.

generation data set. One of a collection of historically related non-VSAM data sets; the collection of these data sets is known as a generation data group.

index. As used in this publication, an ordered collection of pairs, each consisting of a key and a pointer, used by VSAM to sequence and locate the records of a key-sequenced data set; organized in levels of index records. (See also index level, index set, and sequence set.)

index component. That part of a key-sequenced data set, catalog, or alternate index, that establishes the sequence of the data records within the object it indexes. The index is used to locate each record in the object's data component, based on the record's key value.

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 set and the sequence set together comprise the index.

integrity. (See data integrity.)

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.

key-sequenced data set. A 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 by means of distributed free space. Relative byte addresses of records can change.

master catalog. A key-sequenced data set with an index containing 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 to accumulate usage statistics for data sets.

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).

object. A logical entity created by VSAM, such as a cluster (VSAM data set) and its components, an alternate index and its components, a VSAM catalog and its components, a path, or a VSAM data space.

page space. A VS2 system data set. A page space is cataloged as an entry-sequenced cluster (that is, the page space entry is similar to a cluster entry, and it points to a data entry).

password. A unique string of characters stored in a catalog that a program or a computer operator at the console must supply to meet security requirements before the program gains access to a data set.

path. A data set name for the combination of an alternate index and its base cluster, or an alias for a VSAM data set.

path entry. A catalog entry that contains information about a path, and that points to the path's related objects.

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.

portability. The ability to use VSAM data sets with different operating systems. Volumes whose data sets are cataloged in a user catalog can be demounted 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.

primary space allocation. Initially allocated space on a direct-access storage device, occupied by or reserved for a particular data set. (See also secondary space allocation.)

prime index. The index component of a key-sequenced data set. (See also index and alternate index.)

prime key. (See key.)

qualified name. A name that is segmented by periods; each name segment is referred to as a qualifier.

RBA. (See relative byte address.)

record. (See index record, data record, stored record.)

recoverable catalog. A catalog defined with the recoverable attribute. Duplicate catalog entries are put into CRAs that can be used to recover data in the event of catalog failure. (See also CRA.)

recovery volume. The first volume of a prime index, if the VSAM data set is a key-sequenced cluster; otherwise, the first volume of the data set, if entry-sequenced.

relative byte address. 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.

secondary space allocation. A contiguous space on a direct-access device, occupied by or reserved for a particular data set, which is allocated after space in the primary extent has been exhausted. (See also 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.

spanned record. A logical record whose length exceeds control interval length, and crosses (or spans) one or more control interval boundaries within a control area.

sphere record. A collection of logically related subrecords in one VSAM logical record.

step catalog. (See job catalog.)

stored record. A data record, together with its control information, as stored in auxiliary storage.

subrecord. The user definition level of a sphere, such as an AIX, cluster, or generation data set.

user catalog. A catalog used in the same way as the master catalog, but optional and pointed to by the master catalog, and also used to lessen the contention for the master catalog and to facilitate volume portability.

user catalog connector. (See catalog connector.) Interval in the data set controlled by the index.

volume cleanup. The process of deleting all VSAM data spaces from a volume and removing a VSAM catalog's ownership of the volume, even though the volume contains VSAM data records.

volume entry. A catalog entry that describes a volume owned by the catalog. All VSAM data spaces on the volume are described in the volume entry, and the catalog is no longer available.

volume table of contents. A table on a direct access volume that describes each data set on the volume.

VTOC. (See volume table of contents.)

A

B

C

D

E

F

G

H

I

J

K

O

P

R

MVS/XA
VSAM Catalog Administration:
AMS Reference

Reader's Comment Form

GC26-4136-3

This manual is part of a library that serves as a reference source for system analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: Do not use this form to request IBM publications. If you do, your order will be delayed because publications are not stocked at the address printed on the reverse side. Instead, you should direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

If you have applied any technical newsletters (TNLs) to this book, please list them here: _____

Chapter/Section _____

_____ Page No. _____

Comments:

Note: Staples can cause problems with automatic mail-sorting equipment. Please use pressure-sensitive or other gummed tape to seal this form.

If you want a reply, please complete the following information.

Name _____ Phone No. (_____) _____

Company _____

Address _____

_____

Thank you for your cooperation. No postage is necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail them directly to the address in the Edition Notice on the back of the title page.)

GC26-4136-3

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, NY

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Programming Publishing
P.O. Box 49023
San Jose, CA 95161-9023

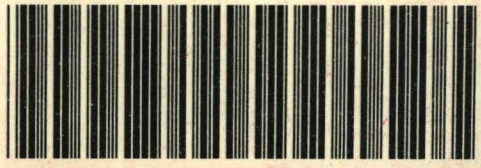
Fold and tape | Please do not staple | Fold and tape

IBM®

MVS/Extended Architecture
VSAM Catalog Administration
Access Method Services
Reference

File Number S370-34

GC26-4136-03



Printed in U.S.A.