StorageTek®

PART NUMBER
312614701

VERSION NUMBER
6.1

REVISION A

VTCS

VIRTUAL TAPE CONTROL SYSTEM (MSP IMPLEMENTATION)

COMMAND AND UTILITY REFERENCE

PRODUCT TYPE
SOFTWARE

Virtual Tape Control System

Command and Utility Reference

Version 6.1

MSP

312614701

Revision A

-

We welcome your feedback. Please contact the Global Learning Solutions Feedback System at:

GLSFS@Stortek.com
or
Global Learning Solutions
Sun Microsystems Inc.
One StorageTek Drive
Louisville, CO 80028-3256
USA

About this Book

Virtual Tape Control System 6.1.0 (VTCS 6.1.0, hereafter referred to as "VTCS") is Revision A host software, which together the portions of NCS 6.1.0 that support VTCS and the Virtual Tape Storage Subsystem (VTSS), comprises Virtual Storage Manager (VSM).

Audience

This reference provides VTCS and NCS reference information for StorageTek or customer personnel who are responsible for all VTCS tasks. Also see the following:

- *VTCS Installation and Configuration Guide* for information about installing and configuring VTCS.
- *VTCS Administrator's Guide* for information about VTCS administration tasks.

Reader's Comments

If you have comments on this book, please e–mail us at `glsfs@stortek.com` and include the document title and number with your comments.

Prerequisites

To perform the tasks described in this guide, you should already understand the following:

To perform the tasks described in this guide, you should already understand the following:

- MSP/EX operating system
- JES
- System Management Facility (SMF)
- System Modification Program Extended (SMP)
- Nearline Control Solution (NCS)
- VTCS and VSM

About the Software

This guide applies to VTCS/MSP 6.1.0 and NCS/MSP 6.1.0 and above. VTCS executes in the native MSP/EX environment.

How this Guide is Organized

This guide contains the following sections:

- Chapter 1 “VTCS Utilities and Commands”
- Chapter 3 “HSC Enhancements and Additions for VSM”
- Chapter 4 “LibraryStation Enhancements and Additions for VSM”
- Chapter 5 “MVS/CSC Enhancements and Additions for VSM”
- Appendix A “VTCS SMF Record Format”
- Appendix B “VTD Command Reference”
- Appendix C “NCS/VTCS Alphabetic Volsers”
- Appendix D “Using the HSC Significant Event Notification Facility”
- “Glossary”
- “Index”

What's New in This Reference?

VTCS 6.1, Revision A

The VTCS 6.1, Revision A of this reference contains information about the VTCS 6.1 enhancements described in Table 1.

Table 1. VTCS 6.1.0 Updates to VTCS Command and Utility Reference, Revision A

This Enhancement...	...is described in...
Bi-Directional Clustering	• "CLUSTER Statement" on page 25 • "CLINK Statement" on page 26 • "DECOM Output" on page 43 • "Display CLUSTER Output" on page 79 • "VARY CLINK" on page 159 • "STORCLAS Control Statement" on page 219
Near Continuous Operations (NCO)	• "CONFIG" on page 13 • "Display RTD Output" on page 55 • "Display CONFIG Output" on page 71
SMC 6.1 changes	"SMC Enhancements and Additions for VSM" on page 183
HSC SEN Facility	"Using the HSC Significant Event Notification Facility" on page 321
Media Management enhancements	• "AUDIT" on page 5 • "MVCDRAIN" on page 104
SMF record updates	• "SLSSMF13 - VTCS SMF Subtype 13 Record" on page 264 • "SLSSMF14 - VTCS SMF Subtype 14 Record" on page 266

Related Publications

The following publications provide additional information about VSM and StorageTek's Automated Cartridge System software and hardware.

VTCS and VSM for MSP

The VTCS and VSM documentation set consists of the following:

- *Introduction to VSM*, which you can request from your StorageTek representative
- *Virtual Tape Control System Installation and Configuration Guide*
- *Virtual Tape Control System Administrator's Guide*
- *Virtual Tape Control System Command and Utility Reference*
- *Virtual Tape Control System Messages*
- *Virtual Tape Control System XML Reference*

VTSS

- *Virtual Storage Manager Planning, Implementation, and Usage Guide*
- *Virtual Storage Manager Physical Planning Guide*
- *VTSS Installation Guide*

HSC-MSP Environment

- *Configuration Guide*
- *Operator's Guide*
- *System Programmer's Guide*
- *Messages and Codes*
- *System Programmer's Reference Summary*
- *Operator's Reference Summary*

ExPR

- *Introduction to ExPR*
- *ExPR SMP Installation*
- *ExPR MSP Configuration*
- *ExPR MSP Reports*
- *ExPR MSP Reference*

ExLM

The ExLM documentation set consists of the following:

- The ExLM 5.1.0 Information CD-ROM, which contains PDF file formats of the ExLM publications
- *ExLM Installation Guide*
- *ExLM System Administrator's Guide*
- *ExLM Messages and Codes*
- *ExLM Quick Reference* (includes information formerly provided in the *ExLM 4.0.0 System Administrator's Guide - Field Tables Supplement*)

IBM Publications

- *IBM ESA/390 Common I/O-Device Commands and Self Description*
- *IBM 3490 Magnetic Tape Subsystem Models A01, A02, A10, A20, B02, B04, B20, and B40 Introduction*
- *IBM 3490 Magnetic Tape Subsystem Models A01, A02, A10, A20, B02, B04, B20, and B40 Hardware Reference*
(Referred to in this book as the *IBM 3490 Hardware Reference*)
- *IBM 3490 Command Reference*
- *IBM 3480 Magnetic Tape Subsystem Reference*
- *IBM 3480 Installation Guide and Reference*

Conventions for Reader Usability

Conventions are used to shorten and clarify explanations and examples within this book.

Typographic

The following typographical conventions are used in this book:

- **Bold** is used to introduce new or unfamiliar terminology.
- `Letter Gothic` is used to indicate command names, filenames, and literal output by the computer.
- **`Letter Gothic Bold`** is used to indicate literal input to the computer.
- *`Letter Gothic Italic`* is used to indicate that you must substitute the actual value for a command parameter. In the following example, you would substitute your name for the "username" parameter.
- Logon *username*
- A bar (|) is used to separate alternative parameter values. In the example shown below either username or systemname must be entered.
- Logon *username*|*systemname*
- Brackets [ ] are used to indicate that a command parameter is optional.
- Ellipses (...) are used to indicate that a command may be repeated multiple times.
- The use of mixed upper and lower case characters (for non–case sensitive commands) indicates that lower case letters may be omitted to form abbreviations. For example, you may simply enter **Q** when executing the **Quit** command.

Keys

Single keystrokes are represented by double brackets [[ ]] surrounding the key name. For example, press [[ESC]] indicates that you should press only the escape key.

Combined keystrokes use double brackets and the plus sign (+). The double brackets surround the key names and the plus sign is used to add the second keystroke. For example, press [[AL]] + [[C]] indicates that you should press the alternate key and the C key simultaneously.

Enter Command

The instruction to "press the [[ENTER]] key" is omitted from most examples, definitions, and explanations in this book.

For example, if the instructions asked you to "enter" **Logon pat**, you would type in **Logon pat** *and* press lENTERm.

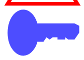
However, if the instructions asked you to "type" **Logon pat**, you would type in **Logon pat** and you would *not* press [[ENTER]].

Symbols

The following symbols are used to highlight text in this book.

Warning: Information necessary to keep you from damaging your hardware or software.

Caution: Information necessary to keep you from corrupting your data.



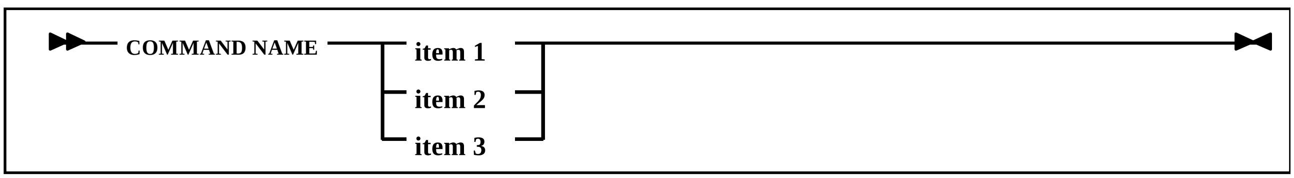
Hint: Information that can be used to shorten or simplify your task or they may simply be used as a reminder.

Note: Information that may be of special interest to you. Notes are also used to point out exceptions to rules or procedures.

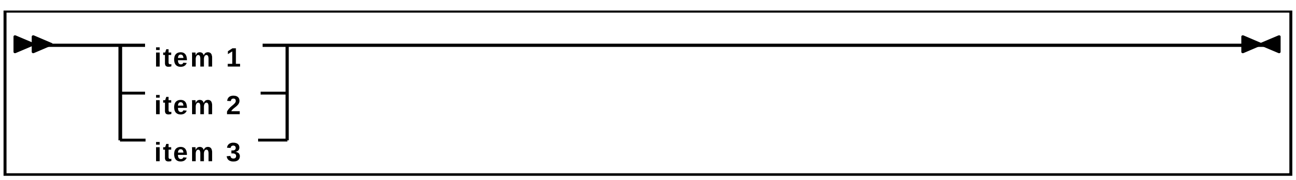
Syntax

Syntax flow diagram conventions include the following:

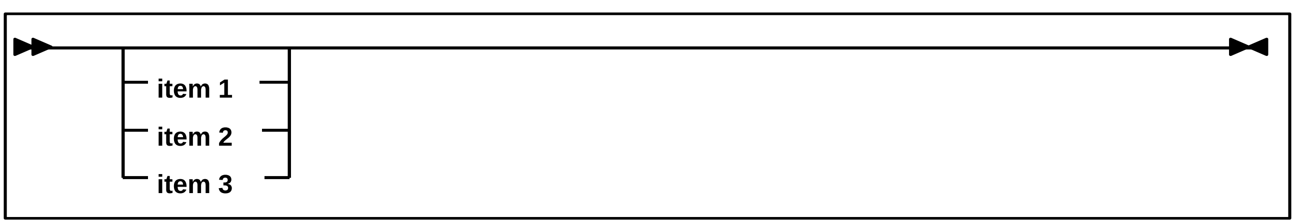
Flow Lines—Syntax diagrams consist of a horizontal baseline, horizontal and vertical branch lines and the command text. Diagrams are read left to right and top to bottom. Arrows show flow and direction.



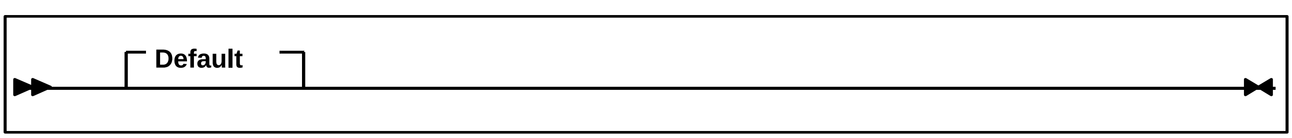
Single Required Choice—Branch lines (without repeat arrows) indicate that a single choice must be made. If one of the items to choose from is on the baseline of the diagram, one item must be selected.



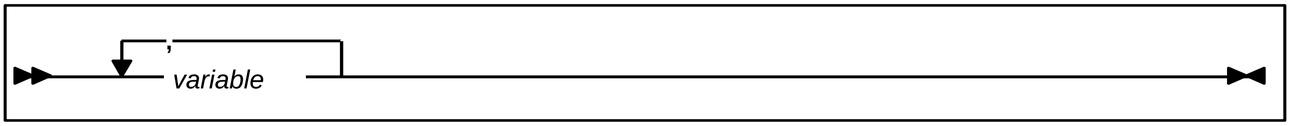
Single Optional Choice—If the first item is on the line below the baseline, one item may optionally be selected.



Defaults—Default values and parameters appear above the baseline.



Repeat Symbol—A repeat symbol indicates that more than one choice can be made or that a single choice can be made more than once. The repeat symbol shown in the following example indicates that a comma is required as the repeat separator.



Keywords—All command keywords are shown in all upper case or in mixed case. When commands are not case sensitive, mixed case implies that the lowercase letters may be omitted to form an abbreviation.

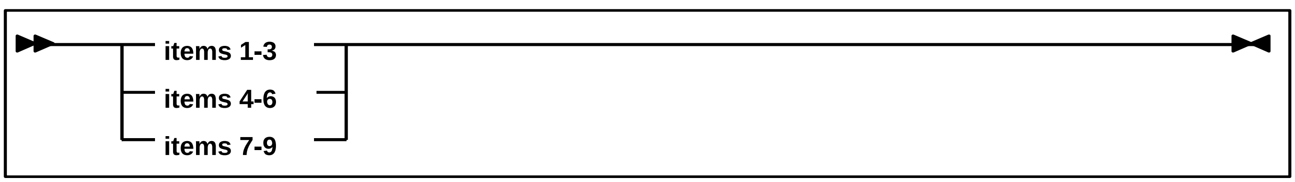
Variables—Italic type is used to indicate a variable.

Alternatives—A bar (|) is used to separate alternative parameter values.

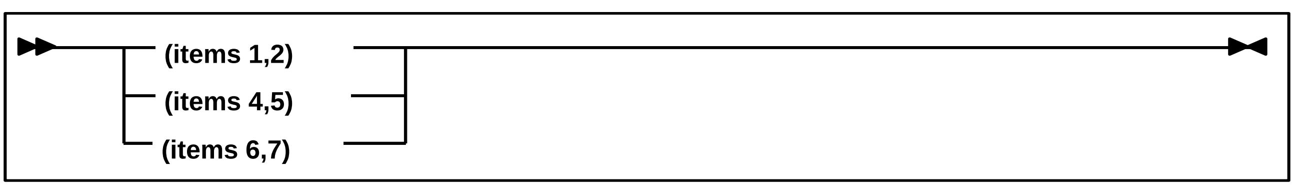
Optional—Brackets [ ] are used to indicate that a command parameter is optional.

Delimiters—If a comma (,), a semicolon (;), or other delimiter is shown with an element of the syntax diagram, it must be entered as part of the statement or command.

Ranges—An inclusive range is indicated by a pair of elements of the same length and data type, joined by a dash. The first element must be strictly less than the second element.



Lists—A list consists of one or more elements. If more than one element is specified, the elements must be separated by a comma or a blank and the entire line must be enclosed by parentheses.



Additional Information

Sun Microsystems, Inc. (Sun) offers several methods for you to obtain additional information.

Sun's External Web Site

Sun's external Web site provides marketing, product, event, corporate, and service information. The external Web site is accessible to anyone with a Web browser and an Internet connection.

The URL for the Sun external Web site is: http://www.sun.com

The URL for Sun StorageTek™ brand-specific information is: http://www.storagetek.com

Customer Resource Center

The Sun StorageTek product Customer Resource Center (CRC) is a Web site that enables members to resolve technical issues by searching code fixes and technical documentation for StorageTek brand products. CRC membership entitles you to other proactive services, such as HIPER subscriptions, technical tips, answers to frequently asked questions, addenda to product documentation books, and online product support contact information. Customers who have a current warranty or a current maintenance service agreement may apply for membership by clicking on the `Request Password` button on the CRC home page. Sun employees may enter the CRC through the SunWeb PowerPort.

The URL for the CRC is http://www.support.storagetek.com

Partners Site

The StorageTek Partners site is a Web site for partners with a StorageTek Partner Agreement. This site provides information about products, services, customer support, upcoming events, training programs, and sales tools to support StorageTek Partners. Access to this site, beyond the Partners Login page, is restricted. On the Partners Login page, Sun employees and current partners who do not have access can request a login ID and password and prospective partners can apply to become StorageTek resellers.

The URL for the StorageTek Partners site is: http://members.storagetek.com

The URL for partners with a Sun Partner Agreement is: http://www.sun.com/partners/

Hardcopy Publications

Contact a Sun sales or marketing representative to order additional paper copies of this publication or to order other StorageTek brand product customer publications in paper format.

Customer Support

Customer support is available 24 hours a day, seven days a week, to customers with Sun or StorageTek maintenance contracts and to Sun employees. You can find additional information about customer support on the Customer Resource Center (CRC) Web site at:

http://www.support.storagetek.com

Customer-initiated Maintenance

Customer-initiated maintenance begins with a telephone call from you to Sun Microsystems StorageTek Support. You receive immediate attention from qualified Sun personnel, who record problem information and respond with the appropriate level of support.

To contact Sun Microsystems StorageTek Support about a problem:

1. Use the telephone and call:

    ☎ **800.525.0369** (inside the United States)

    ☎ **303.673.4056** (outside the United States)

2. Describe the problem to the call taker. The call taker will ask several questions and will either route your call to or dispatch a support representative.

If you have the following information when you place a service call, the process will be much easier:

Account name	
Site location number	
Contact name	
Telephone number	
Equipment model number	
Device address	
Device serial number (if known)	
Urgency of problem	
Fault Symptom Code (FSC)	
Problem description	

Sun’s Worldwide Offices

You may contact any of Sun’s worldwide offices to discuss complete storage, service, and support solutions for your organization. You can find address and telephone number information on Sun’s external Web site at:

http://www.sun.com/worldwide/

Document Effectivity

EC Number	Date	Doc Kit Number	Type	Effectivity
132246	July 2006	---	Revision A	This document applies to VTCS, Version 6.1.0.

Contents

List of Figures

List of Tables

Chapter 1. VTCS Utilities and Commands

This chapter contains reference information for the following VTCS commands and utilities, where the “Interfaces” section describes the valid interfaces (command only, utility only, or both). For more information, see “Using VTCS Utilities” on page 2 and “Using VTCS Commands” on page 4.

Note: Both VTCS commands and utilities have a limit of 32000 bytes per command string (even continued over multiple lines).

Using VTCS Utilities

You use the `SWSRTV` program to run the `RTV` utility. To invoke all other VTCS utilities, you use the `SWSADMIN` program, which follows the same syntax rules and accepts the same input parameters as the HSC `SLUADMIN` utility program (including that the maximum length of a control statement is 32,767 characters).

The `SWSADMIN` program determines the HSC Primary CDS as follows:

- If your JCL **does not** specify the CDS and HSC is up, `SWSADMIN` queries HSC for the Primary CDS and `SWSADMIN` uses that CDS. The JCL examples in this chapter show this method.
- If your JCL specifies all defined copies of the CDS, `SWSADMIN` queries these copies and uses the correct copy. Note that for the `CONFIG` utility, you must explicitly specify the CDS in your JCL because you should run `CONFIG` when HSC is down.

Caution: As described in “Using VTCS Commands”, entering VTCS commands requires a `.VT` before the command name. The `SWSADMIN` program does not require a `.VT` before the utility name, and adding a `.VT` produces an error.

Note: You can produce utility reports in structured XML format. You can then process the XML output as described in “Sample REXX Execs” on page 3.

Hint: HSC provides utilities that manage library resources. These utilities are also available to VSM, and include the Scratch Update utilities, which you can use to manage the scratch status of VTVs and MVCs. For more information about the HSC utilities, see Chapter 5, “Utility Functions” in the *HSC System Programmer’s Guide for MSP*.

Note that you cannot use the Scratch Update utilities to scratch MVCs unless you have removed them from the MVC pool.

Sample REXX Execs

Two sample REXX execs are provided in the NCS SAMPLIB (`ncs_610.SLSSAMP`) to process the XML output from MVC and VTV reports:

- `SWSMVCXM` - processes MVC report XML output.
- `SWSVTVXM` - processes VTV report XML output.

These execs convert the XML tags and data into a file containing comma separated variables that can be loaded into a Microsoft Excel spreadsheet.

Using the Sample REXX Execs

To use the sample execs:

1. **Produce an MVC or VTV report with XML output.**

   For more information, see “MVCRPT” on page 121 and “VTVRPT” on page 173.

2. **Run the REXX in foreground or batch TSO to generate the CSV file.**
3. **Download the CSV file to a PC using a file transfer program that specifies translation from EBCDIC to ASCII.**
4. **Open the CSV file in MicroSoft Excel and process as desired.**

Using VTCS Commands

Like HSC commands, VTCS commands consist of the HSC command prefix character followed immediately by a command name and required or optional positional parameters and keyword parameters.

The VTCS command prefix is the same command prefix used by the HSC with which the VTCS is communicating. This allows HSC to intercept and interpret the command. For example, to cancel all active and queued VSM processes that use an RTD, enter the following:

```
.VT CAN T(ALL)
```

For more information about the rules governing commands, refer to Chapter 2, “Commands, Control Statements, and Utilities,” in *HSC Operator’s Guide for MSP*.

AUDIT

AUDIT updates the MVC and VTV information in the HSC CDS.

Syntax

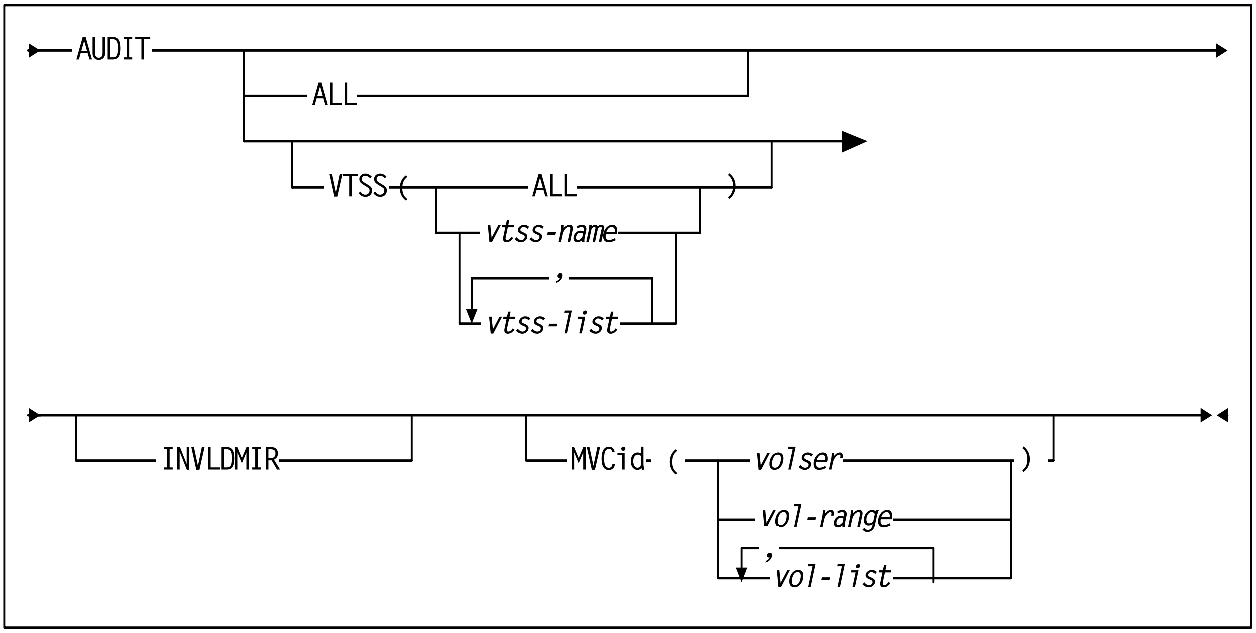


Figure 1. Audit ***syntax***

Parameters

ALL

specifies an audit of your entire VSM system, including all VTSSs and all MVCs.

Note: The ALL parameter causes VTCS to attempt to audit all defined MVCs. If any MVC is outside the ACS, HSC will prompt you to enter the MVC into the correct LSM.

VTSS

specifies an audit of one or more VTSSs.

ALL

specifies all VTSSs.

vtss-name **or** vtss-list

the names of one or more VTSSs.

INVLDMIR

specifies an audit of MVCs with invalid MIRs.

MVCid

specifies an audit of one or more MVCs.

volser, vol-range **or** vol-list

the volsers of one or more MVCs.

Interfaces

SWSADMIN utility only.

Usage

Use the AUDIT to update the MVC and VTV information in the HSC CDS.

When you run the AUDIT, VTCS splits the work into multiple audit subtasks which can use all available RTDs. The audit subtasks compete for RTDs with other VTCS tasks, such as recalls. When an audit subtask for an individual MVC completes, the RTD becomes available for other tasks, such as recalls or other audit subtasks.

Note, however, that VTCS will only run one AUDIT batch job at a time. If you submit multiple AUDIT batch jobs, all audit tasks for jobs after the first job submitted are queued behind tasks for the first job.

For more information on using AUDIT to do CDS recovery, see *VTCS Administrator's Guide*.

JCL Requirements

The following are the required and optional statements for the AUDIT JCL:

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the AUDIT report.

SLSIN
: specifies the input to the SWSADMIN program (AUDIT utility name and parameters).

JCL Example

Figure 2 shows example JCL to run AUDIT for your entire VSM system.

```
//AUDIT     EXEC PGM=SWSADMIN
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  AUDIT ALL
```

Figure 2. ***Example JCL for the*** AUDIT ***utility***

Audit Report

An audit report lists the VTVs and MVCs that are different from those listed in the CDS as shown in Figure 3. In this figure, the report shows all MVCs or VTVs as new entries in the CDS, which is typical of the output of a VTCS audit run after you lost all copies of the CDS, then ran the recovery procedure described in “Usage” on page 6.

```
SWSADMIN (6.1.0)                    STORAGETEK VTCS SYSTEM UTILITY
TIME 03:15:42                                 VTCS AUDIT

AUDIT REPORT FOR MVC EVT500
X28955 VTV ADDED AS PRIMARY COPY (BLOCK:00000000)
X20000 VTV ADDED AS PRIMARY COPY (BLOCK:0940044D)
======== AUDIT OF MVC EVT500 COMPLETED SUCCESSFULLY ========

AUDIT REPORT FOR MVC EVT501
X28956 VTV ADDED AS PRIMARY COPY (BLOCK:00000000)
X20007 VTV ADDED AS PRIMARY COPY (BLOCK:0940044D)
X20010 VTV ADDED AS SECONDARY COPY (BLOCK:11400899)
X20069 VTV NOT CURRENT (BLOCK:1A400CE5)
X20067 VTV NOT CURRENT (BLOCK:334016AB)
======== AUDIT OF MVC EVT501 COMPLETED SUCCESSFULLY ========

AUDIT REPORT FOR VTSS HBVTSS17
X20000 VTV VALID
X20002 VTV VALID
X20005 VTV VALID
X20006 VTV VALID
X20007 VTV VALID
X30052 VTV VALID
X30053 VTV VALID
X30054 VTV VALID
 ======== AUDIT OF VTSS HBVTSS17 COMPLETED SUCCESSFULLY ========

AUDIT REPORT FOR VTSS HBVTSS16
X20183 VTV VALID
X20185 VTV VALID
X20188 VTV VALID
X20190 VTV VALID
X20191 VTV VALID
X20194 VTV VALID
X41091 VTV VALID
X41093 VTV VALID
======== AUDIT OF VTSS HBVTSS16 COMPLETED WITH 1 WARNING ========
AUDIT EXCEPTION REPORT
 VTSS HBVTSS16: 1 WARNINGS REPORTED
 SLS1315I SWS500.V5.CDS WAS SELECTED AS THE PRIMARY CONTROL DATA SET
```

Figure 3. ***Example*** `AUDIT` ***utility report***

Note: An audit also generates:

- MVC summary and detail reports. For more information, see “Output” on page 52.
- `Display VTSS` summary and detail output. For more information, see “Output” on page 52.
- For every VTV resident on the VTSS, the VTV volser, size in Mb, and Management Class.

Audit Report Messages

For every VTV found on an MVC or VTSS, the audit report lists one of following:

`vvvvvv VTV possibly corrupt (Block:bbbbbb)`

Explanation: During the audit, an I/O error occurred for VTV `vvvvvv` at block `bbbbbb` on the MVC being audited.

`vvvvvv VTV not found [ , no MVC copies left ]`

Explanation: The audit did not find VTV `vvvvvv` on the MVC or VTSS being audited. If `no MVC copies left` appears, no MVCs contain copies of the VTV.

`vvvvvv VTV not found on CDS (Block:bbbbbb)`

Explanation: The audit expected but did not find VTV `vvvvvv` at block `bbbbbb` on the MVC being audited.

`vvvvvv VTV not current (Block:bbbbbb)`

Explanation: The audit found a non-current copy of VTV `vvvvvv` at block `bbbbbb` on the MVC being audited.

`vvvvvv VTV copy valid (Block:bbbbbb)`

Explanation: The audit found a valid copy VTV `vvvvvv` at block `bbbbbb` of the MVC being audited; its location matches the CDS entry for the VTV.

`vvvvvv VTV Added as primary copy (Block:bbbbbb)`

Explanation: The audit found the most current copy of VTV `vvvvvv` at block `bbbbbb` of the MVC being audited; the audit added this location to the CDS as the primary MVC copy of the VTV.

`vvvvvv VTV Added as secondary copy (Block:bbbbbb)`

Explanation: The audit found the second most current copy of VTV `vvvvvv` at block `bbbbbb` of the MVC being audited; the audit added this location to the CDS as the secondary MVC copy of the VTV.

`vvvvvv Duplicate copy ignored (Block:bbbbbb)`

Explanation: The audit found a duplicate copy of VTV `vvvvvv` at block `bbbbbb` and ignored this copy.

`vvvvvv Link to old version on MVC mmmmmm removed`

Explanation: The audit found a newer version of the VTV and removed the link to the old version from the CDS.

`vvvvvv Old VTV version deleted from VTSS ssssssss`

Explanation: The audit found an old version of the VTV and deleted it from the VTSS.

`vvvvvv Old version of VTV retained [ VTSS ssssssss ]`

Explanation: The audit found an old version of the VTV, which is the only copy, and retained this version. If `VTSS ssssssss` appears, the audit found the VTV on a different VTSS than the one that was audited.

`vvvvvv Version older than MVC copies [ VTSS ssssssss ]`

Explanation: The audit found an version of the VTV that is older than copies on the MVC. If `VTSS ssssssss` appears, the audit found the VTV on a different VTSS than the one that was audited.

`vvvvvv Newer version of VTV found [ on VTSS ssssssss ]`

Explanation: The audit found a newer version of the VTV and updated the CDS with this location. If `on VTSS ssssssss` appears, the audit found the VTV on a different VTSS than the one that was audited.

`vvvvvv VTV discovered [ VTSS ssssssss ]`

Explanation: The audit found a current version of the VTV on a VTSS whose location was unexpected and updated the CDS with this location. If `on VTSS ssssssss` appears, the audit found the VTV on a different VTSS than the one that was audited.

`vvvvvv VTV valid [ VTSS ssssssss ]`

Explanation: The audit found a valid version of the VTV and updated the CDS with this location. If `VTSS ssssssss` appears, the audit found the VTV on a different VTSS than the one that was audited.

`*** vvvvvv no access to VTSS ssssssss ***`

Explanation: The audit found a valid version of the VTV which is on a VTSS that the host cannot access.

`MVC mmmmmm STATUS CHANGED FROM EXPORT TO CONSOLIDATE VOLUME`

Explanation: The audit discovered current VTVs on an export MVC that was created by export by VTV or Management Class. The audit changed the MVC status from export to consolidate and updated the CDS to add the MVC and its VTVs.

`EXPORT MVC mmmmmm IS NOW MADE EMPTY IN THE CDS`

Explanation: The audit discovered no current VTVs on an export MVC that was created by export by VTV or Management Class. The audit marked the MVC as empty.

```
WARNING MVC mmmmmm IS AN OUTPUT MVC FROM AN EXPORT OPERATION -
FORCING READONLY
```

Explanation: The audit forced read-only status on export MVC `mmmmmm`.

```
Audit terminated. Unable to determine the position of the end of
VTV vvvvvv on MVC mmmmmm
```

Explanation: VTCS issued an Inventory MVC ECAM request to determine the position and volser of a VTV on the MVC being audited. VTSS indicated, in the ECAM response, that it was unable to determine the position of the end of the VTV. Because VTCS needs that information to determine the position of the next VTV on the MVC (assuming end of tape has not been reached), VTCS had to terminate the audit with RC=8. The MVC is left in Audit status. To resolve the condition, drain the MVC. If you cannot drain the MVC, contact StorageTek Software Support.

CANCEL

CANcel stops active and queued processes that use an RTD.

Syntax

```
►►─CANcel──────────┬─ID(process-id)─────────────────────────────────►◄
                   └───Type───( ─┬──ALL──────┬ )─┘
                                 ├─MIGrate───┤
                                 ├─RECAll────┤
                                 └─RECLaim───┘
```

Figure 4. CANcel syntax

Parameters

ID

specifies a process to cancel.

process-id

the process ID.

Type

specifies the type of process to cancel.

ALL

cancel all processes.

MIGrate

cancel all migration processes.

RECAll

cancel all recall processes.

RECLaim

cancel all reclaim processes.

Interfaces

SWSADMIN utility and VT command.

Usage

For exception conditions only, use CANcel to stop all active and queued processes. VTCS tries to stop processes without affecting system resources or information; therefore, the cancellation may not occur immediately. For example, VTCS may wait for hardware time out periods before terminating a process using a specific RTD. See “DISPLAY” on page 45 for information about determining process IDs.

If you cancel a parent request, you stop the parent and all child requests. If you cancel a child request, the parent request continues processing. See “Command Examples” on page 50 for an example of parent and child requests.

Caution: If you cancel a task associated with migration scheduler (either with the MIGrate parameter or by specific process ID), this task will terminate but migration scheduler will start another migration task at its next timer interval. You can, however, use migrate-to-threshold to stop automigration by specifying a value greater than the current DBU. For more information, see “MIGRATE” on page 98.

Command Example

To cancel a process with an ID of 10, enter the following:

```
.VT CAN ID(10)
```

JCL Requirements

The following are the required and optional statements for the AUDIT JCL:

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the AUDIT report.

SLSIN
: specifies the input to the SWSADMIN program (AUDIT utility name and parameters).

JCL Example

Figure 2 shows example JCL to cancel a process with an ID of 10.

```
//AUDIT     EXEC PGM=SWSADMIN
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  CANCEL ID(10)
```

Figure 5. Example JCL for the CANCEL utility

CONFIG

CONFIG defines or modifies the VSM configuration stored in the HSC CDS.

The following sections show the syntax of the CONFIG utility and of the input statements to CONFIG. As shown in “JCL Examples” on page 32, you create a single file that contains the CONFIG statement and its input statements.

CONFIG Statement

The CONFIG statement specifies whether this a new or updated configuration (via the RESET parameter). This statement is required.

Syntax

```
►►──CONFIG──┬─────────┬─────────┬─────────────────────────────────┬──►◄
            └──RESET──┘         └─CDSLEVEL(─┬─V6ABOVE──┬─)─┘
                                            └─V61ABOVE─┘
```

Figure 6. ***CONFIG statement syntax***

Parameters

RESET
: is required as described in Table 2.

Table 2. ***VTCS 6.1 CONFIG RESET Requirements***

CDS is at “F” Level	CDS is below “F”
Required **only** when changing the CDS level.	You **must** specify RESET when you make hardware changes such as: • Adding a VTSS to your configuration. • Adding RTDs, removing RTDs, or reordering their sequence in your configuration. • Physically removing a VTSS from your configuration. • Changing the CDS level. • You **do not** need to specify RESET if you: • Change VSM volumes (such as adding VTVs and MVCs). • Change VSM policies (such as changing AMT values).

Note: HSC must be down on all hosts when you run CONFIG RESET. The changes you made to RTD definitions will take effect when you restart HSC.

`CDSLEVEL`

One of the following CDS levels:

`V6ABOVE`

creates an “E” level CDS, which supports the following features:

- Full VSM4 Support
- 4 MVC copies
- 800 Mb VTVs (see for additional requirements)

`V61ABOVE`

creates an “F” level CDS, which supports the following features:

- Full VSM4 Support
- 4 MVC copies
- 800 Mb VTVs (see for additional requirements)
- Near Continuous Operations (NCO)
- Bi-directional clustering

Interfaces

`SWSADMIN` utility only.

GLOBAL Statement

The GLOBAL statement specifies global values for MVCs. This statement is required.

Syntax

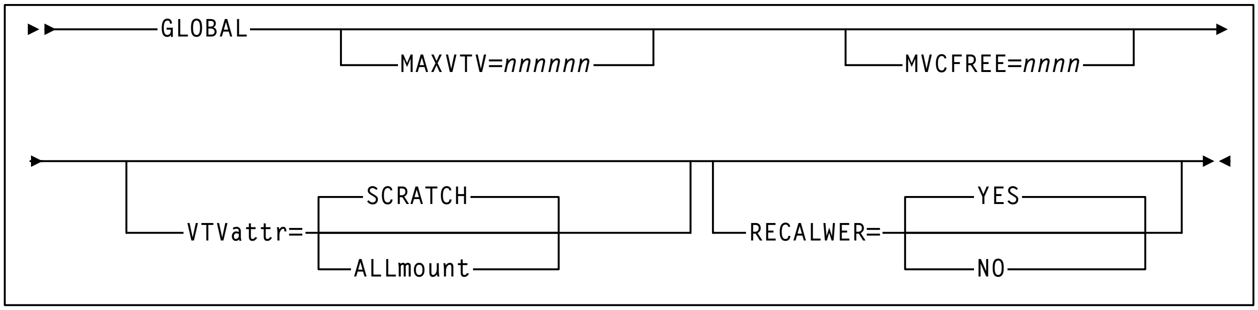


Figure 7. GLOBAL ***statement syntax***

Parameters

MAXVTV=nnn

specifies the maximum number of VTVs that can be migrated to a single MVC. Valid values are 4 to 32000. The default is 32000.

MVCFREE=nnn

specifies the minimum number of free MVCs in the MVC pool. A free MVC has 100% usable space and does not contain any migrated VTVs. Valid values are 0 to 255. The default is 40.

If free MVCs is equal or less than this value, VTCS issues message SLS6616I and starts an automatic space reclamation.

Note: If you set MVCFREE=0, VTCS actually uses the default value (40).

VTVattr=SCRATCH | ALLmount

specifies when VTCS assigns a Management Class to a VTV.

SCRATCH

Assign a Management Class only when VTCS does a scratch mount of the VTV (the default).

ALLmount

Assign a Management Class whenever VTCS mounts the VTV.

Caution: If you specify that VTCS assigns a Management Class whenever VTCS mounts a VTV, these attributes can change, which can cause undesirable or unpredictable results.

For example, if an application writes data set PROD.DATA to VTV100 with a Management Class of PROD, then writes data set TEST.DATA to VTV100 with a Management Class of TEST, then the VTV (and both data sets) has a Management Class of TEST. Similarly, it is possible to write TAPEREQ statements or SMS routines that assign different Management Classes to the same data set (for example, based on jobname), which can also cause a VTV's Management Class to change.

`RECALWER`

specifies whether VTCS recalls VTVs with read data checks (applies to recall and drain operations).

`YES`

recall VTVs with read data checks (the default).

`NO`

Do not recall VTVs with read data checks.

RECLAIM Statement

The `RECLAIM` statement controls demand and automatic MVC space reclamation. This statement is required.

Syntax

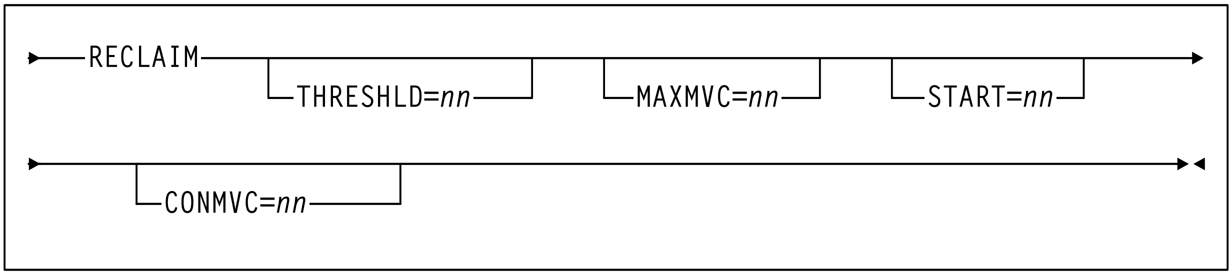


Figure 8. `RECLAIM` ***statement syntax***

Parameters

`THRESHLD=mm`
: specifies the percentage of fragmented space that makes an MVC eligible for demand or automatic reclamation. Valid values are 4 to 98. The default is 40.

`MAXMVC=nn`
: specifies the maximum number of MVCs that will be processed by a single space reclamation task. Valid values are 1 to 98. The default is 40.

For automatic space reclamation to start, the number of eligible MVCs (determined by the `THRESHLD` parameter) must also exceed the `MAXMVC` value.

`START=nn`
: specifies the level at which automatic space reclamation starts for each ACS (not globally for all ACSs). Specify a percentage value, which is equal to:

(`Reclaim Candidates/Reclaim Candidates + Free MVCs`) * 100

Where:

`Reclaim Candidates`
: is the number of Reclaim Candidates determined by the `CONFIG RECLAIM THRESHLD` parameter.

`Reclaim Candidates + Free MVCs`
: equals the number of Reclaim Candidates plus the number of free MVCs.Valid values are 1 to 98. The default is 35.

`CONMVC=nn`
: specifies the maximum number of MVCs that VTCS concurrently processes for both drain and reclaim.

Valid values are 1 to 99. The default is 1.

VTVVOL Statement

The VTVVOL statement defines a range of VTVs.

You can only add new VTV ranges. A range can consist of a single volume. You cannot delete or modify existing ranges. You can, however, respecify existing VTV ranges when you add new ranges (for example, by adding new VTV ranges to the output of the DECOM utility).

Note: The following restrictions when you respecify existing ranges:

- If you respecify any existing range, you must respecify all existing ranges.
- The high and low volume serial numbers of each respecified range must exactly match the previously specified range.
- The volume type for each respecified range must be the same as the original specification (MVC or VTV).
- Each range can be respecified only once.

Syntax

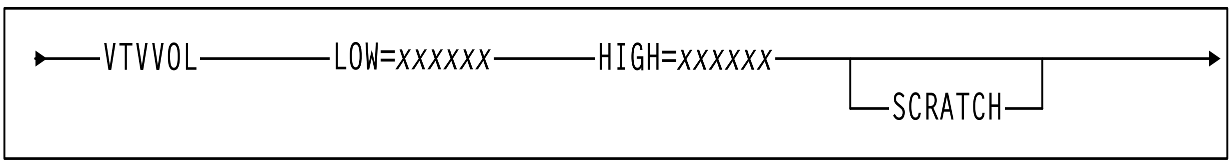


Figure 9. VTVVOL ***statement syntax***

Parameters

LOW=xxxxxx
: specifies the start of a range of VTVs.

HIGH=xxxxxx
: specifies the end of a range of VTVs.

Note: The incremental part of a range cannot exceed 5 characters for numeric and 4 characters for alpha ranges.

SCRATCH
: specifies that the VTVs added to the CDS are placed in scratch status, which is not the default for the VTVVOL parameter.

Warning: If you are using the ExLM SYNCVTV function for VTV scratch synchronization, StorageTek recommends that you define VTV ranges in scratch status. If you do not, you must use the HSC SLUADMIN utility to scratch these volumes.

MVCVOL Statement

The `MVCVOL` statement defines a range of MVCs available to VTCS.

You can only add new MVC ranges. A range can consist of a single volume. You cannot delete or modify existing ranges. You can, however, respecify existing MVC ranges when you add new ranges (for example, by adding new MVC ranges to the output of the `DECOM` utility).

Note: The following restrictions when you respecify existing ranges:

- If you respecify any existing range, you must respecify all existing ranges.
- The high and low volume serial numbers of each respecified range must exactly match the previously specified range.
- The volume type for each respecified range must be the same as the original specification (MVC or VTV).
- Each range can be respecified only once.

Syntax

```
►──MVCVOL──────────LOW=xxxxxx──────HIGH=xxxxxx──────────────────────────►◄
```

Figure 10. `MVCVOL` ***statement syntax***

Parameters

`LOW=xxxxxx`
: specifies the start of a range of MVCs.

`HIGH=xxxxxx`
: specifies the end of a range of MVCs.

VTSS Statement

The VTSS statement defines a VTSS and sets its operating values. This statement is required.

When you define a new VTSS, place its definition after any existing VTSS definitions, which must remain in their original order. You must specify RESET when you add a VTSS to your configuration as described in “RESET” on page 13.

Syntax

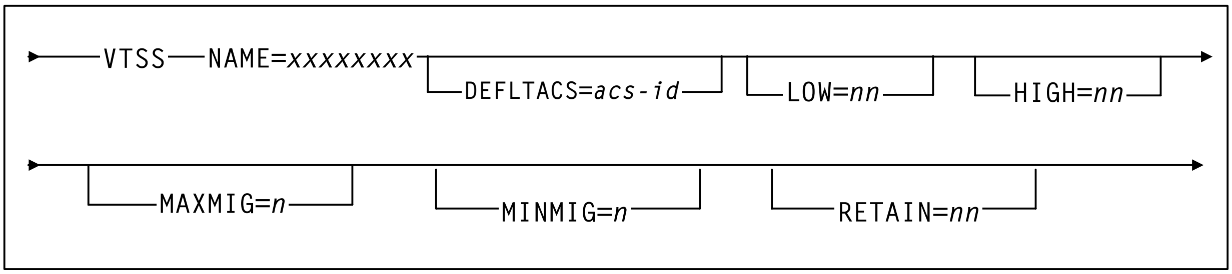


Figure 11. **VTSS statement syntax**

Parameters

Note: If you physically remove a VTSS from your configuration, reconfigure the VTSS with a VTSS statement only and no parameters. See Figure 21 on page 36 for an example.

NAME=xxxxxxxx

specifies the VTSS identifier. This parameter is required; there is no default value.

Caution: Note the following:

- You specify the VTSS identifier *only* via the NAME parameter, which sets the VTSS identifier in both the VTSS microcode (as displayed in the Subsystem Name field in the LOP) and in the configuration area of the HSC CDS. After VSM is put into operation, the VTSS identifier is also stored in each VTV record in the CDS. Each VTV record contains the VTSS identifier on which that VTV is resident or, if the VTV is migrated, the VTV record contains the VTSS identifier from which the VTV was migrated.

- Once you set the VTSS identifier via the `NAME` parameter, you *cannot* change this identifier in the HSC CDS. That is, the `CONFIG` utility *will not* let you change the `NAME` parameter after an initial setting and changing the VTSS identifier using the Subsystem Name field of the LOP *cannot* change the VTSS identifier in the HSC CDS.
- It is especially critical that you *do not* attempt to rename a VTSS that contains data on VTVs, which includes VTSS-resident VTVs and migrated VTVs!
- For an initial setting *only* (not a change), you can set the VTSS identifier in the `NAME` parameter only if the VTSS identifier value in the VTSS microcode is:
  - The factory setting (all blanks).
  - A value of `99999999` (eight 9s).

  Therefore, for an initial setting *only*, if the name in the VTSS microcode *is not* all blanks or `99999999`, your StorageTek hardware representative must use the VTSS LOP to set the VTSS identifier to `99999999` so you can set the VTSS identifier to the value you want via the `NAME` parameter.

`DEFLTACS=acs-id`

VTCS supports multi-VTSS confirmations, and supports connecting two ACSs to each VTSS. In configurations where a VTSS is connected to two ACSs, you can use the `DEFLTACS` parameter to specify the default ACS from which MVCs will be selected for migration, consolidation, and reclaim processing.

If you do not specify `DEFLTACS`, the default value is x’FF’, which allows VTCS to select MVCs from either ACS.

Note: VTCS ignores the value on the `DEFLTACS` parameter if you specify the `DEFLTACS` parameter and do either of the following:

- Specify the `ACSlist` parameter of the `MGMTclas` statement.
- Use a Storage Class.

`LOW=nn`

specifies the low automatic migration threshold (LAMT) for this VTSS.

Valid values are 5 to 95 and must be less than the `HIGH` default threshold. The default is 70.

`HIGH=nn`

specifies the high automatic migration threshold (HAMT) for this VTSS.

Valid values are 6 to 95 and must be greater than the `LOW` default threshold. The default is 80.

`MAXMIG=n`

specifies the maximum number of concurrent automatic migration, immediate migration, and migrate-to-threshold tasks for this VTSS.

Valid values are 1 to the number of RTDs on the VTSS or 8, whichever is less. The default is half the number of RTDs attached to the VTSS.

`MINMIG=n`

specifies the minimum number of concurrent automatic migration, immediate migration, and migrate-to-threshold tasks for this VTSS.

Valid values are 1 to the `MAXMIG` setting. The default is 1 task.

`RETAIN=nn`

specifies the number of minutes that VTCS will retain an MVC on an RTD in idle mode after a migration or recall. Retaining the MVC can reduce MVC mounts.

Valid values are 1 to 60. The default is 10.

RTD Statement

The RTD statement defines the RTDs connected to the VTSS. This statement is required and must follow the VTSS statement that defines the VTSS to which the RTDs are connected. For a VSM2 or VSM3, you can specify a maximum of 8 RTDs. For a VSM4, you can specify a maximum of 16 RTDs.

Note: You must specify the RESET parameter to change RTD definitions; for more information, see "RESET" on page 13. For an initial RTD definition, if the RTD name displayed at the VTSS LOP is anything other than all blanks, you must also specify RESET.

Syntax

```
►──RTD────NAME=xxxxxxxx────DEVNO=nnnn────CHANIF=nn──────────►
```

Figure 12. RTD ***statement syntax***

Parameters

NAME=*xxxxxxxx*

specifies the 1 to 8 character identifier of the RTD.

You set or change the RTD identifier *only* via the RTD NAME parameter; to do so, the RTD identifier must be all blanks as displayed at the VTSS LOP.

This parameter is required; there is no default value.

DEVNO=*nnnn*

specifies the unit address of the RTD.

This parameter is required; there is no default value.

CHANIF=*nn*

specifies the channel interface on the VTSS that communicates with the RTD. This value must match the Nearlink channel interface defined at the VTSS LOP by your StorageTek hardware representative at VTSS installation and configuration.

This parameter is required; there is no default value.

The value must be two characters in length and have a value from 0A to 1P. The first digit is the VTSS cluster ID (valid values are 0 or 1). The second digit is the group or adapter ID (valid values are A to P).

Caution: Do not use the LINK number shown at the LOP instead of the VTSS cluster ID for the first character of the CHANIF value!

VTD Statement

The VTD statement defines the MSP unit address range of the VTDs in a VTSS. This statement is required and must follow the VTSS statement where the VTDs reside.

Note: VSM2s and VSM3s provide 64 VTDs per VTSS. VSM4s provide 256 VTDs per VTSS.

You can specify the VTD unit addresses to either apply to all hosts or to define which VTDs are available to specific hosts; for more information, see “Specifying VTD Unit Addresses” on page 30.

Syntax

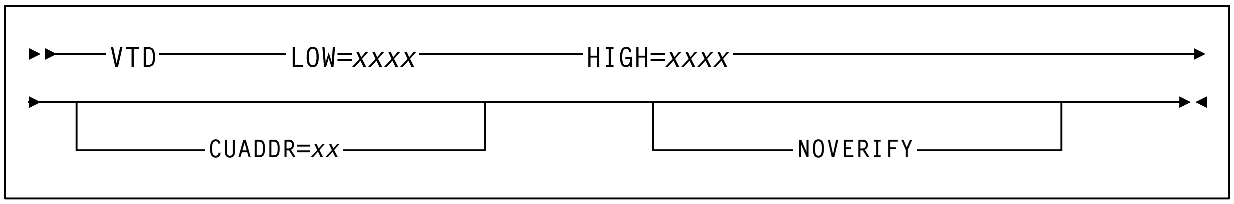


Figure 13. VTD ***statement syntax***

Parameters

LOW=xxxx
: specifies a four character valid MSP unit address as the start of a range of VTDs.

HIGH=xxxx
: specifies a four character valid MSP unit address as the end of a range of VTDs.

CUADDR=xx
: specifies a control unit identifier that matches its value in the IOCP. Valid values are 0 through 15 for VSM4 and 0 through 3 for VSM2/3. This statement is required only if a partial VTD range is specified for a host and the host does not have a path to the VTDs.

NOVERIFY
: specifies that VTCS will attempt the verification of all predicted VTD identifiers associated with Virtual Tape Drives. In order to do this each MSP I/O address must be available in order that the associated Host may issue the ECAM-T request Virtual_Device_Identify.

In specific cases where, for example, VTCS is providing support for a remote client Host connected via MVS/CSC, the local host, acting as a server for the remote client, may not have paths to the MSP I/O address. In these cases, any attempt at verification of the predicted VTD identifier will fail and will result in error messages posted to the operator.

CLUSTER Statement

The CLUSTER statement defines two VTSSs in a Cluster.

Syntax

```
►►──CLUSTER──────NAME=nnnnnnnn──────────────────────────►

►────────VTSSs=(vtssname1,vtssname2)────────────────────►◄
```

Figure 14. *CLUSTER* ***statement syntax***

Parameters

NAME=nnnnnnnn

specifies the 1 to 8 character identifier of the Cluster.

This parameter is required; there is no default value.

VTSSs=(vtssname1,vtssname2)

Specifies the names of two VTSSs in a cluster. The relationship between the two VTSS is defined by the CONFIG CLINK statement; for more information, see “CLINK Statement” on page 26.

There is no default value. This parameter replaces and is mutually exclusive with the PRIMARY and SECONDRY parameters.

Note: The VTSS names that you specify on a CLUSTER statement must be known to VTCS. That is, the VTSS names must be already defined in the CDS or must be specified in VTSS statements that already exist in the current set of CONFIG statements. For, example, the following is valid:

```
VTSSNAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
VTSSNAME=VTSS2  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
CLUSTER NAME=CLUSTER1 VTSSs=(VTSS1,VTSS2)
```

The following is **not** valid unless VTSS1 and VTSS2 are already defined in the CDS:

```
CLUSTER NAME=CLUSTER1 VTSSs=(VTSS1,VTSS2)
VTSSNAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
VTSSNAME=VTSS2  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
```

CLINK Statement

The CLINK statement defines the channel interface between two VTSSs in a Cluster. The VTSSs can be in one of two modes:

- Primary-Secondary, in which case you write `CLINK` statements for *only* the Primary.
- Peer-to-Peer, in which case you write `CLINK` statements for both VTSSs to enable bi-directional VTV replication.

For examples, see *VTCS Administrator's Guide*.

Syntax

```
►►──CLINK──────────────VTSS=vvvvvvvv──────────CHANIF=nn──────────────►◄
```

Figure 15. `CLINK` ***statement syntax***

Parameters

`VTSS=vvvvvvvv`

specifies the name of the one VTSS in a Cluster.

This parameter is required; there is no default value.

`CHANIF=nn`

specifies the channel interface for communication between two VTSSs in a Cluster. This value must match the Nearlink channel interface defined at the VTSS LOP by your StorageTek hardware representative at VTSS installation and configuration.

This parameter is required; there is no default value.

The value must be two characters in length and have a value from 0A to 1P. The first digit is the VTSS Cluster ID (valid values are 0 or 1). The second digit is the group or adapter ID (valid values are A to P).

Caution: Do not use the LINK number shown at the LOP instead of the VTSS Cluster ID for the first character of the `CHANIF` value!

Note: The VTSS name that you specify on a `CLINK` statement must be known to VTCS. That is, the VTSS name must be already defined in the CDS or it must be specified in a `VTSS` statement that already exists in the current set of `CONFIG` statements. For, example, the following is valid:

```
VTSSNAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
CLINK VTSS=VTSS1 CHANIF=0A
```

The following is **not** valid unless `VTSS1` is already defined in the CDS:

```
CLINK VTSS=VTSS1 CHANIF=0A
VTSSNAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
```

HOST Statement

The HOST statement is an optional statement that defines an MSP host and, optionally, the NOMIGRAT and/or NORECLAM parameters.

Note the following:

- If specified, the HOST statement must follow the VTSS statement for the VTSS attached to that host.
- You must either specify all host definitions or none; if you specify only some of the hosts attached to a VTSS, VTCS will issue an error.

Syntax

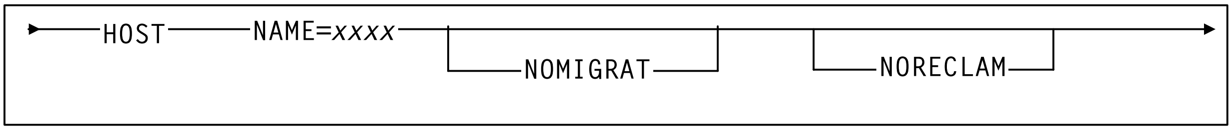


Figure 16. HOST ***statement syntax***

Parameters

NAME=xxxx
: specifies the LIBGENed hostname.

NOMIGRAT
: specifies that this host cannot do migrations, consolidations, or export by VTV or Management Class from the VTSS(s) that the host accesses. NOMIGRAT controls both automatic and demand migrations and consolidations. This parameter is optional.

Note: Specifying NOMIGRAT also causes NORECLAM to be set.

IMMEDmig KEEP and IMMEDmig DELETE are mutually exclusive with CONFIG HOST NOMIGRAT. If you specify both, the IMMEDmig value overrides NOMIGRAT, and VTCS does not issue a message about this override.

NORECLAM
: specifies that this host cannot initiate automatic or demand reclaim processing using the VTSS(s) that the host accesses (the host can still do demand MVC drains via MVCDRain). This parameter is optional.

Note: If you use the HSC SET utility to add a new host ID, or change an existing host ID, then you must run the VTCS CONFIG utility after the HSC SET utility is complete. The SET Utility adds or modifies a record to the native tape portion of the CDS for the host that you're adding/modifying, but does **not** modify the VTCS section of the CDS.

To modify the VTCS section of the CDS:

1. **After you've run the HSC SET utility, run the VTCS DECOM utility...**

   ...to see what accesses your hosts have to your VTSSs. Modify the DECOM output to what you would like your access to be for any hosts that were changed with the HSC SET utility.

2. **Use that edited DECOM output as the input to the V**TCS CONFIG utility**.**

   Note that if you are only accommodating a new host ID or a change in host IDs, it is not necessary to use the RESET parameter with CONFIG. Therefore, no hosts **other** than the one you are modifying will need to shut down HSC.

   When HSC is started on the host that you have added or modified, any changes to CONFIG for that host will take effect.

Usage

Use the CONFIG utility to define or modify the VSM configuration stored in the HSC CDS. An asterisk (*) in column 1 denotes comments in the file that contains the CONFIG utility statements.

You typically run the CONFIG utility when you:

- Initially install and configure your VSM system; see Figure 17 on page 32 and Figure 18 on page 33 for examples.
- Change VSM hardware (such as adding RTDs); see Figure 19 on page 34 for an example.
- Change VSM volumes (such as adding VTVs and MVCs); see Figure 20 on page 35 for an example.
- Change VSM policies (such as AMT values); see Figure 20 on page 35 for an example.
- Physically remove a VTSS from your configuration, see Figure 21 on page 36 for an example.

Note: You must specify RESET to change your existing RTD definitions by adding RTDs, removing RTDs, or reordering the sequence of RTDs; see "RESET" on page 13. HSC must be down on all hosts when you run CONFIG RESET. The changes you made to RTD definitions will take effect when you restart HSC. You must also specify RESET when you add a VTSS to your configuration.

You can run CONFIG without RESET concurrently with an active HSC, but you must then restart HSC for the changes to take effect.

Specifying VTD Unit Addresses

You can specify VTD addresses by doing one of the following:

- Specify the VTD unit addresses on a `VTD` statement following a `VTSS` statement and do *not* specify any `HOST` statements following the `VTSS` statement. All hosts physically connected to the VTSS have access to its VTDs by the default addresses specified on the `VTD` statement. See Figure 17 on page 32 for an example of this configuration.

- Do *not* specify the VTD unit addresses on the `VTD` statement following a `VTSS` statement. Instead, place a `VTD` statement after a `HOST` statement for only those hosts for which you want to define connections to the previously defined VTSS. You must specify a placeholder (`HOST NAME` with no `VTD` parameter) for any hosts that you do not want connected to this VTSS.

  Note that the VTVs created and MVCs initially written to from a VTSS are considered that VTSS's resources, so only hosts with access to a VTSS also have access to its VTVs and MVCs. In this type of "restricted" access configuration, therefore, each host should have a separate VTV scratch pool to ensure that each host has accurate scratch counts. Similarly, free MVCs and MVC reclaim counts are reported on each host for the MVCs associated with the VTSS to which the host is connected.

  See Figure 18 on page 33 for an example of this configuration. You can specify different address ranges for each host, although StorageTek recommends that you specify the same address ranges for all hosts for consistency of operations. If you specify different address ranges for different hosts, use the HSC SET DRVHOST and SMC DRIVEMAP statements if you have a client/server configuration.

Caution: In a multi-host, multi-VTSS configuration, you can use this VTD addressing method to deny access to VTSSs to which hosts are physically connected. You **must**, however, use this method to deny access from hosts that are *not* physically connected to a VTSS. If you do not deny access, VTCS on a host that does not have physical connections to a VTSS may wait trying to communicate with the VTSS while VSM operations may be stalled on all other hosts.

JCL Requirements

`STEPLIB`
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

`SLSPRINT`
: specifies the destination for the `CONFIG` report.

`SLSIN`
: specifies the input to the `SWSADMIN` program (`CONFIG` utility name and parameters).

JCL Examples

Note: See *VTCS Installation and Configuration Guide* for VSM4 CONFIG examples.

CONFIG Example: VSM2 or VSM3 Initial Configuration - All Hosts Access All VTDs

Figure 17 shows example CONFIG JCL to initially define a VSM2 or VSM3 configuration as follows:

- The VTD statements specify default VTD addresses 8900 - 893F for VTSS1 and addresses 9900 - 993F for VTSS2.
- No HOST statements follow the VTSS statements, so all hosts have access to all VTDs in both VTSSs by the default addresses specified on the VTD statements.

```
//CREATECFG   EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB     DD DSN=hlq.SLSLINK,DISP=SHR
//SLSCNTL     DD DSN=FEDB.VSMLMULT.DBASEPRM,DISP=SHR
//SLSCNTL2    DD DSN=FEDB.VSMLMULT.DBASESEC,DISP=SHR
//SLSSTBY     DD DSN=FEDB.VSMLMULT.DBASETBY,DISP=SHR
//SLSPRINT    DD   SYSOUT=*
//SLSIN       DD   *
 CONFIG
 GLOBAL   MAXVTV=32000   MVCFREE=40
 RECLAIM  THRESHLD=70    MAXMVC=40  START=35
 VTVVOL LOW=905000 HIGH=999999 SCRATCH
 VTVVOL LOW=C00000 HIGH=C25000 SCRATCH
 VTVVOL LOW=RMM000 HIGH=RMM020 SCRATCH
 MVCVOL LOW=N25980 HIGH=N25989
 MVCVOL LOW=N35000 HIGH=N35999
 VTSS  NAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS18800 DEVNO=8800 CHANIF=0A
       RTD    NAME=VTS18801 DEVNO=8801 CHANIF=0I
       RTD    NAME=VTS18802 DEVNO=8802 CHANIF=1A
       RTD    NAME=VTS18803 DEVNO=8803 CHANIF=1I
       VTD    LOW=8900 HIGH=893F
 VTSS  NAME=VTSS2  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS28804 DEVNO=8804 CHANIF=0A
       RTD    NAME=VTS28805 DEVNO=8805 CHANIF=0I
       RTD    NAME=VTS28806 DEVNO=8806 CHANIF=1A
       RTD    NAME=VTS28807 DEVNO=8807 CHANIF=1I
       VTD    LOW=9900 HIGH=993F
```

Figure 17. CONFIG ***example: initial VSM2 or VSM3 configuration, all hosts access all VTDs***

CONFIG Example: Initial Configuration - All Hosts Access VTDs in One VTSS, Only Selected Hosts Access VTDs in Second VTSS

Figure 18 shows example CONFIG JCL to initially define a VSM configuration as follows:

- The VTD statement specifies default VTD addresses 8900 - 893F for VTSS1. All hosts have access to these VTDs by their default addresses.
- No default VTD addresses are specified for VTSS2. The HOST statements for MSP1 and MSP2 specify that only these hosts can access the VTDs in VTSS2 by the addresses 9900 - 993F. HOST statement MSP3 is a placeholder; this host cannot access the VTDs in VTSS2.

```
//CREATECFG    EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSCNTL      DD DSN=FEDB.VSMLMULT.DBASEPRM,DISP=SHR
//SLSCNTL2     DD DSN=FEDB.VSMLMULT.DBASESEC,DISP=SHR
//SLSSTBY      DD DSN=FEDB.VSMLMULT.DBASETBY,DISP=SHR
//SLSPRINT     DD   SYSOUT=*
//SLSIN        DD   *
 CONFIG
 GLOBAL   MAXVTV=32000   MVCFREE=40
 RECLAIM  THRESHLD=70    MAXMVC=40  START=35
 VTVVOL LOW=905000 HIGH=999999 SCRATCH
 VTVVOL LOW=C00000 HIGH=C25000 SCRATCH
 VTVVOL LOW=RMM000 HIGH=RMM020 SCRATCH
 MVCVOL LOW=N25980 HIGH=N25989
 MVCVOL LOW=N35000 HIGH=N35999
 VTSS  NAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
       RTD    NAME=VT128800 DEVNO=8800 CHANIF=0A
       RTD    NAME=VTS18801 DEVNO=8801 CHANIF=0I
       RTD    NAME=VTS18802 DEVNO=8802 CHANIF=1A
       RTD    NAME=VTS18803 DEVNO=8803 CHANIF=1I
       VTD    LOW=8900 HIGH=893F
 VTSS  NAME=VTSS2  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS28804 DEVNO=8804 CHANIF=0A
       RTD    NAME=VTS28805 DEVNO=8805 CHANIF=0I
       RTD    NAME=VTS28806 DEVNO=8806 CHANIF=1A
       RTD    NAME=VTS28807 DEVNO=8807 CHANIF=1I
       HOST   NAME=MSP1
              VTD    LOW=9900 HIGH=993F
       HOST   NAME=MSP2
              VTD    LOW=9900 HIGH=993F
       HOST   NAME=MSP3 NOMIGRAT NORECLAM
```

Figure 18. **CONFIG example: initial configuration, all hosts access VTDs in one VTSS, selected hosts access VTDs in second VTSS**

CONFIG Example: Update Configuration to Add RTDs

Figure 19 shows example JCL to run `CONFIG` to add RTDs `VTS18811` and `VTS18813` (connected to `VTSS1`) to the configuration shown in Figure 18 on page 33. In this example, you specify the `RESET` parameter to clear the existing RTD definitions, then respecify the existing RTDs and add new definitions for RTDs `VTS18811` and `VTS18813`.

```
//UPDATECFG    EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSCNTL      DD DSN=FEDB.VSMLMULT.DBASEPRM,DISP=SHR
//SLSCNTL2     DD DSN=FEDB.VSMLMULT.DBASESEC,DISP=SHR
//SLSSTBY      DD DSN=FEDB.VSMLMULT.DBASETBY,DISP=SHR
//SLSPRINT     DD   SYSOUT=*
//SLSIN        DD   *
 CONFIG    RESET
 GLOBAL   MAXVTV=32000   MVCFREE=40
 RECLAIM  THRESHLD=70    MAXMVC=40  START=35
 VTVVOL LOW=905000 HIGH=999999 SCRATCH
 VTVVOL LOW=C00000 HIGH=C25000 SCRATCH
 VTVVOL LOW=RMM000 HIGH=RMM020 SCRATCH
 MVCVOL LOW=N25980 HIGH=N25989
 MVCVOL LOW=N35000 HIGH=N35999
 VTSS  NAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS18800 DEVNO=8800 CHANIF=0A
       RTD    NAME=VTS18801 DEVNO=8801 CHANIF=0I
       RTD    NAME=VTS18802 DEVNO=8802 CHANIF=1A
       RTD    NAME=VTS18803 DEVNO=8803 CHANIF=1I
       RTD    NAME=VTS18811 DEVNO=8811 CHANIF=0E
       RTD    NAME=VTS18813 DEVNO=8813 CHANIF=1E
       VTD    LOW=8900 HIGH=893F
 VTSS  NAME=VTSS2  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS28804 DEVNO=8804 CHANIF=0A
       RTD    NAME=VTS28805 DEVNO=8805 CHANIF=0I
       RTD    NAME=VTS28806 DEVNO=8806 CHANIF=1A
       RTD    NAME=VTS28807 DEVNO=8807 CHANIF=1I
       HOST   NAME=MSP1
              VTD    LOW=9900 HIGH=993F
       HOST   NAME=MSP2
              VTD    LOW=9900 HIGH=993F
       HOST   NAME=MSP3
```

Figure 19. `CONFIG` ***example: updating configuration to add RTDs***

CONFIG Example: Update Configuration to Add MVCs and VTVs and Change AMTs

Figure 20 shows example JCL to run CONFIG to modify the configuration shown in Figure 19 on page 34 by:

- Respecifying the existing VTV and MVC ranges.
- Adding VTVs C25001 to C50000 as scratch.
- Adding MVCs N45000 to N45999. For more information about adding MVCs to VSM.
- Changing the LAMT to 50 and the HAMT to 85 on both VTSS1 and VTSS2.

```
//UPDATECFG    EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSCNTL      DD DSN=FEDB.VSMLMULT.DBASEPRM,DISP=SHR
//SLSCNTL2     DD DSN=FEDB.VSMLMULT.DBASESEC,DISP=SHR
//SLSSTBY      DD DSN=FEDB.VSMLMULT.DBASETBY,DISP=SHR
//SLSPRINT     DD   SYSOUT=*
//SLSIN        DD   *
 CONFIG
 GLOBAL   MAXVTV=32000   MVCFREE=40
 RECLAIM  THRESHLD=70    MAXMVC=40  START=35
 VTVVOL LOW=905000 HIGH=999999 SCRATCH
 VTVVOL LOW=C00000 HIGH=C25000 SCRATCH
 VTVVOL LOW=RMM000 HIGH=RMM020 SCRATCH
 VTVVOL LOW=C25001 HIGH=C50000 SCRATCH
 MVCVOL LOW=N25980 HIGH=N25989
 MVCVOL LOW=N35000 HIGH=N35999
 MVCVOL LOW=N45000 HIGH=N45999
 VTSS  NAME=VTSS1  LOW=50 HIGH=85  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS18800 DEVNO=8800 CHANIF=0A
       RTD    NAME=VTS18801 DEVNO=8801 CHANIF=0I
       RTD    NAME=VTS18802 DEVNO=8802 CHANIF=1A
       RTD    NAME=VTS18803 DEVNO=8803 CHANIF=1I
       RTD    NAME=VTS18811 DEVNO=8811 CHANIF=0E
       RTD    NAME=VTS18813 DEVNO=8813 CHANIF=1E
       VTD    LOW=8900 HIGH=893F
 VTSS  NAME=VTSS2  LOW=50 HIGH=85  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS28804 DEVNO=8804 CHANIF=0A
       RTD    NAME=VTS28805 DEVNO=8805 CHANIF=0I
       RTD    NAME=VTS28806 DEVNO=8806 CHANIF=1A
       RTD    NAME=VTS28807 DEVNO=8807 CHANIF=1I
       HOST   NAME=MSP1
              VTD    LOW=9900 HIGH=993F
       HOST   NAME=MSP2
              VTD    LOW=9900 HIGH=993F
       HOST   NAME=MSP3
```

Figure 20. CONFIG ***example: updating configuration to add MVCs and VTVs and change AMTs***

CONFIG Example: Denying Host Access to a Physically Removed VTSS

Figure 21 shows example JCL to run `CONFIG` to update the configuration shown in Figure 18 on page 33 to deny host access to VTSS2 that you physically removed from your configuration. In this example, you:

- Specify the `RESET` parameter to clear the existing RTD definitions.
- Respecify the `VTSS` statement for `VTSS2` with no parameters to deny host access to this VTSS.

```
//UPDATECFG    EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSCNTL      DD DSN=FEDB.VSMLMULT.DBASEPRM,DISP=SHR
//SLSCNTL2     DD DSN=FEDB.VSMLMULT.DBASESEC,DISP=SHR
//SLSSTBY      DD DSN=FEDB.VSMLMULT.DBASETBY,DISP=SHR
//SLSPRINT     DD   SYSOUT=*
//SLSIN        DD   *
 CONFIG    RESET
 GLOBAL   MAXVTV=32000   MVCFREE=40
 RECLAIM  THRESHLD=70    MAXMVC=40  START=35
 VTSS  NAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS18800 DEVNO=8800 CHANIF=0A
       RTD    NAME=VTS18801 DEVNO=8801 CHANIF=0I
       RTD    NAME=VTS18802 DEVNO=8802 CHANIF=1A
       RTD    NAME=VTS18803 DEVNO=8803 CHANIF=1I
       RTD    NAME=VTS18811 DEVNO=8811 CHANIF=0E
       RTD    NAME=VTS18813 DEVNO=8813 CHANIF=1E
       VTD    LOW=8900 HIGH=893F
 VTSS  NAME=VTSS2
```

Figure 21. `CONFIG` ***example: updating configuration to deny host access to a physically removed VTSS***

CONSolid

`CONSolid` consolidates VTVs on MVCs; for more information, see “Usage” on page 38.

Syntax

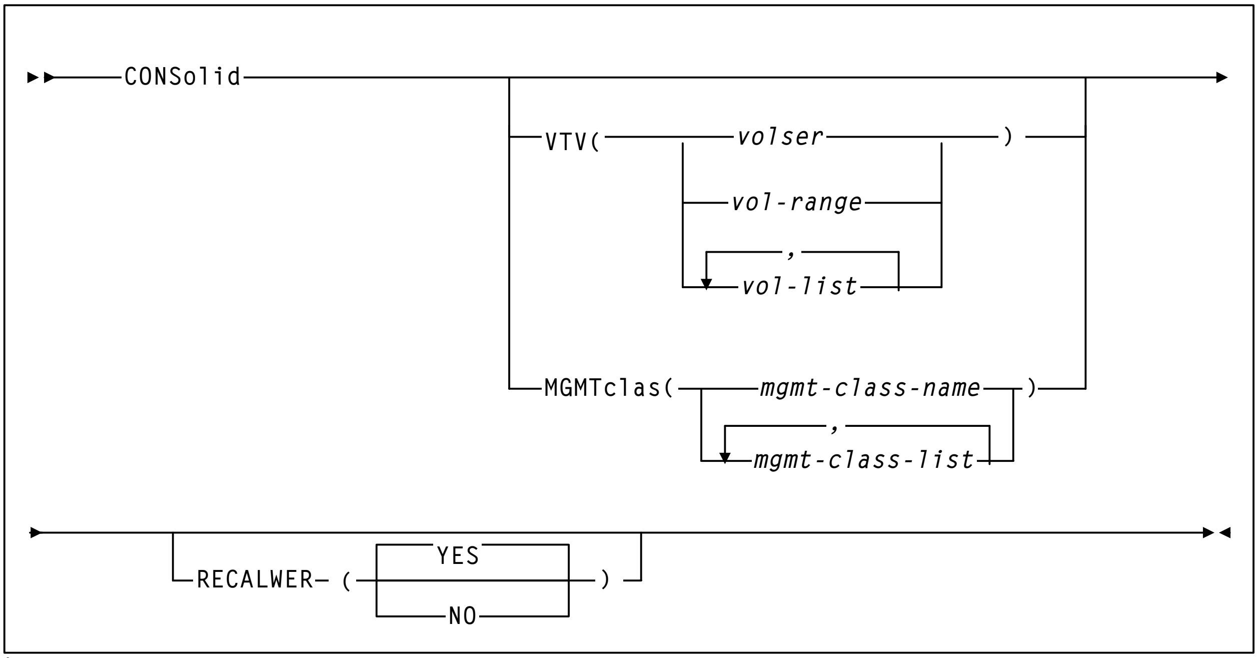


Figure 22. `CONSolid` ***utility syntax***

Parameters

`VTV`

specifies one or more VTVs to consolidate.

`volser`, `vol-range` **or** `vol-list`

the volsers of one or more VTVs. You can specify a maximum of 2,000 VTVs.

`MGMTclas`

specifies the names of one or more Management Classes that determine the VTVs to consolidate.

`mgmt-class-name` | `mgmt-class-list`

the names of one or more Management Classes that you defined on the `MGMTclas` control statement; for more information, see “MGMTCLAS Control Statement” on page 204. You can consolidate a maximum of 2,000 VTVs by specifying a Management Class.

`RECALWER`

specifies whether VTCS recalls VTVs with read data checks.

`YES`

recall VTVs with read data checks (the default).

`NO`

Do not recall VTVs with read data checks.

Interfaces

`SWSADMIN` utility only.

Usage

The following sections tell how to use the `CONSolid` utility to consolidate VTVs on MVCs:

- “Consolidating VTVs by Specifying VTVs on the VTV Parameter”
- “Consolidating VTVs by Specifying a Management Class on the MGMTclas Parameter”

Note: If you need to run a VSM audit, rerun any consolidation jobs that may have been impacted by the loss of the CDS or VSM resources. For more information, see *VTCS Administrator’s Guide*.

Consolidating VTVs by Specifying VTVs on the VTV Parameter

You can consolidate VTVs by specifying VTVs on the `VTV` parameter. VTCS will consolidate every VTV you specify that is not mounted, non-scratch, not in-use or in recovery, and not already consolidated.

To use this method, first run a tape management system or ExLM data set name report to select the VTVs to consolidate. Figure 23 on page 40 shows a JCL example of specifying VTVs to consolidate on the `VTV` parameter.

Consolidating VTVs by Specifying a Management Class on the MGMTclas Parameter

You can also consolidate VTVs by specifying a Management Class or list of Management Classes on the `MGMTclas` parameter.

To consolidate VTVs by specifying a Management Class, do the following:

1. **Define a Management Class or use an existing Management Class.**

   Note: Note if the Management Class you use for consolidation specifies the `DUPlex` parameter, duplexing is ignored for consolidation for this Management Class but duplexing *is* supported for migration for this Management Class.

2. **Create a TAPEREQ statement or use the HSC/DFSMS interface to route data to the Management Class you defined in Step 1.**

3. **Use the `CONSolid` utility to specify the Management Class you defined in Step 1.**

   The `CONSolid` utility will query the CDS to determine which VTVs have been created with this Management Class, then build a list of VTVs to consolidate. VTCS consolidates all VTVs in the list that are not mounted, non-scratch, not in-use or in recovery, and not already consolidated.

JCL Requirements

`STEPLIB`
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

`SLSPRINT`
: specifies the destination for the `CONSolid` report.

`SLSIN`
: specifies the input to the `SWSADMIN` program (`CONSolid` utility name and parameters).

JCL Examples

Figure 23 shows example JCL to run the CONSolid utility. In this example, the VTV parameter specifies consolidating VTV100 to VTV200.

```
//CONSOLIDATE  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 CONS VTV(VTV100 - VTV200)
```

Figure 23. `CONSolid` ***utility example: specifying VTVs to consolidate***

Figure 24 shows example JCL to run the CONSolid utility. In this example, the MGMT parameter specifies consolidating VTVs in Management Class PAYRVLT.

```
//CONSOLIDATE  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 CONS MGMT(PAYRVLT)
```

Figure 24. `CONSolid` ***example: specifying Management Class to consolidate***

Consolidation Reports

The consolidation report displays the following messages:

`MIGRATE ONLY FROM VTSS vtssname`

Explanation: The VTV is resident on VTSS `vtssname`.

`REMIGRATE FROM MVC mvcname VIA VTSS vtssname`

Explanation: VTCS is recalling a VTV from MVC `mvcname` to consolidate the VTV.

`VTV vtvname NOT SELECTED; VTV IS SCRATCH`

Explanation: VTCS will not consolidate the specified VTV, which is either scratch or not initialized.

`VTV vtvname NOT SELECTED; VTV ALREADY CONSOLIDATED`

Explanation: The specified VTV is already consolidated.

`VTV vtvname NOT SELECTED; VTV RECORD NOT FOUND`

Explanation: VTCS will not consolidate the specified VTV, which has no record in the CDS.

`VTV vtvname NOT SELECTED; VTV STILL MOUNTED ON DRIVE`

Explanation: VTCS cannot consolidate the specified VTV, which is mounted or in recovery.

`REDRIVING REQUEST BECAUSE OF ERROR`

Explanation: VTCS is retrying an unsuccessful consolidation request.

`CONSOLID CMD PROBLEM DECODING VCI REQUEST FROM HSC`

Explanation: The consolidation failed.

`VTV vtvnumber NOT SELECTED: LIMITED ACCESS TO VTSS`

Explanation: The consolidation request failed because a host not enabled for consolidation (via the `NOMIGRAT` parameter) issued the request.

`MIGRATE NO MVCS AVAILABLE`

Explanation: Sufficient free MVCs are not available to complete the request.

DECOM

DECOM lists the VSM configuration information in the HSC CDS.

Syntax

```
►►────────DECOM──────────┬──────────────────────┬──────►◄
                         └──FLATdd(ddname)──────┘
```

Figure 25. ***DECOM utility syntax***

Parameters

FLATdd
: specified the output destination ddname if a flat file is required.

  ddname
  : the ddname of the flat file included in the JCL.

Interfaces

SWSADMIN utility only.

Usage

Use the DECOM utility to list the VSM configuration stored in the HSC CDS. If you need to rerun the CONFIG utility but no longer have the input file to CONFIG, run the DECOM utility with the FLATdd parameter or to SLSPRINT. You can then edit the output file from DECOM and use it as input to the CONFIG utility. For more information, see “DECOM Output” on page 43.

Hint: The output file contains all statements (including CONFIG) from the original CONFIG input. Also note that the CONFIG utility allows you to respecify existing MVC and VTV ranges, which are also included in the DECOM output. For more information, see “VTVVOL Statement” on page 18 and “MVCVOL Statement” on page 19.

JCL Requirements

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the DECOM report.

SLSIN
: specifies the input to the SWSADMIN program (DECOM utility name and parameters).

JCL Example

Figure 26 shows example JCL to run the DECOM utility with output to flat file CFG22202.

```
//DECOM     EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//CFG22202     DD DSN=FEDB.VSMLMULT.CFG22202,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 DECOM FLATDD(CFG22202)
```

Figure 26. ***Example JCL for the* DECOM *utility***

DECOM Output

FLATDD Parameter Output

Figure 27 shows an example of DECOM output with the FLATDD parameter.

```
CONFIG
GLOBAL        MAXVTV=32000MVCFREE=40
RECLAIM       THRESHLD=70MAXMVC=40  START=35
VTVVOL LOW=905000 HIGH=999999 SCRATCH
VTVVOL LOW=C00000 HIGH=C25000 SCRATCH
VTVVOL LOW=RMM000 HIGH=RMM020 SCRATCH
MVCVOL LOW=N25980 HIGH=N25989
MVCVOL LOW=N35000 HIGH=N35999
VTSS   NAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS18800 DEVNO=8800 CHANIF=0A
       RTD    NAME=VTS18801 DEVNO=8801 CHANIF=0I
       RTD    NAME=VTS18802 DEVNO=8802 CHANIF=1A
       RTD    NAME=VTS18803 DEVNO=8803 CHANIF=1I
       VTD    LOW=8900 HIGH=893F
VTSS   NAME=VTSS2  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
       RTD    NAME=VTS28804 DEVNO=8804 CHANIF=0A
       RTD    NAME=VTS28805 DEVNO=8805 CHANIF=0I
       RTD    NAME=VTS28806 DEVNO=8806 CHANIF=1A
       RTD    NAME=VTS28807 DEVNO=8807 CHANIF=1I
       VTD    LOW=9900 HIGH=993F
```

Figure 27. ***Example DECOM FLATDD output***

SLSPRINT Output

Figure 28 shows an example of DECOM output to SLSPRINT.

```
SWSADMIN (5.0.0)        STORAGETEK VTCS SYTEM UTILITY                 PAGE 0002
TIME 09:07:06 CONTROL CARD IMAGE LISTING
DECOM
SLS1315I SSRDMP.P775644.TESTCDS WAS SELECTED AS THE PRIMARY CONTROL DATA SET
TIME 09:07:06 VTCS DECOMPILE
CONFIG VER(NEW)
GLOBAL       MAXVTV=32000MVCFREE=40
RECLAIM      THRESHLD=70MAXMVC=40  START=35
VTVVOL LOW=905000 HIGH=999999 SCRATCH
VTVVOL LOW=C00000 HIGH=C25000 SCRATCH
VTVVOL LOW=RMM000 HIGH=RMM020 SCRATCH
MVCVOL LOW=N25980 HIGH=N25989
MVCVOL LOW=N35000 HIGH=N35999
VTSS  NAME=VTSS1  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
      RTD    NAME=VTS18800 DEVNO=8800 CHANIF=0A
      RTD    NAME=VTS18801 DEVNO=8801 CHANIF=0I
      RTD    NAME=VTS18802 DEVNO=8802 CHANIF=1A
      RTD    NAME=VTS18803 DEVNO=8803 CHANIF=1I
      VTD    LOW=8900 HIGH=893F
VTSS  NAME=VTSS2  LOW=70 HIGH=80  MAXMIG=3  RETAIN=5
      RTD    NAME=VTS28804 DEVNO=8804 CHANIF=0A
      RTD    NAME=VTS28805 DEVNO=8805 CHANIF=0I
      RTD    NAME=VTS28806 DEVNO=8806 CHANIF=1A
      RTD    NAME=VTS28807 DEVNO=8807 CHANIF=1I
      VTD    LOW=9900 HIGH=993F
```

Figure 28. ***Example DECOM SLSPRINT output***

DISPLAY

`Display` displays the status of the following:

- VTSSs.
- VTDs.
- RTD usage or the status of active or queued processes that use an RTD. Use `Display` to determine the ID of a process you want to cancel with the `CANCEL` on page 11.
- Scratch subpools.
- MVC pools.
- Specific VTVs and MVCs.
- `CONFIG` parameter settings.
- Migrations.
- Tasks.
- Locks.
- Cluster links.
- Clusters.
- VTV replications.
- `U`sage information about a VTCS command or help information about an HSC message (including but not limited to the messages listed in the “HSC Messages for VTCS Events” section of *VTCS Messages*).

Syntax

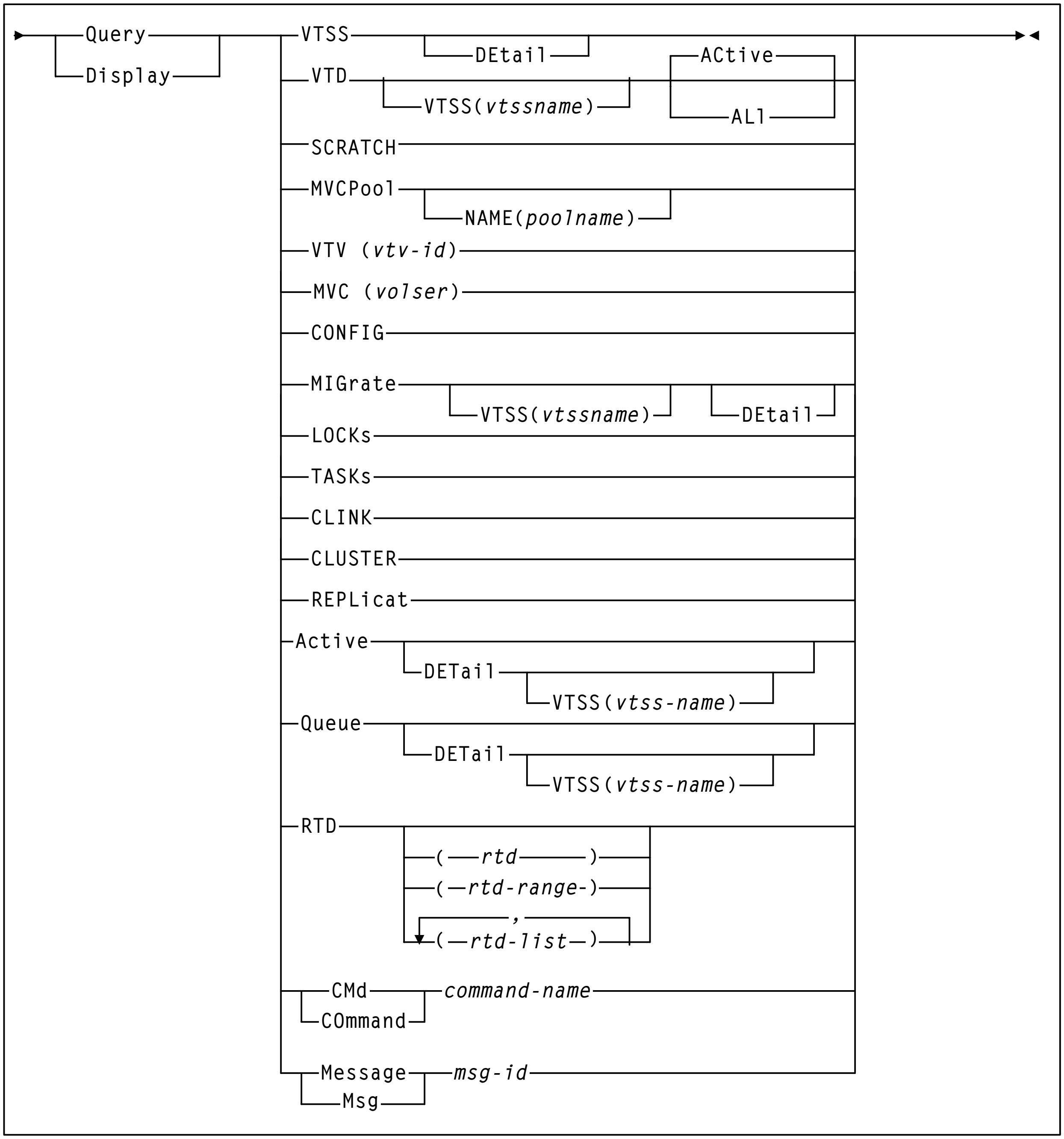


Figure 29. `Query Display` ***syntax***

Parameters

`VTSS`
: display VTSS information.

  `DETail`
  : display detailed host status.

`VTD`
: display VTD information. Lists and ranges of RTDs are limited to 64 items for VSM2s and VSM3s and 256 items for VSM4s.

  `VTSS`
  : display status for the VTDs connected to the specified VTSS.

    `vtss-name`
    : the VTSS name.

  `ACtive | ALl`
  : display status for VTDs that have VTVs mounted (`ACtive`) or all VTDs (`ALl`).

`SCRATCH`
: display scratch subpool information.

`MVCPool`
: display MVC pool information.

  `NAME`
  : display information for the specified Named MVC Pool.

    `poolname`
    : the name of an MVC Pool that you defined on the `MVCPool` control statement; for more information, see "MVCPOOL Control Statement" on page 216. Alternatively, you can specify `ALL` to display information for all Named MVC pools (including the default pool `DEFAULTPOOL`).

`VTV`
: display information about a specific VTV.

  `vtv-id`
  : the ID of the VTV.

`MVC`
: display information about a specific MVC.

  `volser`
  : the volser of the MVC.

`CONFIG`
: display `CONFIG` parameter settings.

`MIGrate`
display migration status.

`DETail`
display migration status by Storage Class.

`VTSS`
the VTSS whose migration status you want to display.

`vtssname`
the VTSS identifier.

`TASKs`
display task status.

`LOCKs`
display lock status.

`CLINK`
display Cluster link status.

`CLUSTER`
display Cluster status.

`REPLicat`
display VTV replication status.

`Active`
display active processes.

`DETail`
display detailed status.

`VTSS`
display processes for the specified VTSS.

`vtss-name`
the VTSS name.

`Queue`
display queued processes.

`DETail`
display detailed status.

`VTSS`
display processes for the specified VTSS.

`vtss-name`
the VTSS name.

`RTD`
display usage information for the specified RTDs. Lists and ranges of RTDs are limited to 8 items for VSM2s and VSM3s and 16 items for VSM4s.

`rtd-id`, `rtd-range`, or `rtd-list`
the unit addresses of one or more RTDs.

`CMd` or `Command`

display syntax and use information for a VTCS command.

`cmd-name`

the command name.

`Msg` or `Message`

display detailed HSC message information.

`msg-id`

the four–digit numerical portion of the message identifier. Leading zeros are not required.

Interfaces

`SWSADMIN` utility and `VT` command **except** for the following, which are only valid as commands:

- `VT Display CMD` or `VT Query CMD`
- `VT Display MSG` or `VT Query MSG`

Usage

Use `DISPLAY` to display the status of the VSM system, including the following:

- The capacity, VTSS count, and operating policies for all VTSSs
- VTD configuration and usage
- RTD usage or the status of active or queued processes that use an RTD.
- Available scratches in each scratch subpool
- The number of free MVCs and reclaim candidates in each ACS (optionally, by Named MVC Pool)
- Information about specific VTVs and MVCs
- `CONFIG` parameter settings
- Migrations
- Tasks
- Locks
- Cluster links
- Clusters
- VTV replications
- Usage information about a VTCS command or help information about an HSC message (including but not limited to the messages listed in the “HSC Messages for VTCS Events” section of *VTCS Messages*.)

Command Examples

To display the number of free MVCs and MVC reclaim candidates in each MVC pool, enter:

```
.VT QU MVCP
```

To display detailed status of all active processes, enter:

```
.VT QU A DET
```

Note: `Display` shows both parent and child requests. All child requests appear immediately after their parent requests. This includes children of children. For example, Figure 30 shows example of the output from the following command:

```
.VT QU A DET
```

FUNCTION	ID	VTV	MVC	RTD	VTSS
DRAIN@	02064	-	-	-	-
MVCDRain	02065=	-	022522	-	-
VTVmover	02066=	-	022522	-	-
Recall	02069=	Y03054	022522	2A07	HBVTSS19
Migrate	02070=	Y05088	022680	2A08	HBVTSS19

Figure 30. ***Example output from a*** `Display Active DETail` ***command: parent and child requests***

In Figure 30, the function DRAIN@ has the parent request ID of 00191 and the subsequent child requests (indicated by the equal sign (=)).

JCL Requirements

The following are the required and optional statements for the `DISPLAY` command JCL:

`STEPLIB`

specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

`SLSPRINT`

specifies the destination for the `DISPLAY` command output.

`SLSIN`

specifies the input to the `SWSADMIN` program (`DISPLAY` command and parameters).

JCL Examples

Figure 31 shows example JCL to display the number of free MVCs and MVC reclaim candidates in each MVC pool.

```
//MVCR      EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  QU MVCP
```

Figure 31. ***Example JCL to display the number of free MVCs and MVC reclaim candidates in each MVC pool***

Figure 32 shows example JCL to display detailed status of all active processes.

```
//MVCR      EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  QU A DET
```

Figure 32. ***Example JCL to display detailed status of all active processes***

Output

Display VTSS Output

Figure 33 shows an example of `Display VTSS` output.

VTSSNAME	CAPACITY(MB)	DBU	HI AMT	LOW AMT	VTV COUNT	MX MT	MN MT	DEF ACS	AUTOMIG	STATE
HBVTSS16	56,209	9	35	30	204	6	1	--		ONLINE-P
HBVTSS17	56,209	7	35	30	218	4	3	02		ONLINE-P
HBVTSS18	N/A	N/A	35	30	N/A	3	1	01		OFFLINE
HBVTSS19	93,184	5	35	30	110	3	1	01		ONLINE

Figure 33. **Example output from** `Display VTSS`

VTSSNAME
the name of the VTSS.

CPCTY(MB)
the total physical capacity in megabytes of the specified VTSS.

DBU
the percentage of disk buffer used of the total buffer capacity.

HI AMT
the high AMT.

LOW AMT
the low AMT.

VTV COUNT
the number of VTVs resident on the VTSS.

MX MT
the current `MAXMIG` value.

MN MT
the current `MINMIG` value.

DEF ACS
the default ACS.

AUTOMIG
indicates which host is performing the auto migration and the threshold to which the VTSS is migrating.

STATE
one of the following global VTSS states (described in Table 8 on page 164) for all hosts:

QUIESCING
Quiescing state.

QUIESCED
Quiesced state.

OFFLINE
Offline state.

OFFLINE-P
Offline pending state.

ONLINE
Online state.

ONLINE-P
Online pending state.

STARTED
The VTSS is initialized and in process of going to the requested state (online, offline, or quiesced).

Display VTSS DEtail Output

Figure 34 shows an example of the additional fields for `Display VTSS DEtail` output.

HOST	VTSSNAME	NOMIGRAT?	NORECLAM?	STATE
EC104	HBVTSS16	Y	Y	ONLINE
EC21	HBVTSS16	N	Y	ONLINE
EC21	HBVTSS17	N	Y	QUIESCED
EC10	HBVTSS17	Y	Y	OFFLINE

Figure 34. ***Example additional output from*** `Display VTSS Detail`

HOST
: the hosts that have access to the VTSSs in the **VTSSNAME** field.

VTSSNAME
: the VTSSs that the hosts in the **HOST** field can access.

NOMIGRAT
: whether `NOMIGRAT` is set on for this host.

NORECLAM
: whether `NORECLAM` is set on for this host.

STATE
: one of the following VTSS states (described in Table 8 on page 164) for this host:

  QUIESCING
  : Quiescing state.

  QUIESCED
  : Quiesced state.

  OFFLINE
  : Offline state.

  OFFLINE-P
  : Offline pending state.

  ONLINE
  : Online state.

  ONLINE-P
  : Online pending state.

  STARTED
  : The VTSS is initialized and in process of going to the requested state (online, offline, or quiesced).

Display VTD Output

Figure 35 shows an example of `Display VTD` output.

DRIVE	LOCATION	VTV	STATUS
A800	HBVTSS16	X00778	MOUNTED
A801	HBVTSS16		AVAILABLE
A802	HBVTSS16		AVAILABLE
A803	HBVTSS16		AVAILABLE

***Figure 35.*Example output from** `Display VTD`

DRIVE
the MSP device address of the VTD.

LOCATION
the VTSS that contains the VTD.

VTV
the VTV volser if **STATUS** is **Mounted.**

STATUS
one of the following VTD statuses:

Mounted
the VTV volser shown in the **VTV** column is mounted on the VTD.

Available
the VTD is available for work.

Display RTD Output

Figure 36 shows an example of `Display RTD` output.

RTD	STATUS	MOUNT	ALLOC	HOST	VTSS
B200	ONLINE/FREE	-	-	-	DHSS16
B201	ONLINE/FREE	-	-	-	DHSS16
0B79	ONLINE/FREE	-	-	-	DHSS16
0B7A	RECALL VTV	DMV051*	DMV051	EC20	DHSS18
1600	MSP1 :MIGRATE	-	-	-	-
1601	MSP1 :MIGRATE	-	-	-	-

Figure 36.* Example output from a** `VT Display RTD` ***command

RTD
the unit address of the RTD.

Status
One of the following RTD statuses:

RECOVER RTD
The RTD is being reset after a problem, a vary, or an initialization.

MIGRATE VTV
The RTD is migrating a VTV.

RECALL VTV
The RTD is recalling a VTV.

UNLOAD MVC
A forced unload of the RTD is occurring.

VTV TRANSFER
The RTD is migrating a VTV before recalling it on another VTSS.

AUDIT MVC
An MVC is being audited.

BUSY
The RTD is busy (non-specific task).

IDLE
An MVC is allocated to the RTD but the MVC is not being used.

ONLINE/FREE
The RTD is online and available.

MAINTENANCE
The RTD is in maintenance mode.

OFFLINE
The RTD is offline and unavailable to all hosts and VTSSs

RECOVERY
The RTD is being reset following an error or a vary online mode.

INITIALIZE
The host is verifying RTD status and availability.

SUSPEND
The RTD operations are suspended. This occurs when two RTDs are connected to two separate ports on the same VSM4 ICE3 card CIP. When one RTD is active, the other is in Suspend mode.

PATH OFFLINE
The RTD status is unknown because the VTSS cannot contact the RTD or if the paths were not correctly configured.

MOUNT

the volser of the MVC currently mounted (an * indicates an in-process mount.

ALLOC

the volser of the MVC allocated for mounting on the RTD.

HOST

the host that currently owns the RTD.

VTSS

the VTSS that is currently connected to the RTD.

Display Active and Display Queue Output

Figure 37 shows an example of `Display Active` output.

MIGRATES=3	RECALLS=2	RECLAIMS=0

Figure 37. Example output from the VT Display ***command (no detail)***

Figure 38 shows example of `Display Active DETail` output.

FUNCTION	ID	VTV	MVC	RTD	VTSS	Task
Reclaim@	00003	-	-	-	-	RCM
.ReclmMVC	00009	-	DMV055	-	-	RCM
. VtvMover	00012	-	DMV055	-	-	RCM
. Migrate	00024	DY069	DMV068	2A00	DHSS16	RTD
Display@	00027	-	-	-	-	DSP

Figure 38. Example output from a `VT Display Active DETail` ***command***

Note: *ABORT appears in the display of reclaim requests if the request has cancelled or abended.

Figure 39 shows an example `Display Queue DETail` output.

FUNCTION	ID	VTV	MVC	RTD	VTSS	Task	REASON
Reclaim@	00003	-	-	-	-	RCM	Child finish
.ReclmMVC	00008	-	DMV053	-	-	RCM	Child finish
. VtvMover	00010	-	DMV053	-	-	RCM	Child finish
. Migrate	00020	-	DMV068	-	DHSS16	DRV	RTD allocation
.ReclmMVC	00009	-	DMV055	-	-	RCM	Child finish
. VtvMover	00012	-	DMV055	-	-	RCM	Child finish
. Recall	00013	DY069	DMV055	2A00	DHSS16	RTD	MVC mount

Figure 39. Example output from a `VT Display Queue DETail` ***command***

The `Function` column shows the hierarchy of requests by indenting. The `VTSS` column shows either a VTSS name or a VTSS list. The `Task` column shows which task is currently servicing the request (same information as reported in `Display Tasks`).

Function

One of the following processes:

Audit#
Audit utility request.

Cancel@
Cancel command.

CONSOLID
`CONSolid` utility subtask.

CONSOLd#
`CONSolid` utility task.

Dismount
VTV dismount.

Display@
Display command.

Drain
Drain VTVs from MVC.

Drain@
Drain command.

Migrate
A child of a **VTVMover** request, which is responsible for the migrate portion of the VTV movement.

Migrate@
Migrate command or utility.

Mig_set@
Set migration threshold command.

Mig_thr@
Migrate to threshold command.

Mount
VTV mount.

MVC_chek
Display MVC.

MVCDrain
There is one **MVCDrain** request per MVC being drained. **MVCDrain**, which is a child request of a **Drain@** request, is responsible for managing the entire drain process for a single MVC.

MVC_inv
Audit of an MVC.

MVCReclm
There is one **MVCReclm** request per MVC being reclaimed. **MVCReclm**, which is a child request of a **Reclaim@** request, is responsible for managing the entire reclaim process for a single MVC.

MVC_upd
Reset MVC status.

QRY/SET@
Display or set command.

Recall
A child of a **VTVMover** request, which is responsible for the recall portion of the VTV movement.

Recall@
Recall command or utility.

Reclaim@
Reclaim command or auto reclaim request.

Scratch
Scratch VTV.

Sel_scr
PGMI select scratch.

Transfer
Transfer VTV between VTSSs.

Unload
Unload MVC from RTD.

Unscratch
Unscratch VTV.

Vary@
Vary RTD.

VTV_chek
Display VTV.

VTSS_inv
Audit of a VTSS.

VTVMover

There is one **VTVMover** request per MVC being drained or reclaimed. This is a child request of either an **MVCDRain** or **MVCReclm** request. This request is responsible for the movement of VTVs from one MVC to another.

VTV_upd

Resynchronize VTV status and CDS.

ID

The process ID, which is a unique number in the range 0 - 65536. When the process ID reaches 65536 it wraps back to zero.

VTV

the volser of the VTV used in the process.

MVC

the volser of the MVC used in the process. **-TERM-** can also appear in this field for reclaim requests. **-TERM-** indicates that the reclaim request is waiting to terminate because it has outstanding child requests, which are usually the outstanding migrate requests.

VTSS

the name of the VTSS that initiated the process.

RTD

the unit address of the RTD used in the process.

REASON

why the request is queued (queued processes only):

CANCELLED

The request is terminating after being cancelled or another failure.

CHILD FINISH

The request has child requests and is waiting for them to finish.

DEVICE LOCK

The device that the request requires is currently locked.

TASK LOCK

The request is waiting for a general task lock to free.

MVC LOCK

The request is waiting for a lock on an MVC to free.

VTV LOCK

The request is waiting for a lock on a VTV to free.

VTD LOCK

The request is waiting for a lock on a VTD to free.

RTD LOCK

The request is waiting for a lock on an RTD to free.

QUEUED

The request is sitting in the input queue of the task.

WAIT RESOURCE
The request is held awaiting a (non-specific) resource becoming available.

RTD ONLINE
The request requires an RTD to be brought online to continue.

RTD ALLOCATION
The request is queued awaiting a RTD to become idle or free.

MVC SELECTION
The request is queued awaiting a MVC or migration slot becoming available.

STEAL A RTD
The request is waiting to steal an RTD allocation from another request.

DBU DROP
The request is currently held because the DBU is high.

MVC MOUNT
The request is waiting for a MVC to be mounted.

Display SCRATCH Output

Figure 40 shows an example of `Display SCRATCH` output.

SUBPOOL-NAME	SCRATCH-COUNT
VIR000	14,364
VIR0002	13,582
VIRTUAL	19,132
VIRTUAL1	9,905

Figure 40. ***Example output from*** `Display SCRATCH`

SUBPOOL-NAME
the name of the scratch subpool. Note that VTCS **does not** display scratch counts for non-subpool VTVs. If there are no VTV subpools defined, VTCS defines the `Virtual Tapes` pool for all VTVs.

SCRATCH-COUNT
the number of available scratch VTVs in the subpool.

Display MVCPool Output

Figure 41 shows an example of `Display MVCPool NAME (POOL1)` output.

	MVCPOOL (POOL1) INFORMATION						
ACS	MEDIA	FREE-MVCS		RECLAIM-MVCS		USED-MVCS	
		VOLS	GB	VOLS	GB	VOLS	GB
00	ECART	120	96	2	0.5	90	45
00	STK1R	30	600	1	3.5	25	350
00	TOTAL	150	696	3	4.0	115	395

Figure 41.Example output from `Display MVCPool NAME(POOL1)`

Figure 42 shows an example of `Display MVCPool` output (no pool name specified).

	MVCPOOL INFORMATION						
ACS	MEDIA	FREE-MVCS		RECLAIM-MVCS		USED-MVCS	
		VOLS	GB	VOLS	GB	VOLS	GB
00	ECART	310	248	4	1.2	100	65
00	ZCART	120	192	1	0.5	250	400
00	TOTAL	430	440	5	1.7	350	465
01	ECART	90	144	15	6.2	322	485
01	ZCART	35	700	3	11.3	43	675
01	TOTAL	125	844	18	17.5	365	1160
NON-LIB	STK2P	22	1100	0	0	12	1565
NON-LIB	TOTAL	22	1100	0	0	12	1565

Figure 42.Example output from `Display MVCPool (no` ***pool name specified)***

ACS
: the ACS containing the MVC pool. **NONLIB** counts are for initialized MVCs that are now outside the library.

MEDIA
: the MVC media type.

FREE-MVCS
: MVCs that have 100% usable space and do not contain any migrated VTVs. The storage shown is the total free space based on media type capacity.

RECLAIM-MVCS
: MVCs eligible for space reclamation by this host. The storage shown is the total wasted space including those MVCs not yet eligible for space reclaim.

USED-MVCS

Initialized MVCs that are partially or completely full.

Display VTV Output

Figure 43 shows an example of `Display VTV` output.

```
VOLSER:                         X52638
VTSS:                           HBVTSS17
MOUNTED:                        8900
UNCOMPRESSED SIZE (MB)          15.30
COMPRESSED SIZE (MB)            3.90
MAXIMUM VTV SIZE (MB)           800
CREATION DATE:                  2003JUN18 10:44:08
LAST USED DATE:                 2003JUL15 03:10:05
MVC(S):                         EVS124    TIM744   EVS122    TIM778
BLOCK-ID(S):                    0140065B  0140079  024008ED  0240089F
MANAGEMENT CLASS:               MIG22
STATUS:                         INITIALIZED
                                RESIDENT
                                MIGRATED 4 VTV COPIES
```

Figure 43. ***Example output from*** `Display VTV`

VOLSER

the volser of the VTV specified in the query.

VTSS

the VTSS where the VTV resides.

MOUNTED

if the VTV is mounted on a VTD, the VTD unit address is displayed.

UNCOMPRESSED SIZE(MB)

the uncompressed size of the VTV (MB).

COMPRESSED SIZE (MB)

the compressed size of the VTV (MB).

MAXIMUM VTV SIZE (MB)

the maximum size of the VTV (400 or 800).

CREATION DATE

the date when the VTV was created in the VTSS.

LAST USED DATE

the date when the VTV was last mounted in the VTSS.

MANAGEMENT CLASS

the VTV's Management Class.

MVC(S)
: the MVC(s) where the VTV resides. This entry only appears when the VTV is migrated.

BLOCK-ID
: the logical block ID of the beginning of the VTV on the MVC.

STATUS
: one or more of the following statuses:

  CONSOLIDATED
  : VSM has consolidated the VTV.

  DUPLEXED
  : The DUPLEX attribute has been assigned to this VTV. When VSM migrates the VTV, a copy will be written to two MVCs.

  INITIALIZED
  : VTCS has used the VTV at least once.

  MIGRATED
  : VSM has migrated the VTV.

  RESIDENT
  : The VTV is resident on the VTSS.

  SCRATCH
  : The VTV is in scratch status.

  UNINITIALIZED
  : The VTV has been defined via the `CONFIG` utility, but has not ever been used.

  REPLICATION REQUIRED
  : This VTV should be replicated and is currently waiting for replication.

  REPLICATION STARTED
  : Replication is active for this VTV but not yet complete.

  REPLICATION COMPLETE
  : A fully replicated copy of this VTV is now resident in the Secondary VTSS.

Display MVC Output

Figure 44 shows an example of `Display MVC` output.

```
VOLSER:                          9940245
VTV COUNT:                       1
MEDIA:                           STK2P
ACSID:                           00
SIZE(MB):                        10240
%USED:                           0.01
%FRAGMENTED:                     5.66
%AVAILAABLE                      94.33
%USABLE                          0.00
TIMES MOUNTED:                   4
LAST MOUNTED                     2003APR21 05:53:28
OWNER:                           LOCAL
VTSS:                            HBVTSS08
MVCPOOL:                         POOL1
SECURITY ACCESS:                 UPDATE
STATUS:                          INITIALIZED
                                 IN ERROR
                                 DATA CHECK
```

Figure 44. ***Example output from*** `Display MVC`

VOLSER
: the volser of the MVC.

VTV COUNT
: the number of VTVs on the MVC.

MEDIA
: the volume media.

ACSID
: the ACS that contains the MVC.

SIZE(MB)
: the size of the MVC in megabytes

%USED
: the percentage of the MVC used by valid VTVs.

%FRAGMENTED
: the percentage of the MVC that has invalid VTV space that is not available for use until it is reclaimed or the MVC is drained.

%AVAILABLE

the percentage of the MVC that is physically available for use.

%USABLE

the percentage of space on the MVC that can be used by VTCS. This may be zero even if there is still space physically available. For instance, if the VTV per MVC limit is reached then the %Usable will be reported as 0%. Similarly, if an error has been reported against an MVC then VTCS will not use this MVC for output and the %Usable will be reported as %0.

TIMES MOUNTED

the number of times the MVC has been mounted for writing or reading since it was added to the MVC inventory.

LAST MOUNTED

the date and time the MVC was last mounted.

OWNER

the Storage Class that owns the MVC.

VTSS

the last VTSS that wrote to the MVC. `CONSOLIDATE` appears in this field for consolidated VTVs.

MVCPOOL

either an MVC Pool Name (including `DEFAULTPOOL`) or **NO** if the MVC is not defined on an `MVCPool` statement.

SECURITY ACCESS

VTCS permissions for the MVCs defined in an `MVCPOOL` statement (**UPDATE**, **NO UPDATE**, or **NO PROFILE**).

STATUS

one of the following statuses:

BROKEN

This is a generic error that indicates the MVC, drive, or combination of the two has a problem. VTCS attempts to de-preference MVCs with this state. **In general,** to clear this state:

- If the MVC caused the problem, use a `DRAIN(EJECT)` command to remove the MVC from service.
- If the RTD caused the problem, use the `MVCMAINT` utility to reset the MVC state.

Note also that one or more of the following messages is issued for `BROKEN` status: `SLS6686`, `SLS6687`, `SLS6688`, `SLS6690`. For detailed recovery procedures for these messages, see *VTCS Messages and Codes*.

DATA CHECK

A data check condition has been reported against this MVC. VTCS attempts to de-preference MVCs with this state. To clear this state:

- If all VTVs on the MVC are duplexed, use `MVCDRain` on the MVC without the `Eject` option. This recovers all VTVs and removes the MVC from service.
- If all VTVs on the MVC are not duplexed, VTCS `AUDIT` the MVC. The audit will probably fail. After the audit, do an `MVCDRAIN` (no eject). This recalls the VTVs before the data-check area in ascending block-id order and the VTVs after the data-check area in a descending block-id order. Processing the VTVs in this sequence ensures that VTCS recovers as many VTVs as possible from the media. You then need to recreate the data for any VTVs still on the MVC.

DRAINING

The MVC is either currently being drained or has been the subject of a failed `MVCDRain`.

IN ERROR

An error occurred while the MVC was mounted.

INITIALIZED

the MVC has been initialized.

LOST - FAILED TO MOUNT

VTCS attempted to mount an MVC and the mount did not complete within a 15-minute time-out period. VTCS is attempting to recover from a situation that may be caused by hardware problems, HSC problems, or by the MVC being removed from the ACS. VTCS attempts to de-preference MVCs with this state.

If VTCS does perform a subsequent successful mount of an MVC with `LOST(ON)` state, VTCS sets the state to `LOST(OFF)`.

Determine the cause of the error and fix it. You can also use the VTCS `MVCMAINT` utility to set `LOST(OFF)` for the following events:

- `LOST(ON)` was set due to LSM failures or drive errors that have been resolved
- `LOST(ON)` was set because the MVC was outside the ACS and has been reentered.

MARKED FULL

The MVC is full and is not a candidate for future migrations.

MOUNTED

The MVC is mounted on an RTD.

NOT-INITIALIZED

The MVC has been defined via the `CONFIG` utility, but has not ever been used.

READ ONLY

The MVC has been marked read-only because of one of the following conditions:

- The MVC being the target of an export or consolidation process. The read-only state protects the MVC from further updates.
- The MVC media is set to file protect. Correct the error and use the `MVCMAINT` utility to set `READONLY(OFF)`.
- The MVC does not having the appropriate SAF rules set to enable VTCS to update the MVC. Correct the error (for more information, see “Defining a Security System User ID for HSC, SMC, and VTCS” in Chapter 2 of *VTCS Installation and Configuration Guide*) and use the `MVCMAINT` utility to set `READONLY(OFF)`.

BEING AUDITED

The MVC is either currently being audited or has been the subject of a failed audit. If the audit failed, VTCS will not use the MVC for migration. To clear this condition, rerun the `AUDIT` utility against this MVC.

LOGICALLY EJECTED

The MVC has either been the subject of an `MVCDRain Eject` or the MVC was ejected for update by a `RACROUTE` call. The MVC will not be used again for migration or recall. To clear this condition, use `MVCDRain` against the MVC without the `Eject` option.

RETIRED

The MVC is retired. VTCS will recall from, but not migrate to, the MVC. Replace the MVC as soon as possible.

WARRANTY HAS EXPIRED

The MVC’s warranty has expired. VTCS continues to use the MVC. You should start making plans to replace the MVC when it reaches Retired state.

INVALID MIR

VTCS has received status from an RTD to indicate the MIR (media information record) for a 9x40 media is invalid. An invalid MIR does not prevent access to data but may cause significant performance problems while accessing records on the tape. The MVC is not capable of high-speed searches on areas of the tape that do not have a valid MIR entry.

VTCS attempts to de-preference MVCs with this condition. For recalls, if the VTV resides on multiple MVCs, VTCS selects MVCs with valid MIRs ahead of MVCs with invalid MIRs. VTCS avoids using MVCs with invalid MIRs for migration, unless the migration is at the beginning of the tape. Migrating from the beginning of tape will correct the MIR.

VTCS detects the invalid MIR condition at either mount time or dismount time. If detected at mount time and the operation can be completed with another MVC, VTCS dismounts the first MVC and selects the alternate MVC. **Note that** VTCS has only a limited ability to switch to an alternate MVC. That is, it is mainly used for migrate and virtual mount.

For MVCs with invalid MIRs, determine the cause of the error, which may be caused by media or drive problems, and fix the error.

To recover an MVC with an invalid MIR, you simply need to read the MVC to the end of the tape, which can be done via a VTCS audit. If the media is the problem, run an `MVCDRAIN EJECT` to recall the VTVs and cause the MVC to be removed from the MVC pool.

Display CONFIG Output

Figure 45 shows an example of `Display CONFIG` output.

```
MAXVTV              MVCFREE          VTVATTR          RECALWER
4000                10               SCRATCH          YES

CDS LEVEL SUPPORT:  V5               V5.1             V6               V6+
                                                                       *

RECLAIM:            THRESHOLD        MAXMVC           START            CONMVC
                    30               10               10               2

VTSSNAME                    AUTO MIGR THR               MIGR TASKS       DEFAULT          RETAIN
                    LOW              HIGH             MIN   MAX               ACS
HBVTSS16            60               80               1     1                 FF               10
HBVTSS17            60               80               1     4                 02               10
HBVTSS18            60               80               4     4                 01               10
HBVTSS19            60               80               1     1                 01               10

VTSSNAME            RTD NAME         RTD DEVNO        RTD TYPE         RTD-ACS            RTD-CHANIF
HBVTSS16            SS162A00         2A00             9490             00                     0A
HBVTSS16            SS162A01         2A01             9490             00                     0K
HBVTSS16            SS162A02         2A02             9490             00                     0I
HBVTSS16            SS162A0C         2A0C             9840             02                     1C
```

Figure 45. ***Example output from*** `Display CONFIG`

MAXVTV
the `GLOBAL MAXVTV` setting.

MVCFREE
the `GLOBAL MVCFREE` setting.

VTVATTR
the `GLOBAL VTVattr` setting (**SCRATCH** or **ALLmount**).

RECALWER
the `GLOBAL RECALWER` setting (**YES** or **NO**).

CONMVC
the `RECLAIM CONMVC` setting.

CDSLEVEL SUPPORT

the VTCS level(s) that can access the active CDS. For more information, see VTCS Installation and Configuration Guide, Chapter 4, “Reconfiguring NCS,” “Converting the Formatted CDS to VSM Extended Format.”

THRESHOLD

the `RECLAIM THRESHLD` setting.

MAX MVC

the `RECLAIM MAXMVC` setting.

START

the `RECLAIM START` setting.

VTSSNAME

the VTSS identifiers (`VTSS NAME` settings).

AUTO MIGR THR, LOW

The low automatic migration threshold setting (LAMT) for the VTSS.

AUTO MIGR THR, HIGH

The high automatic migration threshold setting (HAMT) for the VTSS.

MIGR TASKS, MIN

The minimum number of concurrent automatic migration tasks setting (`MINMIG`) for the VTSS.

MIGR TASKS, MAX

The maximum number of concurrent automatic migration tasks setting (`MAXMIG`) for the VTSS.

DEFAULT ACS

The default ACS setting (`DEFLTACS`) for the VTSS.

DEVNO

the RTD MSP device numbers for the VTSS (`RTD DEVNO` settings).

RTD TYPE

the RTD type.

ACS

the ACS that contains the RTD.

RETAIN

the `VTSS RETAIN` setting.

VTSSNAME

the VTSS identifiers (`VTSS NAME` settings) of the VTSSs connected to the RTD.

RTD NAME
the RTD names for the VTSS (`RTD NAME` settings).

RTD-CHANIF
the RTD channel interface (`RTD CHANIF` settings).

Display MIGrate Output

Figure 46 shows an example of `Display MIGrate` output.

```
VTSSNAME: HBVTSS16    ACTIVE MIGRATION TASKS: 4
IMMEDIATE MIGRATE: MAX WAIT: 5 MINUTES
AUTO MIGRATE: HOST: EC20MIGRATION TARGET: 70%
```

Figure 46. ***Example output from*** `Display MIGrate`

ACTIVE MIGRATION TASKS
the total number of migration tasks (automatic, immediate, and migrate-to-threshold).

IMMEDIATE MIGRATE
either **Not active** if there are no current or pending immediate migrations or the maximum time that any VTV has been waiting for immediate migration.

☞ **Note:** This field only shows status for the LPAR on which the query was issued.

AUTO MIGRATE
either **Not active** or the name of the host and migration target (LAMT or specified threshold for a migration-to-threshold) if auto migration is active on any host.

Display MIGrate DEtail Output

Figure 47 shows an example of the additional fields from `Display MIGrate DEtail` output.

```
STORAGE      ACS    MAX      ONL RTD    ACTIVE     AUTO     IMMED    WEIGHT     TIMES
CLASS               TASKS    TASKS      TASKS      ?        ?        PERCENT    SKIPPED
HBVTSS16     00     4        4          2          Y        N        50         0
HBVTSS16     01     2        1          1          Y        N        25         0
STORCL100    00     5        4          1          Y        N        15         0
!ERROR       01     3        3          0          Y        N        10         1
** Error time:  2003Aug29 08:58:39 Reason:  All RTDs offline
```

Figure 47. Example additional output from `Display MIGrate DEtail`

STORAGE CLASS
the Storage Class associated with the migration.

Note: If you do not explicitly assign a Storage Class, an MVC's default Storage Class is the name of the last VTSS that wrote to the MVC for reclamation or migration and this class has the VTCS default media selections. To change these defaults, create a Storage Class with the VTSS name and specify the desired media selection order.

ACS
the ACS defined for the Storage Class.

MAX TASKS
the maximum number of RTD tasks based on the Storage Class and RTD configuration definitions.

ONL RTD TASKS
the maximum number of tasks for those RTDs that are actually online (**MAX TASKS** minus the number of offline RTDS).

ACTIVE TASKS
the number of migration tasks currently active for the Storage Class.

AUTO ?
indicates whether the Storage Class contains automatic migration VTVs.

IMMED ?
indicates whether the Storage Class contains immediate migration VTVs.

WEIGHT PERCENT
the priority of the Storage Class compared to other Storage Classes for the VTSS. Storage Classes with higher priorities are assigned a greater proportion of migration tasks.

TIMES SKIPPED

the number of times the scheduling process skipped this Storage Class.

Note: Any Storage Classes in error (shown as `!ERROR`) also display an additional line indicating the last time the error was detected and the type of error (no RTDs available, all RTDs offline, no MVCs available).

Display TASKs Output

Figure 48 shows an example of `Display TASKs` output.

TASK	TASK	VTSS	RTD	CURRENT	WAITQ	PENDQ
NBR	TYPE			PROCESS	COUNT	COUNT
00O	DSP			518		
001	SS	HBVTSS16	SS16B200			
002	RTD	HBVTSS16	SS16B201			
003	RTD	HBVTSS16	SS160B79			
004	RTD	HBVTSS16	SS160B7A		1	
005	RTD	HBVTSS16	SS160B7C			

Figure 48. ***Example output from*** `Display TASKs`

TASK

the task number for each task on the current host.

TYPE

the task type.

VTSS

the VTSS name.

RTD

the RTD name for RTD tasks.

CURRENT PROCESS

the current process ID.

WAITQ COUNT

the count of requests waiting for locks.

PENDQ COUNT

the count of pending requests.

Display LOCKs Output

Figure 49 shows an example of `Display LOCKs` output.

SLOT	OWNING	TASK	TASK	VTD	MVC	VTV	WAITING	WAITING
ID	HOST	NBR	TYPE				HOST	TASK
002	EC21	006	RTD		EVS101		EC10	007
003	EC20	010	RTD		EVS145	X15328		
004	EC20	010		A91E		X153234		

Figure 49. ***Example output from*** `Display LOCKs`

SLOT ID
Slot ID of the lock within the lock buffer.**OWNING HOST**
the host that owns the lock.

TASK NBR
the task number associated with the lock.

TASK TYPE
the task type.

VTD
the associated VTD address on the issuing host.

MVC
the locked MVC.

VTV
the locked VTV.

WAITING HOST
the host waiting for the lock or **ALL** if multiple hosts are waiting.

WAITING TASK
the task waiting for the lock or **ALL** multiple tasks are waiting.

Display CLINK Output

Figure 50 shows an example of `Display CLINK` output.

VTSS	CLINK	STATUS	USAGE	HOST
HBVTSS19	7	ONLINE	REPLICATING	EC21
	6	ONLINE	FREE	
	5	ONLINE	FREE	
	4	ONLINE	FREE	

Figure 50. ***Example output from*** `Display CLINK`

VTSS
the Primary or Sending VTSS name.

CLINK
the link ID.

STATUS
one of the following link statuses:

Maint
The link has failed or it has been varied into maintenance mode.

Offline
The link is offline and unavailable to all hosts and VTSSs.

Online
Available for replication.

P_offlne
The link is pending offline

P_online
The link is pending online

Recovery
The link is being reset following an error or a vary online operation.

Unusable
Not available for replication due to hardware errors or assigned-elsewhere conditions.

USAGE

one of the following link usages:

Assigned

Link is assigned to the host in the HOST field but is not currently replicating. This usage occurs when VTCS is starting or terminating link use or is attempting error recovery on the link after a replication failure.

Free

Link is idle (not doing replications).

Replicating

Link is actively doing replications.

HOST

the host that the link is assigned to.

Display CLUSTER Output

Figure 51 shows an example of `Display CLUSTER` output.

NAME	VTSS	STATE	DIRECTION	VTSS	STATE	MODE
CLUSTER1	HBVTSS16	ONLINE	------>	HBVTSS17	ONLINE	FULL-FUNCTION
CLUSTER2	HBVTSS18	ONLINE	<------	HBVTSS19	QUIESCED	DEGRADED
CLUSTER3	HBVTSS20	ONLINE	<------	HBVTSS21	ONLINE	FULL-FUNCTION
CLUSTER4	HBVTSS22	QUIESCED	<----->	HBVTSS23	ONLINE	DEGRADED

Figure 51. Example output from `Display CLUSTER`

NAME
the Cluster name.

VTSS
one of the VTSS in a Cluster.

DIRECTION
One of the following:

----->

OR

<-----
Indicates the direction of VTV replication in uni-directional Cluster. VTVs can only be replicated from the Sending to the Receiving VTSS.

<----->
Indicates that the VTSSs are configured as a Bi-Directional (Peer-to-Peer) Cluster. VTVs can be replicated from either VTSS to the other.

STATE
one of the following VTSS states:

QUIESCING
Quiescing state.

QUIESCED
Quiesced state.

OFFLINE
Offline state.

OFFLINE-P
Offline pending state.

ONLINE
Online state.

ONLINE-P
Online pending state.

STARTED

The VTSS is initialized and in process of going to the requested state (online, offline, or quiesced).

MODE

one of the following Cluster operating modes:

FULL-FUNCTION

Both VTSSs in the Cluster are online to VTCS. Production workload can go to either VTSS, but in the case of a Uni-Directional (Primary/Secondary) Cluster, VTVs can only be replicated from the Sending VTSS.

DEGRADED

One of the two VTSSs in a Bi-Directional (Peer-to-Peer) Cluster is either offline or quiesced. Production workload can go the remaining online VTSS. VTVs requiring replication, however, are allocated to the remaining VTSS only if no other Full-Function clusters are available and suitable. In this case, replicate VTVs are migrated immediately with keep and queued for replication when the other VTSS comes online.

When the other VTSS comes online, VTCS reconciles the contents of both VTSSs.

DEGRADED SECONDARY

The Primary is online to VTCS and the Secondary is either offline or quiesced. Workload can run on the Primary. VTVs requiring replication, however, are allocated to the Primary only if no other Full Function Clusters are available. In this case, Replicate VTVs are migrated immediately with keep and are queued for replication, which occurs when the Secondary comes online.

DEGRADED PRIMARY

The Secondary is online to VTCS and the Primary is either offline or quiesced. Workload can run on the Secondary. VTVs requiring replication, however, are allocated to the Secondary only if no other Full Function Clusters are available. When the Primary comes back ONline, VTCS reconciles the contents of the Primary and Secondary.

NON-OPERATIONAL

No workload is possible on this Cluster.

Display REPLicat Output

Figure 52 shows an example of `Display REPLicat` output.

VTSS	HOST	QDEPTH
HBVTSS19	EC10	0
	EC20	0
	EC21	1
	ECCL	0
	ECCY	1
	EC31	0

Figure 52. Example output from `Display REPLicat`

VTSS
: the Primary VTSS name.

HOST
: the hosts attached to the Primary VTSS.

QDEPTH
: the number of VTVs waiting to be replicated.

EXPORT

EXPORT consolidates VTVs (if required) and creates a manifest file that lists the VTVs and MVCs available for export from a VSM system.

Note: EXPORT is valid only if FEATures VSM(ADVMGMT) is specified; for more information, see “FEATURES Control Statement” on page 194.

Syntax

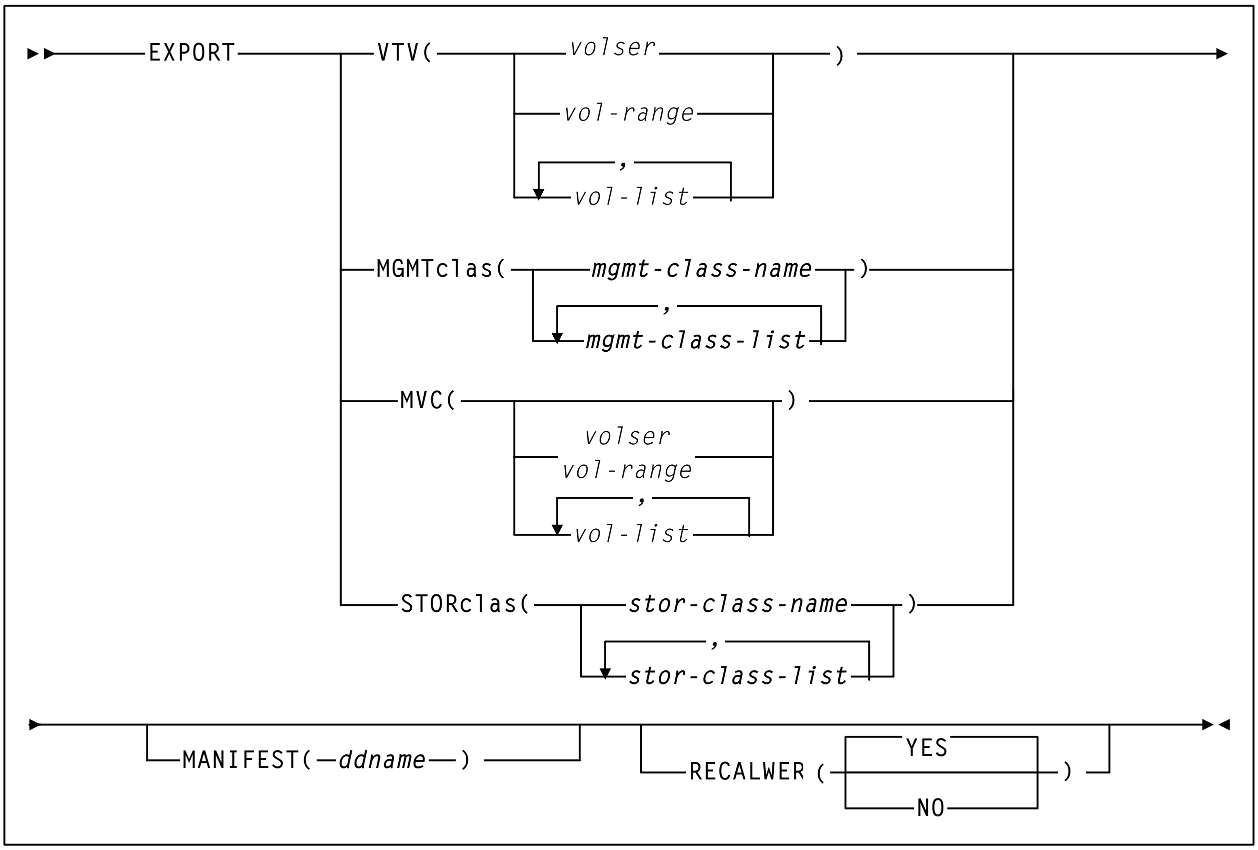


Figure 53. EXPORT ***utility syntax***

Parameters

`VTV`

specifies one or more VTVs to consolidate for export.

`volser, vol-range` **or** `vol-list`

the volsers of one or more VTVs. You can specify an unlimited number of VTVs.

`MGMTclas`

specifies one or more Management Classes that determine one or more VTVs to consolidate for export.

`mgmt-class-name` | `mgmt-class-list`

the names of one or more Management Classes that you defined on the `MGMTclas` control statement; for more information, see “MGMTCLAS Control Statement” on page 204.

`MVC`

specifies one or more MVCs for export.

`volser, vol-range` or `vol-list`

the volsers of one or more MVCs.

`STORclas`

specifies one or more Storage Classes that determine one or more MVCs for export.

`stor-clas-name` | `stor-clas-list`

the names of one or more Storage Classes that you defined on the `STORclas` control statement; for more information, see “STORCLAS Control Statement” on page 219.

`MANIFEST`

specifies the output destination ddname of the manifest file.

`ddname`

ddname of the manifest file. The default is `MANIFEST`.

`RECALWER`

specifies whether VTCS recalls VTVs with read data checks.

`YES`

recall VTVs with read data checks (the default).

`NO`

Do not recall VTVs with read data checks.

Interfaces

`SWSADMIN` utility only.

Usage

Use the EXPORT utility to create a manifest file that lists the VTVs and MVCs (which are marked readonly) available for export from a VSM system. You can create this manifest file by specifying either:

- The VTV or MGMT parameters, which select the VTVs for export and consolidates the selected VTVs. These type of exports require that HSC is running and the CDS is active.
- The MVC or STOR parameters, which select the MVCs (and the VTVs they contain) for export. These type of exports can be run when HSC is down and the CDS is inactive. For example, if you lose all resources at the source VSM system except a copy of the CDS and MVCs containing all the source system's VTVs, you can run EXPORT against the CDS copy at the target VSM system to create a manifest file, then do an import to recreate the source system resources.

Note: The manifest file includes a checksum, which is written at the end of the file, and which covers the entire file. If a manifest file is changed, including writing to it with DISP=MOD, the checksum is invalid and the manifest file is unusable for import.

Note: An export also generates MVC summary and detail reports. For more information, see "MVCRPT" on page 121.

Optional and Required JCL

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules (required).

manifest file DD
: DD statement for the manifest file (optional).

SLSPRINT
: specifies the destination for the EXPORT report (required).

SLSIN
: specifies the input to the SWSADMIN program (EXPORT utility name and parameters) (required).

JCL Examples

Figure 54 shows example JCL to run the `EXPORT` utility. In this example, the `VTV` parameter specifies consolidating `VTV100` to `VTV200` to MVCs for export. The output manifest file is `REMOTE1`.

Figure 54. `EXPORT` ***utility example: specifying VTVs to consolidate for export***

```
//EXPORT  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//REMOTE1      DD DSN=FEDB.VSMLMULT.REMOTE1,DISP=(,CATLG,DELETE),
//              UNIT=SYSDA,SPACE=(CYL,(1,1),RLSE),
//              DCB=(RECFM=FB,LRECL=80,BLKSIZE=27920)
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 EXPORT VTV (VTV100 - VTV200) MANIFEST(REMOTE1)
```

Figure 55 shows example JCL to run the `EXPORT` utility. In this example, the `STOR` parameter specifies making available for export the MVCs in Storage Class `REMOTE`. The output manifest file is `REMOTE2`.

```
//EXPORT  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//REMOTE2      DD DSN=FEDB.VSMLMULT.REMOTE2,DISP=(,CATLG,DELETE),
//              UNIT=SYSDA,SPACE=(CYL,(1,1),RLSE),
//              DCB=(RECFM=FB,LRECL=80,BLKSIZE=27920)
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 EXPORT STOR (REMOTE) MANIFEST(REMOTE2)
```

Figure 55. `EXPORT` ***utility example: specifying Storage Class to determine MVCs for export***

IMPORT

IMPORT imports VTVs and MVCs listed on a manifest file into a VSM system.

Note: IMPORT is valid only if FEATures VSM(ADVMGMT) is specified; for more information, see “FEATURES Control Statement” on page 194.

Syntax

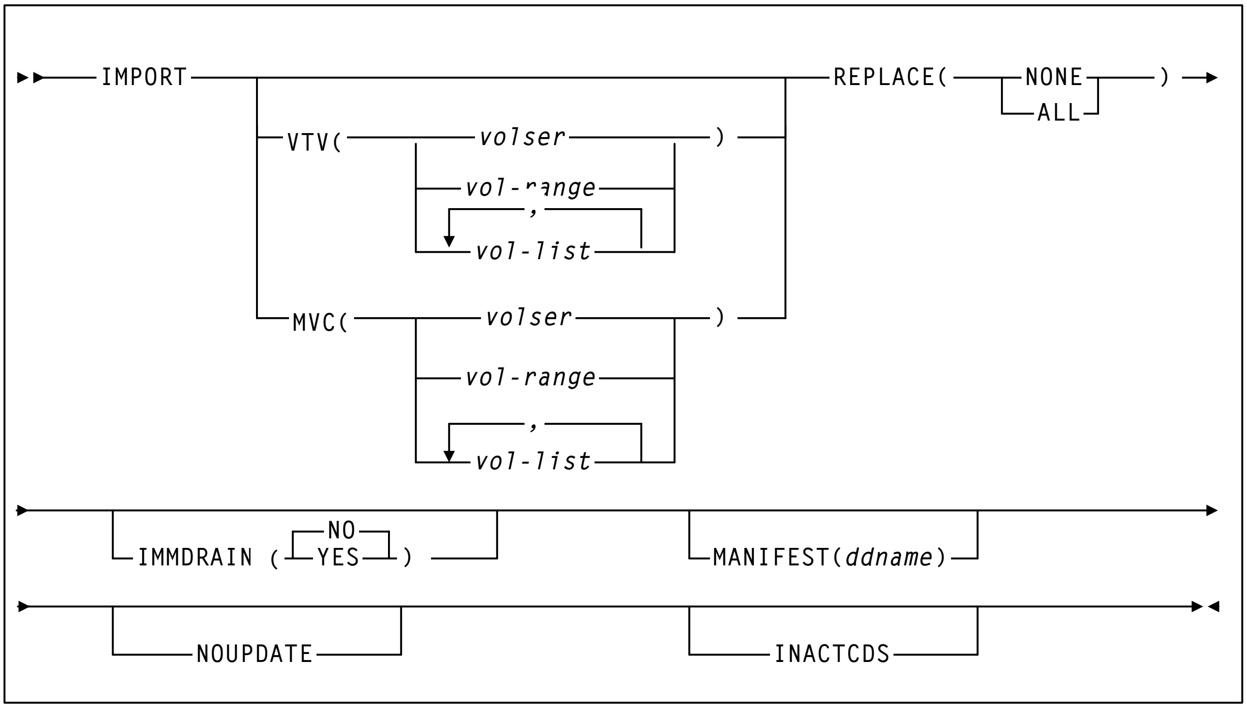


Figure 56. IMPORT ***utility syntax***

Parameters

`VTV`

specifies one or more VTVs to import.

`volser`, `vol-range` **or** `vol-list`

the volsers of one or more VTVs.

`MVC`

specifies one or more MVCs to import.

`volser`, `vol-range` or `vol-list`

the volsers of one or more MVCs.

`REPLACE`

specifies whether VSM replaces the VTV record in the target CDS. There is no default; you must specify a value.

`NONE`

Do not replace the VTV record. VTCS only creates new records for VTVs that are not duplicates and replaces records for VTVs not initialized in the target CDS.

`ALL`

Replace any duplicate VTV records in the target CDS.

Caution: Ensure that you actually want to replace duplicate VTV records in the target CDS before you specify the `ALL` parameter! You may want to do a "validate" run with `NOUPDATE` to see which VTV records will be replaced.

Also note that if a VTV record is replaced, all existing VTSS and MVC copies of the VTV are invalidated.

Finally, you cannot import an MVC if the target CDS records show that the MVC contains VTVs, even if you specify `REPLACE(ALL)`. In this situation, you must first drain (with eject) the MVC on the target system and eject it from the ACS. You can then import the MVC that you exported from the source system.

`IMMDRAIN`

specifies whether VSM will immediately drain imported MVCs.

`NO`

Do not drain MVCs (the default).

`YES`

Drain MVCs.

`MANIFEST`

specifies the input ddname of the manifest file.

`ddname`

ddname of the manifest file. The default is `MANIFEST`.

`NOUPDATE`

specifies that VSM does not update the CDS, validates the import operation, and writes information messages to the job log.

`INACTCDS`

specifies that the import uses a different CDS from the CDS currently active on the HSC system where you are running the import job. You specify the alternate CDS in the SWSADMIN JCL. **Note that** if HSC is down on the system where you are running the import job, the CDS on that system is assumed to be inactive, so you do not need to specify `INACTCDS`.

Interfaces

`SWSADMIN` utility only.

Usage

Use the `IMPORT` utility to import VTVs and MVCs listed on a manifest file into a VSM system. Typically, you do a “validation” import run, as shown in Figure 57 on page 92, followed by an actual import, as shown in Figure 58 on page 92. Imported MVCs are readonly; for information on making them writable, see “MVCMAINT” on page 108.

Note that you can specify a merged manifest file or multiple manifest files.

Note: For each VTV imported, the only MVC copies that will be created will be for MVCs that have been exported and imported via the same statements.

This has a particular significance when importing Duplexed VTVs. Such a VTV will only have copies on both MVCs after the import if both MVCs are present in the same Manifest file and imported as a result of the same `IMPORT` statement.

Note: An import also generates MVC summary and detail reports. For more information, see “MVCRPT” on page 121.

Importing to an Active or Inactive CDS

As described in "INACTCDS" on page 88, you can import to an inactive CDS in two ways: (1) HSC is down on the system where you are running the import job and the CDS is inactive. This **does not** require you to specify the INACTCDS parameter or (2) You specify the INACTCDS parameter and on the SWSADMIN JCL you specify an alternate CDS (not the CDS being used by HSC/VTCS on the system where you are running the import job).

How does import to an inactive CDS work? Basically, as follows:

1. The CDS is reserved **for the duration of the import** to prevent HSC (on any host) from using the CDS while the import is active. When it starts, the import job checks that the CDS is not in use by HSC on any host. The reserve also ensures that HSC is not started while the import job is running. Additionally the Utilities Reserve (major name STKENQNM, minor name -SLUDBRSV) is obtained **for the duration of the import**, which stops other utilities from running against any CDS while import is running.
2. All MVC/VTV records from the CDS are copied to a data space.
3. The data space is updated with the imported MVCs/VTVs.
4. All MVC/VTV updates are written back to the CDS from the data space.
5. The CDS is Released, as is the Utilities Reserve.

What are guidelines for choosing to import to an active vs. an inactive CDS? **In general**, if you are importing a small amount of data to a large CDS, import to an active CDS runs faster. If you are importing a large amount of data, **generally speaking**, import to an inactive CDS runs faster. Table 3 summarizes the differences between importing to an active or inactive CDS.

Table 3. Comparison: Importing to Active or Inactive CDS

Import to Active CDS	**Import to Inactive CDS**
Requires HSC/VTCS to be active on the system where the import job runs. HSC/VTCS must be at FULL service mode (that is, HSC **cannot** be at BASE mode).	If `INACTCDS` is not specified, HSC should be inactive on the system where the import job runs. If `INACTCDS` is specified, HSC can be inactive or active and at FULL service level on the system where the import job runs. If HSC is active, it must be using a different CDS than the one used by import.
The IMPORT JCL must either **not** specify a CDS or must specify the same CDS as is used by HSC/VTCS on the system where the import job runs.	The IMPORT JCL **must** specify (via `//SLSCNTL` and, if appropriate, `//SLSCNTL2` and `//SLSSTBY`) the CDS that is to be updated with the imported data. The specified CDS **cannot** be in use by HSC on any host.
Supports `IMMDRAIN(YES)`.	Does NOT support `IMMDRAIN(YES)` because the import job requires an active HSC/VTCS, using the same CDS as used by IMPORT, to drain an MVC.
Deletes, from the relevant VTSS, any VTVs that are being imported and replaced that were resident before the import began.	Does **not** delete, from the relevant VTSS, any VTVs that are being imported and replaced that were resident before the import began. VTCS issues a message that the deletion cannot be performed. An Audit of the relevant VTSS will be required to resynchronize the VTSS and the CDS.
Writes SMF records.	Does **not** write SMF records.

JCL Requirements

`STEPLIB`
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

manifest file `DD`
: DD statement for the manifest file.

`SLSPRINT`
: specifies the destination for the `IMPORT` report.

`SLSIN`
: specifies the input to the `SWSADMIN` program (`IMPORT` utility name and parameters).

`SLSCNTL`
: Specifies the Primary CDS. **Not required** if importing to the CDS being used by the HSC/VTCS system that is active on the system where the import job runs. Otherwise it is required, to specify the Primary CDS to be updated by import.

  If you specify the `INACTCDS` parameter, this CDS must be **different** from the CDS being used by HSC/VTCS (if active).

`SLSCNTL2`
: Specifies the Secondary CDS, if it exists. **Not required** if importing to the CDS being used by the HSC/VTCS system that is active on the system where the import job runs. Otherwise it is required, to specify the Secondary CDS to be updated by import.

  If you specify the `INACTCDS` parameter, this CDS must be **different** from the CDS being used by HSC/VTCS (if active).

`SLSSTBY`
: Specifies the Standby CDS, if it exists. **Not required** if importing to the CDS being used by the HSC/VTCS system that is active on the system where the import job runs. Otherwise it is required, to specify the Standby CDS to be updated by import.

  If you specify the `INACTCDS` parameter, this CDS must be **different** from the CDS being used by HSC/VTCS (if active).

JCL Examples

JCL Example: IMPORT Validation Run, HSC is Active

Figure 57 shows example JCL to run the IMPORT utility. In this example, the MANIFEST parameter specifies importing all MVCs and VTVs listed on input manifest file REMOTE1. This example shows a “validation” run where:

- REPLACE(NONE) (the default) specifies that VTCS does not overwrite duplicate VTVs.
- IMMDRAIN(NO) (the default) specifies that VTCS does not drain all imported VTVs to VTSS space.
- NOUPDATE specifies that the CDS is not updated.
- HSC is active.

```
//IMPORT  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//REMOTE1      DD DSN=FEDB.VSMLMULT.REMOTE1,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 IMPORT MANIFEST(REMOTE1) NOUPDATE
```

Figure 57. ***IMPORT utility example: validation import run, HSC is active***

JCL Example: Import Production Run, HSC is Active

Figure 58 shows example JCL to run the IMPORT utility. In this example, the MANIFEST parameter specifies importing all MVCs and VTVs listed on input manifest file REMOTE1. This example shows an actual import run where:

- REPLACE(ALL) specifies that VTCS overwrites all duplicate VTVs.
- IMMDRAIN(YES) specifies that VTCS drains all imported VTVs to VTSS space.
- The NOUPDATE parameter is not specified so that the CDS is updated.
- HSC is active.

```
//IMPORT  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//REMOTE1      DD DSN=FEDB.VSMLMULT.REMOTE1,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 IMPORT MANIFEST(REMOTE1) REPLACE(ALL) IMMDRAIN(YES)
```

Figure 58. ***IMPORT utility example: actual import run, HSC is active***

JCL Example: IMPORT Using Multiple File Generations, HSC is Active

Figure 59 shows example JCL to run the `IMPORT` utility. In this example, the `MANIFEST` parameter specifies importing multiple generations of a file, HSC is active, and VTCS overwrites all duplicate VTVs via the `REPLACE(ALL)` parameter.

```
//IMPORT  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//ALLGEN       DD DISP=SHR,DSN=EXPORT(0)
//             DD DISP=SHR,DSN=EXPORT(-1)
//             DD DISP=SHR,DSN=EXPORT(-2)
//             DD DISP=SHR,DSN=EXPORT(-3)
//             DD DISP=SHR,DSN=EXPORT(-4)
//             DD DISP=SHR,DSN=EXPORT(-5)
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  IMPORT MANIFEST(ALLGEN) REPLACE(ALL)
```

Figure 59. `IMPORT` ***utility example: import multiple file generations, HSC is active***

JCL Example: IMPORT with No HSC Active on the System Where Import is to Run, No Hosts Using the CDS

Figure 60 shows example JCL to run the `IMPORT` utility with Manifest File `DMFST`. In this example:

- No HSC is active on the system where IMPORT is to run, so the `SLSCNTL`, `SLSCNTL2`, and `SLSSTBY` DD statements are specified.
- When HSC is active it uses `FEDB.PROD.DBASExxx`.
- No hosts are using the CDS.
- VTCS overwrites all duplicate VTVs via the `REPLACE(ALL)` parameter.

```
//IMPORT EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB  DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT DD SYSOUT=*
//SLSCNTL  DD DSN=FEDB.PROD.DBASEPRM,DISP=SHR
//SLSCNTL2 DD DSN=FEDB.PROD.DBASESEC,DISP=SHR
//SLSSTBY  DD DSN=FEDB.PROD.DBASETBY,DISP=SHR
//DMFST    DD DSN=FEDB.PROD.DMFST,DISP=SHR
//SLSIN DD *
   IMPORT MANIFEST(DMFST) REPLACE(ALL)
```

Figure 60. `IMPORT` ***utility example: IMPORT with No HSC Active on the System Where Import is to Run, No Hosts Using the CDS***

JCL Example: IMPORT to An Alternate CDS

Figure 61 shows example JCL to run the IMPORT utility with Manifest File ALTMFST. In this example:

- HSC is active on the system where Import runs, but is using a different CDS (FEDB.PROD.DBASExxx)
- The INACTCDS parameter specifies importing to an alternate CDS specified by the SLSCNTL, SLSCNTL2, and SLSSTBY DD statements.
- No hosts are using the CDS.
- VTCS overwrites all duplicate VTVs via the REPLACE(ALL) parameter.

```
//IMPORT EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB  DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT DD SYSOUT=*
//SLSCNTL  DD DSN=FEDB.VSMALT.DBASEPRM,DISP=SHR
//SLSCNTL2 DD DSN=FEDB.VSMALT.DBASESEC,DISP=SHR
//SLSSTBY  DD DSN=FEDB.VSMALT.DBASETBY,DISP=SHR
//ALTMFST  DD DSN=FEDB.VSMALT.ALTMFST,DISP=SHR
//SLSIN    DD *
IMPORT MANIFEST(ALTMFST) REPLACE(ALL) INACTCDS
```

Figure 61. IMPORT ***utility example: IMPORT to An Alternate CDS***

MERGMFST

MERGMFST merges multiple manifest files produced by EXPORT into a single file.

Syntax

```
►──MERGMFST──────MERGEIN(manifin)──────────MERGEOUT(manifout)──────►◄
```

Figure 62. MERGMFST ***utility syntax***

Parameters

MERGEIN
: specifies the DD statement for one or more input manifest files.

  manifin
  : the DD name.

MERGEOUT
: specifies the DD statement for the merged manifest file.

  manifout
  : the DD name.

Interfaces

SWSADMIN utility only.

Usage

Use MERGMFST to merge multiple manifest files into a single file, which can be used as input to the following:

- IMPORT
- MVCMAINT
- MVCRPT
- VTVRPT

Note: For VTCS 5.0 and above, running MERGMFST produces an MVC Detail Report.

JCL Requirements

`STEPLIB`
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

`manifin DD`
: DD statement for the input manifest file(s).

`manifout DD`
: DD statement for the merged manifest file.

`SLSPRINT`
: specifies the destination for the `MERGMFST` report.

`SLSIN`
: specifies the input to the `SWSADMIN` program (`MERGMFST` utility name and parameters).

JCL Examples

Figure 57 shows example JCL to run the `MERGMFST` utility to merge 5 manifest files. **Note that** because each file is explicitly specified, they can be specified in any order.

```
//MERGMFST  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//manifin      DD DISP=SHR,DSN=EXPORT1
//             DD DISP=SHR,DSN=EXPORT2
//             DD DISP=SHR,DSN=EXPORT3
//             DD DISP=SHR,DSN=EXPORT4
//             DD DISP=SHR,DSN=EXPORT5
//manifout     DD DSN=mergefile,DISP=(,CATLG),UNIT=SYSALLDA,
               DCB=(EXPORT5),SPACE=CYCL,(1,1))
//SLSIN        DD *
 MERGMFST MERGEIN(manifin) MERGEOUT(manifout)
```

Figure 63. `MERGMFST` ***utility example: merge 5 specified manifest files***

Figure 64 shows example JCL to run the `MERGMFST` utility to merge 5 generations of manifest files.

```
//MERGMFST  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//manifin      DD DISP=SHR,DSN=EXPORT(0)
//             DD DISP=SHR,DSN=EXPORT(-1)
//             DD DISP=SHR,DSN=EXPORT(-2)
//             DD DISP=SHR,DSN=EXPORT(-3)
//             DD DISP=SHR,DSN=EXPORT(-4)
//manifout     DD DSN=MERGFILE.MANIFEST(+1)
//SLSIN        DD *
 MERGMFST MERGEIN(manifin) MERGEOUT(manifout)
```

Figure 64. `MERGMFST` ***utility example: merge 5 generations of manifest files***

Figure 65 shows example JCL to run the `MERGMFST` utility to merge all generations of manifest files.

```
//MERGMFST  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//manifin      DD DISP=SHR,DSN=EXPORT
//manifout     DD DSN=mergefile,MANIFEST(+1)
//SLSIN        DD *
 MERGMFST MERGEIN(manifin) MERGEOUT(manifout)
```

Figure 65. `MERGMFST` ***utility example: merge all generations of manifest files***

Figure 66 shows example JCL to run the `MERGMFST` utility to merge the current manifest file into a cumulative manifest file.

```
//MERGMFST  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//manifin      DD DISP=SHR,DSN=EXPORT(0)
//             DD DISP=SHR,DSN=MERGED.MANIFEST(0)
//manifout     DD DSN=MERGED,MANIFEST(+1)
//SLSIN        DD *
 MERGMFST MERGEIN(manifin) MERGEOUT(manifout)
```

Figure 66. `MERGMFST` ***utility example: merge the current manifest file into a cumulative manifest file***

MIGRATE

`MIGRATE` migrates VTVs to MVCs.

Syntax - Format 1

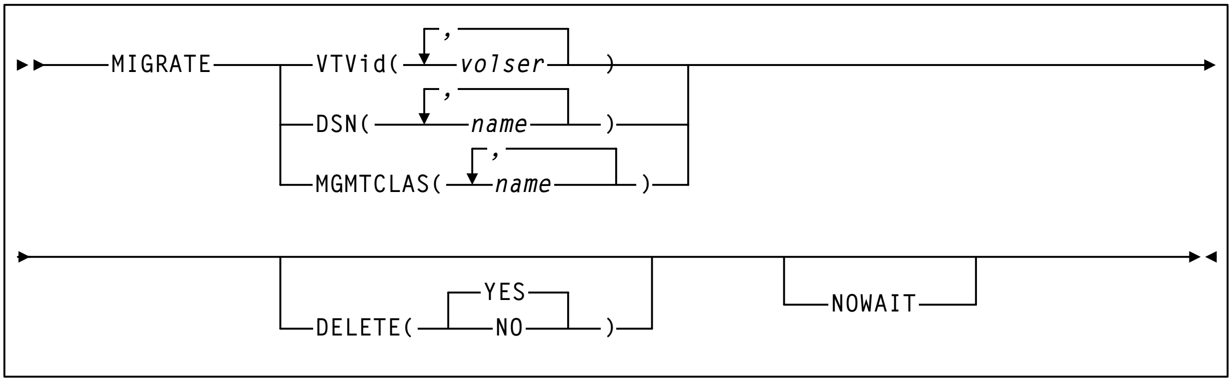


Figure 67. MIGRATE ***utility syntax - Format 1***

Parameters - Format 1

`VTVid`

specifies the VTVs that VSM migrates.

`volser`

the volsers of one or more VTVs. You can also specify one or more ranges.

`DSN`

specifies data sets used to select VTVs to migrate.

`name`

the data set name. Table 4 describes the valid wild cards for data set names. You cannot address a member of a GDG using a wildcard.

Note: Wildcards are only supported on MSP systems running DFSMS/MSP 1.4 or greater. At systems below this level the catalog search does not support wildcard.

Table 4. ***Data Set Name Wildcards***

Wildcard	Stands for...
*	A qualifier or one or more characters within a qualifier. An asterisk can precede or follow a set of characters.
**	zero or more qualifiers. A double asterisk cannot precede or follow any characters; it must be preceded or followed by either a period or a blank.
% or ?	Exactly one alphanumeric or national character.

Table 4. ***Data Set Name Wildcards***

Wildcard	Stands for...
%% or ??	One to eight percent signs or question marks can be specified in each qualifier.

`MGMTCLAS`

specifies one or more Management Classes that determine one or more VTVs to migrate.

`mgmt-class-name` | `mgmt-class-list`

the names of one or more Management Classes that you defined on the `MGMTclas` control statement; for more information, see “MGMTCLAS Control Statement” on page 204.

Note: The `VTVid`, `DSN`, and `MGMTCLAS` parameters are mutually exclusive.

`DELETE`

specifies whether VSM deletes VTVs from the VTSS after migrating the VTVs.

`NO`

do not delete VTVs from the VTSS after migrating the VTVs.

`YES`

delete VTVs from the VTSS after migrating the VTVs (the default).

`NOWAIT`

specifies that the utility does not wait for the operation to complete and returns after the request is submitted.

Syntax - Format 2

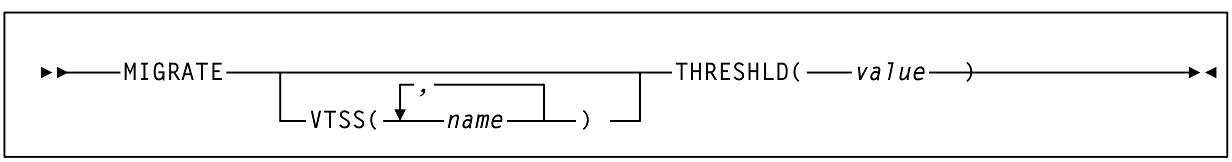


Figure 68. MIGRATE ***utility syntax - Format 2***

Parameters - Format 2

`VTSS`

specifies one or more VTSSs to migrate to the specified threshold.

`name`

the names of one or more VTSSs.

`THRESHLD`

specifies that VTCS runs the VTSS space management/VTV migration cycle until VTSS space reaches the specified threshold. Valid values are 0 to 95%.

`value`

the threshold to migrate to (percent of VTSS space).

Interfaces

SWSADMIN utility and VT command.

Usage

Use the MIGRATE utility to do demand migrations of VTVs to MVCs.

You can do two types of demand migrations:

- Migrate specified VTVs by specifying the VTVid, DSN, or MGMTCLAS parameter (and, optionally, the DELETE (NO) parameter to prevent VSM from deleting the VTVs after migration).
- Use the THRESHLD parameter to force the VTSS space management/VTV migration cycle to run to a specified threshold for specified VTSSs.

Note: The SET HIGHthld and SET HIGHthld parameters are no longer supported on the MIGRATE command and utility. To change the low and high AMT, use the SET MIGOPT command and utility. For more information, see “SET MIGOPT” on page 154.

Command Examples

To run MIGRATE to migrate VTVs determined by Management Class MCLAS1 and not delete them immediately from VTSS space., enter the following:

```
.VT MIGRATE MGMTCLAS(MCLAS1) DELETE(NO)
```

To run MIGRATE to migrate VTSS1 to a threshold of 10%, enter the following:

```
.VT MIGRATE VTSS(VTSS1) THRESHLD(10)
```

JCL Requirements

The following are the required and optional statements for the MIGRATE JCL:

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the migrate report.

SLSIN
: specifies the input to the SWSADMIN program (MIGRATE utility name and parameters).

JCL Examples

Figure 69 shows example JCL to run `MIGRATE` to migrate VTVs determined by Management Class `MCLAS1` and not delete them immediately from VTSS space.

```
//MIGRATE     EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  MIGRATE MGMTCLAS(MCLAS1) DELETE(NO)
```

Figure 69. ***Example JCL for the*** `MIGRATE` ***utility (migrate by Management Class)***

Figure 70 shows example JCL to run `MIGRATE` to migrate `VTSS1` to a threshold of 10%.

```
//MIGRATE     EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  MIGRATE VTSS(VTSS1) THRESHLD(10)
```

Figure 70. ***Example JCL for the*** `MIGRATE` ***utility (migrate to threshold)***

MVCDEF

The VT MVCDEF command loads the MVCPool statements from a specified definition data set.

Syntax

```
►►──MVCDEF──┬─DSN(dataset.name)─────┬──────┬─────────────────┬──►
            └─DATASET(dataset.name)─┘      └─VOLume(volser)─┘

►─┬────────────────┬───────────────────────────────────────────►◄
  └─UNIT(unitname)─┘
```

Figure 71. VT MVCDEF command ***syntax***

Parameters

DSN or DATASET

specifies the definition data set that contains the MVCPool statements to load.

dataset.name

the data set name. If the data set name includes a member name, enclose the data set name in quotes.

VOLume

specifies the DASD volume where the definition data set resides. This parameter is optional, unless the data set is not cataloged, or the data set resides on a volume other than the volume indicated by the catalog.

volser

the DASD volser.

UNIT

specifies the DASD device where the definition data set resides.

unitname

the DASD unit name. If the definition data set is not cataloged and this parameter is omitted, the unit name defaults to SYSALLDA.

Interfaces

VT command only.

Usage

Use the VT MVCDEF command to load MVCPool statements in a specified definition data set; for more information, see "MVCPOOL Control Statement" on page 216. The VT MVCDEF command is valid for base and full service levels of HSC. For more information about using the VT MVCDEF command and the MVCPool statement.

You can enter the VT MVCDEF command as an operator command or specify the VT MVCDEF command as a statement in the HSC PARMLIB. If you specify a VT MVCDef statement in the PARMLIB, HSC startup loads the specified MVCPool statements. After HSC startup, you can enter the VT MVCDEF command as a command to dynamically reload another set of MVCPool statements from a different definition data set. If you restart HSC, it reloads the MVCPool statements specified by the VT MVCDEF command statement in the PARMLIB.

Note that if you dynamically reload another set of MVCPool statements, any MVCs that were in the previous MVC pool but are not in the current MVC pool definition are removed from the MVC pool unless the MVC contains valid VTVs. These MVCs are removed from the pool once those VTVs become invalid.

If the MVCPool definition data set contains errors when the VT MVCDEF command is entered, HSC issues message SLS5627I, which identifies the MVCPool parameter in error, describes the problem, and gives the line number where the error occurred. If an error occurs, VSM does not load the specified MVCPool statements. Correct the MVCPool statements problem and reissue the VT MVCDEF command.

Example

To load the MVCPool statements in data set VSM.MVCPOOL, enter:

```
.VT MVCDEF DSN(VSM.MVCPOOL)
```

MVCDRAIN

MVCDRain recalls all current and scratched VTVs from an MVC and, optionally, “virtually” ejects the MVC (makes it unavailable for VSM use without physically ejecting it from the library). You can use the MVCDRain to override the CONFIG RECLAIM CONMVC setting.

Syntax

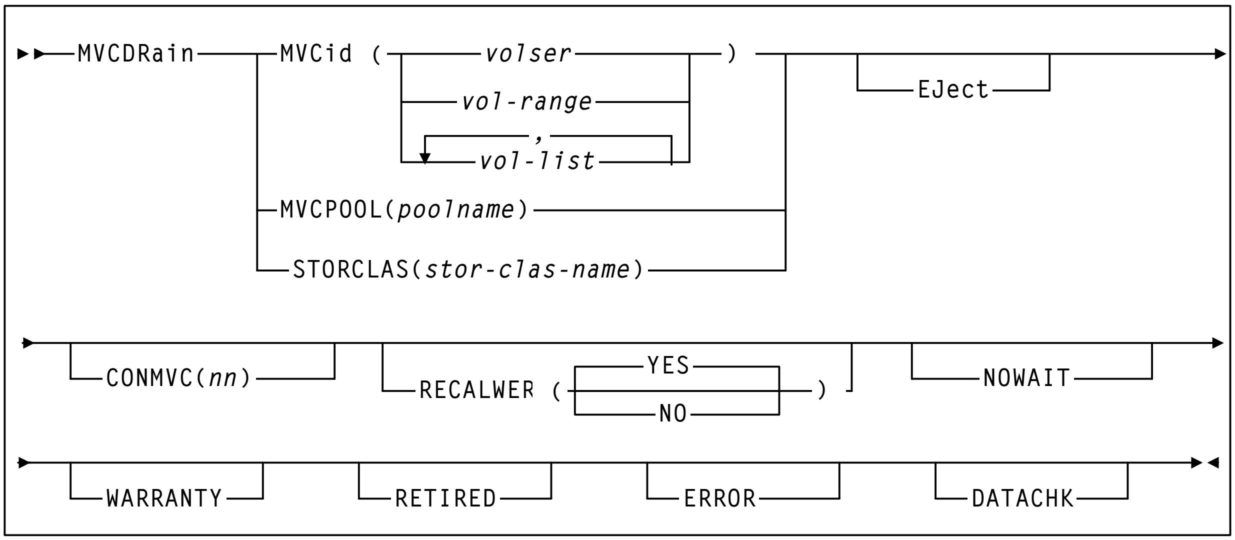


Figure 72. MVCDRain ***syntax***

Parameters

MVCid

drain one or more MVCs by volser.

volser, vol-range, or vol-list

the volsers of one or more MVCs up to a maximum of 50.

MVCPOOL

drain the MVCs in the specified Named MVC Pool.

poolname

the name of an MVC Pool that you defined on the MVCPool control statement; for more information, see “MVCPOOL Control Statement” on page 216.

STORCLAS

drain the MVCs in the specified Storage Class.

stor-class-name

the name of a Storage Class that you defined on the STORclas control statement; for more information, see “STORCLAS Control Statement” on page 219.

EJect

specifies that VTCS “virtually” ejects the MVC (the MVC will not be used for output).

`CONMVC(nn)`
: specifies the maximum number of MVCs that VTCS concurrently processes for both drain and reclaim.

  Valid values are 1 to 99. If not specified, the default is the `CONMVC` value specified on the `CONFIG RECLAIM` statement.

`RECALWER`
: specifies whether VTCS recalls VTVs with read data checks.

  `YES`
  : recall VTVs with read data checks (the default).

  `NO`
  : Do not recall VTVs with read data checks.

`NOWAIT`
: specifies that the utility does not wait for the operation to complete and returns after the request is submitted.

  returns after the request is submitted.

`WARRANTY`
: selects MVCs with expired warranties (denoted by a `W` in the `Status T` column on an MVC Report).

`RETIRED`
: selects MVCs that are retired (denoted by a `T` in the `Status T` column on an MVC Report).

`ERROR`
: selects MVCs that are in error (denoted by a `B` in the `Status B` column on an MVC Report).

`DATACHK`
: selects MVCs that have a data check (denoted by a `D` in the `Status D` column on an MVC Report).

Interfaces

`SWSADMIN` utility and `VT` command.

Usage

Use the MVCDRain to “drain” an MVC. The recall follows the same recall policy used for RECAll; for more information, see “RECALL” on page 135.

To select the MVCs to drain, you can specify one of the following parameters:

- MVCid to drain one or more MVCs by volser.
- MVCPOOL to drain the MVCs in a Named MVC Pool. For more information on Named MVC Pools.
- STORCLAS to drain the MVCs in a Storage Class. For more information on Storage Classes.

You typically drain an MVC for the following reasons:

- An MVC report or Display shows data check errors for the MVC. VSM will not migrate to the MVC and you should remove it from the MVC pool.
- An MVC report or Display shows errors other than data check errors for the MVC.
- A Storage Class or Named MVC Pool is no longer in use and you want to remove or reuse the associated MVCs.

You can use the MVCDRain to override the CONFIG RECLAIM CONMVC setting. You can run the MVCDRain from each host, which starts drain tasks on that host equal to the CONMVC value. These drain tasks can run concurrently with drain tasks initiated by other hosts.

Note: VTCS and HSC must be active to process an MVCDRain request.

Command Example

To run the MVCDRain to drain the MVCs in Storage Class STORCL1, virtually eject the MVCs, and return after the request is submitted, enter the following:

```
.VT MVCDRAIN STORCLAS(STORCL1) EJECT NOWAIT
```

JCL Requirements

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the MVCDRain report.

SLSIN
: specifies the input to the SWSADMIN program (MVCDRain utility name and parameters).

JCL Example

Figure 73 shows example JCL to run MVCDRain to drain the MVCs in Storage Class STORCL1, virtually eject the MVCs, and return after the request is submitted.

```
//MVCDRAIN  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  MVCDRAIN STORCLAS(STORCL1) EJECT NOWAIT
```

Figure 73. MVCDRain ***utility example: draining MVCs by Storage Class***

MVCMAINT

`MVCMAINT` sets MVC attributes.

Syntax

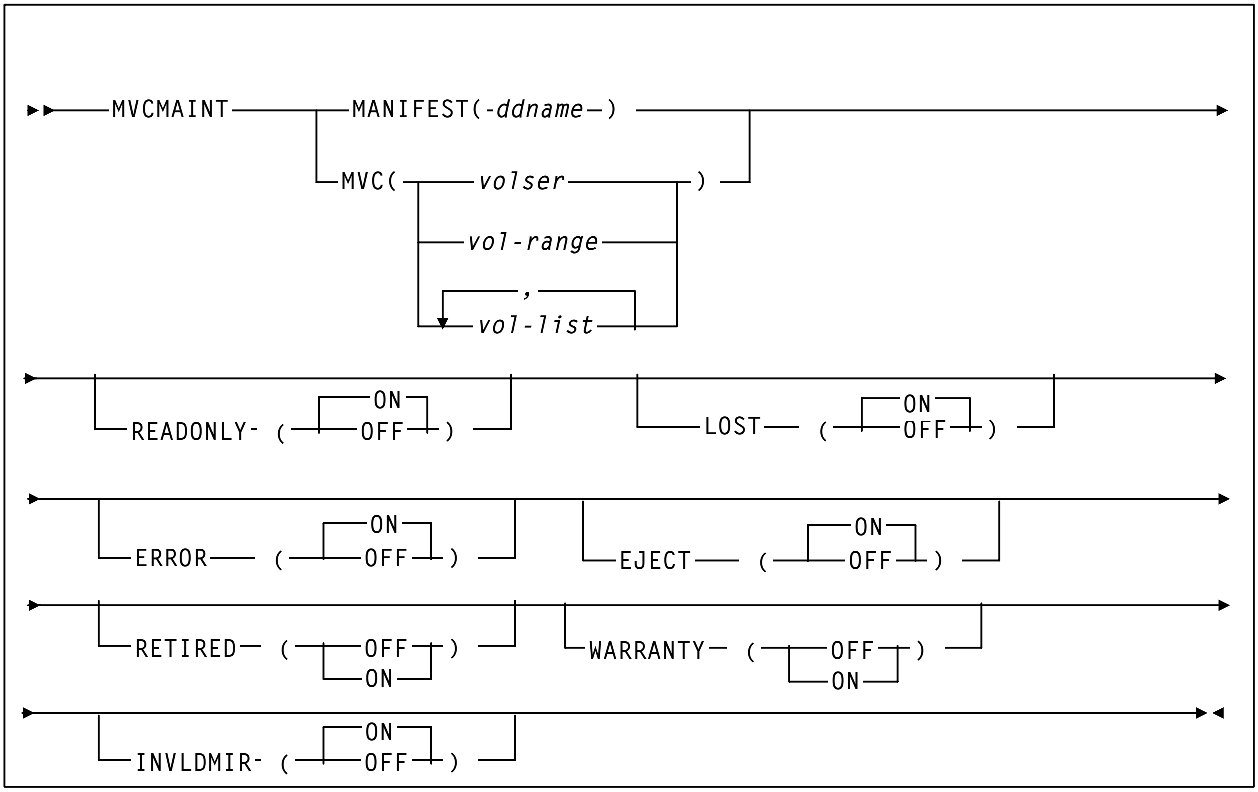


Figure 74. **`MVCMAINT` *syntax***

Parameters

`MANIFEST`

specifies the input ddname of the manifest file. **Note that** you can specify a merged manifest file or multiple manifest files. Figure 59 on page 93 shows example JCL to run the `MVCMAINT` utility where the `MANIFEST` parameter specifies importing multiple generations of a file.

`ddname`

ddname of the manifest file. The default is `MANIFEST`.

`MVC`

specifies the MVCs whose read/write attribute is changed.

`volser`, `vol-range` or `vol-list`

the volsers of one or more MVCs.

`READONLY`

sets the read/write status of the MVC.

`ON`

MVC is readonly (the default).

`OFF`

MVCs is writable.

`LOST`

sets the “lost” status of the MVC.

`ON`

MVC is lost (the default).

`OFF`

MVC is not lost.

`ERROR`

sets the error status of the MVC.

`ON`

MVC is in error (the default).

`OFF`

MVC is not in error.

`EJECT`

sets the “logical eject” status of the MVC.

`ON`

MVC is “logically ejected” (the default).

`OFF`

MVC is not “logically ejected”.

`RETIRED`

sets the “retired” status of the MVC. There is no default; you must specify a value.

`ON`

MVC is retired.

`OFF`

MVC is not retired but is still in “expired warranty” state (still selectable for output).

`WARRANTY`

sets the “expired warranty” status of the MVC. There is no default; you must specify a value.

`ON`

MVC’s warranty has expired.

`OFF`

MVC’s warranty has not expired.

Note: `RETIRED` and `WARRANTY` are mutually exclusive. That is, you cannot specify `RETIRED` and `WARRANTY` on the same `MVCMAINT` command.

`INVLDMIR`

sets the invalid MIR status of the MVC.

`ON`

MIR is invalid (the default).

`OFF`

MIR is not invalid.

Interfaces

`SWSADMIN` utility only.

Usage

Use the `MVCMAINT` to set the following MVC attributes:

- Read/write status (including exported and imported MVC)s.
- “Lost” status. If a VTCS initiated mount of an MVC on an RTD fails to complete (as opposed to completes with an error), VTCS updates the MVC record in the CDS to indicate that the MVC is “lost”. An MVC in “lost” status is depreferenced where ever possible. Duplexed VTVs that reside on this MVC are recalled from their alternate MVC. VTCS does not attempt to use “lost” MVCs for migration unless there are no other valid MVCs. When an MVC in “lost” status is successfully mounted, the “lost” status is removed from the MVC record.

  You can use the `LOST` parameter to explicitly set the “lost” status of an MVC in to recover from known conditions. For example, if an ACS is in manual mode, you can set `LOST(ON)` for the MVCs in that ACS to depreference their use.

- Error status. The following are typical error conditions and corrective actions. After correcting the error, you can use MVCMAINT to set ERROR OFF for the MVC.
  - VTCS does not recognize the volume mounted on the RTD as an MVC. This can be caused by some MSP job overwriting the MVC. Determine what happened to the MVC. If it no longer contains valid VTV data, reinitialize the volume and return it to the MVC pool.
  - The MVC is not writeable, which can be caused by the thumbwheel being set to readonly, or by the security package not allowing VTCS to write to the volume. Reset the thumbwheel, or change the rules in the security package to allow the MVC to be written to.
  - A bad block ID has been detected. Audit the MVC to try to correct the condition. For more information, see “AUDIT” on page 5.
- “Logical eject” status.
- For 9840/T9940 media:
  - VTCS also detects media warranty expiration and sets the WARRANTY status to ON. Alternatively, you can use SMF or LOGREG data to detect MVCs approaching end-of-life and use the MVCMAINT to manually set WARRANTY OFF. Knowing that the warranty has expired lets you plan for media replacement before media end-of-life occurs.

    You can also use the MVCMAINT to set WARRANTY OFF for MVCs erroneously marked as warranty expired.
  - VTCS also automatically detects media end-of-life and sets the RETIRED status to ON. Alternatively, you can use SMF or LOGREG data to detect MVCs approaching end-of-life and use the MVCMAINT to manually set RETIRED ON.

    You can also use the MVCMAINT to set RETIRED OFF for MVCs erroneously marked as retired.
  - VTCS automatically detects an invalid Media Information Region (MIR) and sets the INVLDMIR status to ON. You can recover the MIR by using either the utility available through the operator panel for the transport or by using the utility available through MPST. After you recreate the MIR, you can use the MVCMAINT to set INVLDMIR OFF for the MVC.
  - The MVC report, MVC Pool Report, and Display MVC command report “retired” and invalid MIR status; for more information, see:
    - “MVC Reports” on page 125
    - “Named MVC Pool Report” on page 116
    - “Display MVC Output” on page 66

Note: Running MVCMAINT also produces an MVC report of the volumes affected by the MVCMAINT job.

JCL Requirements

`STEPLIB`
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

manifest file `DD`
: DD statement for the manifest file.

`SLSPRINT`
: specifies the destination for the `MVCMAINT` report.

`SLSIN`
: specifies the input to the `SWSADMIN` program (`MVCMAINT` utility name and parameters).

JCL Examples

Figure 75 shows example JCL to run `MVCMAINT` to make writable all MVCs listed on input manifest file `REMOTE1`.

```
//MVCMAINT  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//REMOTE1      DD DSN=FEDB.VSMLMULT.REMOTE1,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 MVCMAINT MANIFEST(REMOTE1) READONLY(OFF)
```

Figure 75. `MVCMAINT` ***utility example: making MVCs writable***

Figure 76 shows example JCL to run `MVCMAINT` to set on the “lost” status on for MVCs MVC001-MVC100.

```
//MVCMAINT  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//REMOTE1      DD DSN=FEDB.VSMLMULT.REMOTE1,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 MVCMAINT MVC(MVC001-MVC100) LOST(ON)
```

Figure 76. `MVCMAINT` ***utility example: setting MVCs “lost” status on***

MVCMAINT Reports

Figure 77 shows an example of an MVCMAINT report for the following command:

```
MVCMAINT MVC(022577-022579) READONLY=OFF
```

```
SWSADMIN (6.1.0)                 STORAGETEK VTCS SYTEM UTILITY                                       PAGE 0002
TIME 09:26:54                    MVC MAINTENANCE                                                     DATE 2003-05-14


MVCMAINT SUMMARY REPORT
MVC      RC
022577   00
022578   08
022578   08


MVCMAINT EXCEPTION REPORT
*SLS6737I MVC 022578 ALREADY HAS READONLY(OFF); REQUEST IGNORED
 *SLS6737I MVC 022579 ALREADY HAS READONLY(OFF); REQUEST IGNORED
SLS1315I SWS500.V5.CDS WAS SELECTED AS THE PRIMARY CONTROL DATA SET


SWSADMIN (6.1.0)                 STORAGETEK VTCS SYTEM UTILITY                                       PAGE 0002
TIME 09:26:54                    VTCS MVC SUMMARY REPORT                                             DATE 2003-05-14
MVC        NUMBER  %USED  %AVAIL  %FRAG    MEDIA     TIMES  STATUS           <-----LAST MOUNTED---->     ACS  OWNER/
VOLSER    OF VTVS                       SIZE (MB)  MOUNTED  I B L D R U T M  DATE        TIME      VTSS    ID   CONSOLIDATE TIME


022577          0   0.00   99.96   0.04       400      142  I - - - - C - -  2003MAY14   06:23:23          00   2003MAY14
                                                                                                             06:09:23
022578          0   0.00   99.96   0.04       400      197  I - - - - U - -  2003MAY14   06:23:23  VTSS16  00   VTSS16
022579          0   0.00   99.96   0.04       400      142  I - - - - U - -  2003MAY14   16:23:23          00   2003MAY14
                                                                                                             16:09:23
                3  INITIALIZED MVCS PROCESSED
                0  NON-INITIALIZED MVCS PROCESSED
                0  NON-LIBRARY MVCS PROCESSED
```

Figure 77. ***Example* MVCMAINT *report***

As shown in Figure 77, the MVCMAINT report shows:

- Status of MVCs processed - volser and return code (0 - all updates completed, 4 - some updates completed, 8 - no updates completed).
- An exception report of the reason for all uncompleted updates.
- An MVC summary report; for more information, see "MVC Summary Report" on page 125.

MVCPLRPT

The MVCPLRPT reports the status of a Named MVC Pool or MVC Storage Class.

Syntax

```
►►──MVCPLRPT──┬──────────────────────────────┬──────────────►◄
              ├──MVCPOOL(poolname)───────────┤
              └──STORCLAS(stor-clas-name)────┘
```

Figure 78. MVCPLRPT ***syntax***

Parameters

MVCPOOL

report on the MVCs in the specified Named MVC Pool.

poolname

the name of an Named MVC Pool that you defined on the MVCPool control statement; for more information, see “MVCPOOL Control Statement” on page 216.

To report on all Named MVC Pools (including DEFAULTPOOL), specify ALL or omit the MVCPOOL parameter.

STORCLAS

report on the MVCs in the specified Storage Class. The report is produced as if the associated MVC pool name had been specified.

stor-class-name

the name of a Storage Class that you defined on the STORclas control statement; for more information, see “STORCLAS Control Statement” on page 219.

Interfaces

SWSADMIN utility only.

Usage

Use the MVCPLRPT to report the status of a Named MVC Pool or MVC Storage Class. Figure 79 on page 115 shows example JCL to run MVCPLRPT to produce a report by Named MVC Pool and Figure 80 on page 116 shows the report format.

JCL Requirements

The following are the required and optional statements for the MVCPLRPT JCL:

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the MVC report.

SLSIN
: specifies the input to the SWSADMIN program (MVCPLRPT utility name and parameters).

JCL Example

Figure 79 shows example JCL to run MVCPLRPT to produce a report for Named MVC Pool CUST1POOL.

```
//MVCPLR     EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SYSOUT       DD SYSOUT=*
//SLSIN        DD *
  MVCPLRPT MVCPOOL(CUST1POOL)
```

Figure 79. Example JCL for the* MVCPLRPT *utility (report by Named MVC Pool)

Named MVC Pool Report

Figure 80 and Figure 81 on page 117 show an example of a report for Named MVC Pool `CUST1POOL`.

```
SWSADMIN (6.1.0)                            STORAGETEK VTCS SYTEM UTILITY                                       PAGE 0002

TIME 09:26:54                               VTCS MVC SUMMARY REPORT - MVCPOOL=CUST1POOL                         DATE 2003-04-13

MVC        NUMBER  %USED   %AVAIL  %FRAG    MEDIA    TIMES  STATUS            <-----LAST MOUNTED---->           ACS  OWNER/

VOLSER     OF VTVS                         SIZE (MB) MOUNTED I B L D R U T M    DATE        TIME       VTSS     ID   CONSOLIDATE TIME



EVS99          200  10.80    84.57   4.63      2000     310  I - - - - U - M    2003MAR15   03:20:23   VTSS8    00   S1

EVS100           0   0.00   100.00   0.00   UNKNOWN     206  - - L - - U - -    2003MAR10   05:24:04   VTSS8    --

EVS101        1009  99.00     0.00   1.00       400     306  I - - - - U - -    2003MAR15   03:20:23   VTSS8    00   S1

EVS102           5   8.25    91.75   0.00       400       6  I - - - - U - -    2003MAR15   04:23:04   VTSS8    00   S3

EVS103      EXPVTV   0.12    99.88   0.00       400     194  I - - - - J - -    2003MAR15   03:20:28   VTSS10   00   VTSS10

EVS104           0   0.00   100.00   0.00       400       5  I - - - R C - -    2003MAR18   03:49:14   VTSS8    00   2003APR12

                                                                                                                     03:49:14

EVS105         200  10.80    84.57   4.63    102040     254  I - - - R U T -    2003MAR18   04110:09   VTSS8    00

EVS106           0   0.00   100.00   0.00       400     202  I - - - - C - -    2003MAR18   03:49:20   VTSS8    00

EVS107           0   0.00   100.00   0.00       400     171  I - - - R E - -    2003MAR18   04:13:00   VTSS8    00

SUMMARY FOR MVCPOOL=CUST1POOL


           ACS     MEDIA                FREE-MVCS                                  RECLAIM-MVCS              USED-MVCS

                                  VOLS          GB                         VOLS            GB               VOLS   GB



            00     ECART           120          96                            2           0.5                 90   45

            00     STK1R            30         600                            1           3.5                 25  350


            00     TOTAL           150         696                            3           4.0                115  395
```

Figure 80. ***Example* MVCPLRPT *report (Part 1)***

```
SUMMARY OF MVCS BY USAGE:


137 TOTAL MVCS PROCESSED

135 INITIALIZED MVCS PROCESSED

  2 UN-INITIALIZED MVCS PROCESSE

 41 FREE MVCS AVAILABLE

  0 MVCS WITH STATUS AUDIT

  6 MVCS WITH STATUS DRAIN

  4 MVCS WITH STATUS EXPORT

  0 MVCS MARKED EJECTED

 60 MVCS MARKED FULL

  0 MVCS WITH MAXIMUM VTVS

 82 MVCS MARKED READ-ONLY

  3 MVCS WITH STATUS BROKEN

  7 MVCS WITH STATUS LOST

  0 MVCS MARKED RETIRED

  0 MVCS WITH EXPIRED WARRANTY

  0 MVCS HAVE INVALID MIRS

  1 MVCS HAVE DATACHECKS

  5 MVCS WITH STATUS CONSOLIDATE
```

Figure 81. ***Example MVCPLRPT report (Part 2)***

MVCPLRPT Fields

The following list describes the Named MVC Pool report fields. The Summary fields are either for a Storage Class or a Named MVC Pool, depending on which was specified on the report JCL. If a Storage Class specifies a Named MVC Pool, the report gives information for that subpool.

MVC Volser
the MVC volser.

Number of VTVS
the number of current VTVs on the MVC. If the MVC has been used for VTV export, this field reports **EXPVTV**.

%Used
the percentage of the MVC used by current VTVs.

%Avail
the percentage of the MVC that is physically available for use.

%Frag

the percentage of the MVC that contains non-current VTVs. This space is not usable until it is reclaimed or the MVC is drained.

Media Size (MB)

the size of the MVC (MB). This will only be determined after VTCS has used an MVC. “UNKNOWN” appears in this field until VTCS migrates a VTV to the MVC.

Times Mounted

the number of times that the MVC has been mounted for writing or reading since it was added to the MVC inventory.

STATUS

one or more of the following statuses:

I

The MVC has been initialized.

B

The MVC has an error that should be investigated. The error may not make the MVC unusable, but VTCS will not select the MVC for migration for 12 hours after it is marked “B”. After the 12 hour period, the MVC will be least preferred for subsequent migrations, and recalls from the MVC may cause VTCS to drain it. This error condition may be accompanied by messages SLS6686, SLS6687, SLS6688, SLS6690, and/or SLS6693.

Any of the following conditions can cause this MVC error:

- MVC corrupted by another job (other than VTCS/VTSS).
- Attempt to use a read-only MVC for migration.
- A DDR swap failure.
- An RTD failure.

L

The MVC was not mounted in response to the last mount request. The MVC can still be used for migration, but will not select the MVC for migration for 12 hours after it is marked “L”. After the 12 hour period, the MVC will be least preferred. This condition will clear itself the next time that the MVC is mounted.

D

A data check was reported for this MVC. VSM will not use this MVC again for migration.

R

the MVC has been marked read-only.

U

one of the following usage statuses:

U

the MVC is available for output (migration, reclamation, export, or consolidation).

-

the MVC is not available for output (migration, reclamation, export, or consolidation).

A

The MVC is either being audited or the audit failed. If the audit failed, VTCS will not use the MVC for migration. To clear this condition, rerun the `AUDIT` against this MVC.

C

The MVC is a consolidation MVC.

E

The MVC is an export MVC.

F

There is no space available on the MVC.

J

Either you issued a `MVCDRain EJect` for the MVC or the MVC was ejected for update by a `RACROUTE` call. The MVC will not be used again for migration or recall. To clear this condition, use `MVCDRain` against MVC without the `EJect` option.

N

Either:

- The MVC is being drained because of:
  - An automatic drain or demand reclaim.
  - An explicit `MVCDRain` command.

OR

- The previous DRAIN request failed, in which case VTCS will not use the MVC for migration. To clear this condition, enter `MVCDRain` against MVC without the `EJect` option.

X

The MVC has reached the maximum VTVs per MVC.

T

One of the following statuses:

T

The MVC is retired.

W

The MVC's warranty has expired.

M

The MVC has an invalid MIR.

Last Mounted

the date and time that the MVC was last mounted and the VTSS where the MVC was last used.

ACS ID

the ACS where the MVC resides.

Owner/Consolidate Time

If the MVC is empty, this field is null. If the MVC is a consolidation MVC, this field displays the time of the consolidation. If the MVC is a migration MVC and contains current VTVs, this field displays the MVC’s Storage Class. If no Storage Class was explicitly assigned via the `MGMTclas` statement, the default Storage Class is the name of the last VTSS that wrote to the MVC for reclamation or migration.

If VTCS receives a request to migrate a VTV that is assigned to an invalid Management Class, VTCS will dynamically create the !ERROR Storage Class and migrate the VTVs defined by the invalid Management Class to the !ERROR Storage Class. Use this Storage Class to identify and correct invalid Management Classes, drain the affected MVCs, and resubmit the request.

Summary for Storage Class or Named MVC Pool

This section shows number of MVCs (**Vols**) and total storage (**Gb**) by ACS and media type for the following categories:

Free-MVCs

MVCs that have 100% usable space and do not contain any migrated VTVs. The storage shown is the total free space based on media type capacity.

Reclaim-MVCs

MVCs eligible for space reclamation. The storage shown is the total wasted space including those MVCs not yet eligible for space reclaim.

Used-MVCS

Initialized MVCs that are partially or completely full.

Total MVCs

Total MVCs for the Storage Class or Named MVC Pool with subtotals for initialized, uninitialized, and free MVCs.

Summary of MVCs by Usage

This section shows number of MVCs by the task that last used the MVC.

MVCRPT

The `MVCRPT` reports the status of your VSM system's MVCs.

Syntax

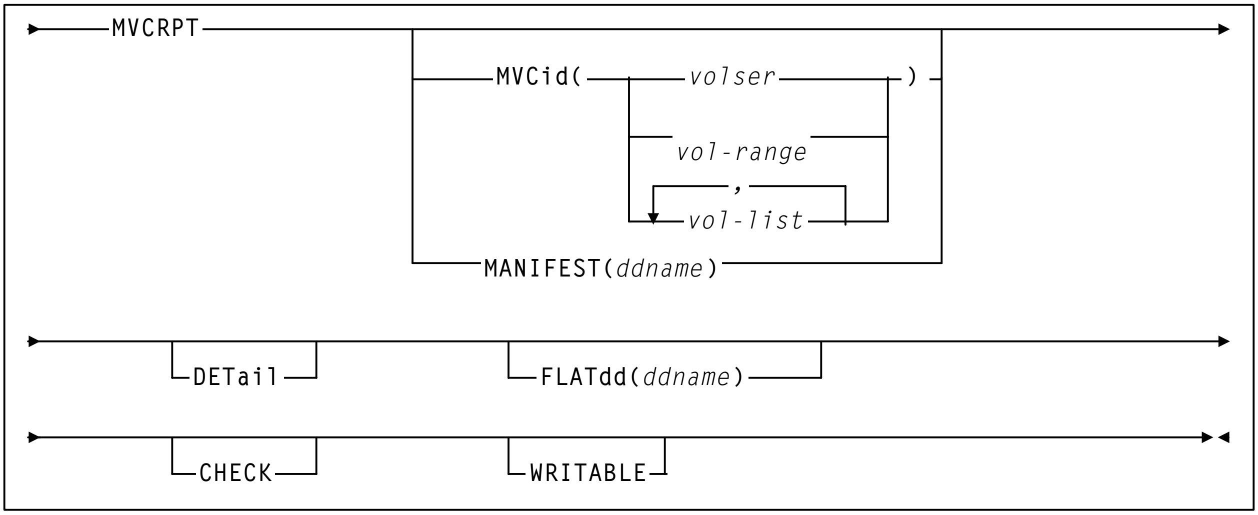


Figure 82. `MVCRPT` ***syntax***

Parameters

`MVCid`

specifies the MVCs for the report. If you do not specify the MVCs, the report includes all MVCs in your VSM system.

`volser`, `vol-range` or `vol-list`

the volsers of one or more MVCs.

`MANIFEST`

specifies the input ddname of the manifest file used to generate the report.

`ddname`

ddname of the manifest file. **Note that** you can specify a merged manifest file or multiple manifest files. Figure 59 on page 93 shows example JCL to run the `MVCMAINT` utility where the `MANIFEST` parameter specifies importing multiple generations of a file.

`DETail`

produce a detailed MVC report; see Figure 88 on page 130 for an example. If you do not specify this option, the default is to produce a summary MVC report; see Figure 87 on page 125 for an example.

`FLATdd`

specifies the destination of the optional flat file output.

`ddname`

name of the DD in the JCL that describes the output data set if a flat file is required.

`CHECK`

`MVCRPT` does limited MVC integrity checking, during which the CDS is reserved. Use **only** if directed by StorageTek Software Support.

`WRITABLE`

`MVCRPT` ignores readonly MVCs on MVC detailed reports and when you specify the `CHECK` parameter.

Interfaces

`SWSADMIN` utility only.

Usage

Use the `MVCRPT` to report the status of your VSM system's MVCs. For more information, see "MVC Reports" on page 125. Figure 83 shows example JCL to run `MVCRPT` to produce a report and Figure 87 on page 125 and Figure 88 on page 130 show the report formats. MVC reports only list MVCs that have been used, not MVCs that are defined but have not been not used, but non-initialized MVCs will be included in the report totals. As shown in Figure 84 on page 123, you can specify the `MANIFEST` parameter to generate a report from an export manifest file instead of from the HSC CDS.

To produce reports for VSM, ExPR requires a flat file format of the MVC report as input. Figure 85 on page 123 shows example JCL to produce flat file format of the report for ExPR. Table 5 on page 132 shows the flat file record format.

Also note that you can produce MVC reports in structured XML format. You can then process the XML output in a programming language of your choice. For information on the VTCS sample REXX execs, see "Sample REXX Execs" on page 313.

JCL Requirements

The following are the required and optional statements for the `MVCRPT` JCL:

`STEPLIB`

specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

`SLSPRINT`

specifies the destination for the MVC report.

`SLSIN`

specifies the input to the `SWSADMIN` program (`MVCRPT` utility name and parameters).

`SYSOUT`

specifies the output destination for SORT messages. This is only required for `DETAIL` MVC reports.

`SLSXML`

specifies the output destination for XML output. Allocate this file as RECFM=VB, LRECL=255.

JCL Examples

Figure 83 shows example JCL to run `MVCRPT` to produce a detailed report of all MVCs in your VSM system.

```
//MVCR      EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//MVCOUT       DD DSN=FEDB.FLAT,UNIT=SYSALLDA,DISP=OLD
//SLSPRINT     DD SYSOUT=*
//SYSOUT       DD SYSOUT=*
//SLSIN        DD *
  MVCRPT DET
```

Figure 83. Example JCL for the `MVCRPT` ***utility (detailed report)***

Figure 84 shows example JCL to run `MVCRPT` to produce a detailed report using manifest file `REMOTE1` as input.

```
//MVCR      EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//REMOTE1      DD DSN=FEDB.VSMLMULT.REMOTE1,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SYSOUT       DD SYSOUT=*
//SLSIN        DD *
  MVCRPT MANIFEST(REMOTE1) DET
```

Figure 84. Example JCL for the `MVCRPT` ***utility (detailed report, manifest file input)***

Figure 85 shows example JCL to run `MVCRPT` to produce a detailed report of all MVCs in flat file format for input to ExPR.

```
//MVCR      EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//MVCOUT       DD DSN=FEDB.FLAT,UNIT=SYSALLDA,DISP=OLD
//SLSPRINT     DD SYSOUT=*
//SYSOUT       DD SYSOUT=*
//SLSIN        DD *
  MVCRPT DET FLATDD(MVCOUT)
```

Figure 85. Example JCL for the `MVCRPT` ***utility (detailed report, flat file output for ExPR)***

Figure 86 shows example JCL to run `MVCRPT` to produce a detailed report of all MVCs in structured XML format.

```
//MVCREPT   JOB  x,CHRIS,CLASS=A,MSGCLASS=E
//*
//*
//*
//MVCRPT     EXEC PGM=SWSADMIN
//STEPLIB     DD DSN=ncs610.LINKLIB,DISP=SHR
//SLSCNTL     DD DSN=hlq.V61.CLUSTER.CDS,DISP=SHR
//SLSPRINT   DD SYSOUT=*
//SYSOUT     DD SYSOUT=*
//SLSXML      DD DISP=(,CATLG),DSN=hlq.MVCREPT.XML,
//            DCB=(RECFM=VB,LRECL=255,BLKSIZE=4096),
//            UNIT=SYSDA,SPACE=(CYL,(x,y))
//SLSIN      DD *
 MVCRPT
/*                                  //*

//TSO         EXEC PGM=IKJEFT01,DYNAMNBR=128
//SYSPROC     DD DSN=ncs610.SLSSAMP,DISP=SHR
//SYSTSPRT   DD SYSOUT=*
//SLSPRINT  DD SYSOUT=*

//INFILE      DD  DISP=SHR,DSN=hlq.MVCREPT.XML
//OUTFILE     DD  DISP=(,CATLG),DSN=hlq.MVCREPT.CSV,
//            DCB=(RECFM=VB,LRECL=1024,BLKSIZE=4096),
//          UNIT=SYSDA,SPACE=(CYL,(x,y))
//SYSTSIN     DD *
 %MVCXML
/*
//
```

Figure 86. ***Example JCL for the `MVCRPT` utility (structured XML format)***

Note: XML output can consume **considerable** space. You may want to consider routing your XML output to a VTV so that the output is compressed.

MVC Reports

The following sections describe the MVC summary and detailed reports that the `MVCRPT` produces.

MVC Summary Report

Figure 87 shows an example of an MVC summary report.

```
SWSADMIN (6.1.0)                    STORAGETEK VTCS SYTEM UTILITY                               PAGE 0002
TIME 09:26:54                       VTCS MVC SUMMARY REPORT                                     DATE 2003-04-13
MVC       NUMBER  %USED  %AVAIL  %FRAG    MEDIA    TIMES  STATUS            <-----LAST MOUNTED---->      ACS  OWNER/
VOLSER    OF VTVS                       SIZE (MB) MOUNTED I B L D R U T M   DATE       TIME      VTSS    ID   CONSOLIDATE TIME

EVS99        200  10.80   84.57   4.63     2000      310  I - - - - U - M   2003MAR15  03:20:23  VTSS8   00   S1
EVS100         0   0.00  100.00   0.00  UNKNOWN      206  - - L - - U - -   2003MAR10  05:24:04  VTSS8   --
EVS101      1009  99.00    0.00   1.00      400      306  I - - - - U - -   2003MAR15  03:20:23  VTSS8   00   S1
EVS102         5   8.25   91.75   0.00      400        6  I - - - - U - -   2003MAR15  04:23:04  VTSS8   00   S3
EVS103    EXPVTV   0.12   99.88   0.00      400      194  I - - - - J - -   2003MAR15  03:20:28  VTSS10  00   VTSS10
EVS104         0   0.00  100.00   0.00      400        5  I - - - R C - -   2003MAR18  03:49:14  VTSS8   00   2003APR12
                                                                                                              03:49:14
EVS104       200  10.80   84.57   4.63   102040      254  I - - - R U T -   2003MAR18  04110:09  VTSS8   00
EVS105       300  15.80   54.57   4.63   102040      154  I - - - R U W -   2003MAR18  04110:09  VTSS8   00
EVS106         0   0.00  100.00   0.00      400      202  I - - - - C - -   2003MAR18  03:49:20  VTSS8   00
EVS107         0   0.00  100.00   0.00      400      171  I - - - R E - -   2003MAR18  04:13:00  VTSS8   00

                   137    Initialized MVCs processed
                     8    Non-Initialized MVCs processed
```

Figure 87. ***Example* MVC *summary report***

The following list describes the MVC summary report fields.

MVC Volser
: the MVC volser.

Number of VTVS
: the number of current VTVs on the MVC. If the MVC has been used for VTV export, this field reports **EXPVTV**.

%Used
: the percentage of the MVC used by current VTVs.

%Avail
: the percentage of the MVC that is physically available for use.

%Frag
: the percentage of the MVC that contains non-current VTVs. This space is not usable until it is reclaimed or the MVC is drained.

Media Size (MB)

the size of the MVC (MB). This will only be determined after VTCS has used an MVC. “UNKNOWN” appears in this field until VTCS migrates a VTV to the MVC.

Times Mounted

the number of times that the MVC has been mounted for writing or reading since it was added to the MVC inventory.

STATUS

one or more of the following statuses:

I

The MVC has been initialized.

B

This is a generic error that indicates the MVC, drive, or combination of the two has a problem. VTCS attempts to de-preference MVCs with this state. To clear this state:

- If the MVC caused the problem, use a `DRAIN(EJECT)` command to remove the MVC from service.
- If the RTD caused the problem, use the `MVCMAINT` utility to reset the MVC state.

Note also that one or more of the following messages is issued for `BROKEN` status: `SLS6686`, `SLS6687`, `SLS6688`, `SLS6690`. For detailed recovery procedures for these messages, see *VTCS Messages and Codes*.

L

VTCS attempted to mount an MVC and the mount did not complete within a 15-minute time-out period. VTCS is attempting to recover from a situation that may be caused by hardware problems, HSC problems, or by the MVC being removed from the ACS. VTCS attempts to de-preference MVCs with this state.

If VTCS does perform a subsequent successful mount of an MVC with `LOST(ON)` state, VTCS sets the state to `LOST(OFF)`.

Determine the cause of the error and fix it. You can also use the VTCS `MVCMAINT` utility to set `LOST(OFF)` for the following events:

- `LOST(ON)` was set due to LSM failures or drive errors that have been resolved
- `LOST(ON)` was set because the MVC was outside the ACS and has been reentered.

D

A data check condition has been reported against this MVC. VTCS attempts to de-preference MVCs with this state. To clear this state:

- If all VTVs on the MVC are duplexed, use `MVCDRain` on the MVC without the `Eject` option. This recovers all VTVs and removes the MVC from service.
- If all VTVs on the MVC are not duplexed, VTCS `AUDIT` the MVC. The audit will probably fail. After the audit, do an `MVCDRAIN` (no eject). This recalls the VTVs before the data-check area in ascending block-id order and the VTVs after the data-check area in a descending block-id order. Processing the VTVs in this sequence ensures that VTCS recovers as many VTVs as possible from the media. You then need to recreate the data for any VTVs still on the MVC.

R

The MVC has been marked read-only because of one of the following conditions:

- The MVC being the target of an export or consolidation process. The read-only state protects the MVC from further updates.
- The MVC media is set to file protect. Correct the error and use the `MVCMAINT` utility to set `READONLY(OFF)`.
- The MVC does not having the appropriate SAF rules set to enable VTCS to update the MVC. Correct the error (for more information, see “Defining a Security System User ID for HSC, SMC, and VTCS” in Chapter 2 of *VTCS Installation and Configuration Guide*) and use the `MVCMAINT` utility to set `READONLY(OFF)`.

U

one of the following usage statuses:

U

the MVC is available for output (migration, reclamation, export, or consolidation).

-

the MVC is not available for output (migration, reclamation, export, or consolidation).

A

The MVC is either being audited or the audit failed. If the audit failed, VTCS will not use the MVC for migration. To clear this condition, rerun the `AUDIT` against this MVC.

C

The MVC is a consolidation MVC.

E

The MVC is an export MVC.

F

There is no space available on the MVC.

J

Either you issued `MVCDRain EJect` for the MVC or the MVC was ejected for update by a `RACROUTE` call. The MVC will not be used again for migration or recall. To clear this condition, use `MVCDRain` against MVC without the `EJect` option.

N

Either:

- The MVC is being drained because of:
  - An automatic drain or demand reclaim.
  - An explicit `MVCDRain` command.

OR

- The previous DRAIN request failed, in which case VTCS will not use the MVC for migration. To clear this condition, enter `MVCDRain` against MVC without the `EJect` option.

X

The MVC has reached the maximum VTVs per MVC.

T

One of the following statuses:

T

The MVC is retired. VTCS will recall from, but not migrate to, the MVC. Replace the MVC as soon as possible.

W

The MVC's warranty has expired. VTCS continues to use the MVC. You should start making plans to replace the MVC when it reaches Retired state.

M

VTCS has received status from an RTD to indicate the MIR (media information record) for a 9x40 media is invalid. An invalid MIR does not prevent access to data but may cause significant performance problems while accessing records on the tape. The MVC is not capable of high-speed searches on areas of the tape that do not have a valid MIR entry.

VTCS attempts to de-preference MVCs with this condition. For recalls, if the VTV resides on multiple MVCs, VTCS selects MVCs with valid MIRs ahead of MVCs with invalid MIRs. VTCS avoids using MVCs with invalid MIRs for migration, unless the migration is at the beginning of the tape. Migrating from the beginning of tape will correct the MIR.

VTCS detects the invalid MIR condition at either mount time or dismount time. If detected at mount time and the operation can be completed with another MVC, VTCS dismounts the first MVC and selects the alternate MVC. **Note that** VTCS has only a limited ability to switch to an alternate MVC. That is, it is mainly used for migrate and virtual mount.

For MVCs with invalid MIRs, determine the cause of the error, which may be caused by media or drive problems, and fix the error.

To recover an MVC with an invalid MIR, you simply need to read the MVC to the end of the tape, which can be done via a VTCS audit. If the media is the problem, run an `MVCDRAIN EJECT` to recall the VTVs and cause the MVC to be removed from the MVC pool.

Last Mounted

the date and time that the MVC was last mounted and the VTSS where the MVC was last used.

ACS ID

the ACS where the MVC resides.

Owner/Consolidate Time

If the MVC is empty, this field is null. If the MVC is a consolidation MVC, this field displays the time of the consolidation. If the MVC is a migration MVC and contains current VTVs, this field displays the MVC's Storage Class. If no Storage Class was explicitly assigned via the `MGMTclas` statement, the default Storage Class is the name of the last VTSS that wrote to the MVC for reclamation or migration.

If VTCS receives a request to migrate a VTV that is assigned to an invalid Management Class, VTCS will dynamically create the !ERROR Storage Class and migrate the VTVs defined by the invalid Management Class to the !ERROR Storage Class. Use this Storage Class to identify and correct invalid Management Classes, drain the affected MVCs, and resubmit the request.

MVC Detailed Report

The MVC detailed report provides all the fields from the MVC summary report and a separate section that lists additional fields. Figure 88 shows an example of these additional fields from an MVC detailed report.

```
SWSADMIN (6.1.0)                STORAGETEK VTCS SYTEM UTILITY                          PAGE 0003
TIME 11:28:30                   MVC EVS102 DETAIL REPORT                               DATE 2003-06-03
VTV             SIZE                 BLOCK        MANAGEMENT       MIGRATION        BLOCK     MESSAGE
VOLSER          (MB)                    ID        CLASS            DATE             COUNT

X20041          76.00             00000000        M5               2004JAN08        10
X20043          76.00             134009C7        M5                                9
X20044          76.00             2A40138D        M5               2004JAN08        9
X20045          76.00             C6401D53        M3                                10
X20047          76.00             A5402719        M3                                10

5 VTVS FOUND FOR MVC:EVS102
WARNING VTV COUNT:5 DOES NOT MATCH MVC SUMMARY RECORD VTV COUNT:22 FOR MVC:EVS102
```

Figure 88. ***Example MVC detailed report (additional fields)***

MVC Detailed Report Additional Fields. The following list describes the additional fields for the MVC detailed report.

VTV Volser
: the volsers of the VTVs on the MVC.

Size (MB)
: the uncompressed size of the VTV (MB).

Block ID
: the logical block ID of the beginning of the VTV on the MVC.

Management Class
: the VTV's Management Class.

Migration Date
: for migrates done by a 6.0 system or above **only**, the approximate date that the VTV copy was migrated. This date is approximate because it is recorded in the CDS as the number of days since VTV creation, and time zone adjustments can cause the value to slip a day.

Block Count
: the decimal number of blocks of data that the VTV occupies on the MVC.

Message
: reports the results of MVC integrity checking. Any message **not** listed below should be reported to StorageTek, as it may indicate a serious problem with the CDS.

```
nnnn empty space to previous
```

Explanation: Informational message, indicating that there are `nnnn` (decimal) blocks of free space before this VTV on the MVC.

Flat File Record Format

Table 5 shows the record format of the flat file produced by `MVCRPT`.

Table 5. MVCRPT flat file record format

Decimal Offset	Hexadecimal Offset	Type	Length	Description
0	0	start of record		start of MVC flat file record
0	0	integer	4	record length
4	4	character	1	character set type of text fields
		X’61’		ASCII
		X’6E’		EBCDIC
5		character	1	record type’M’ (indicates an MVC report)
6	5	character	6	MVC volser
12	C	integer	4	number of current VTVs on the MVC
16	10	integer	4	percentage of the MVC used by current VTVs
20	14	integer	4	percentage of the MVC that is available for use
24	18	integer	4	percentage of the MVC that contains non-current VTVs, which is not available for use until it is reclaimed or the MVC is drained
28	1C	integer	4	number of times that the MVC has been mounted for writing or reading since it was added to the MVC inventory
32	20	time_t	4	TOD - MVC was last Mounted (starting from 01/01/1970)
36	24	integer	4	size of the MVC (MB)
40	28	time_t	4	Consolidation date/time (time_t format) or X'00'
44	2C	character	1	MVC exported (Y or N)
45	2D	character	1	MVC initialized (I or -)
46	2E	character	1	MVC broken (B or -)
47	2F	character	1	MVC lost (L or -)
48	30	character	1	MVC has data check (D or -)
49	31	character	1	MVC readonly (R or -)

Table 5. MVCRPT flat file record format

Decimal Offset	Hexadecimal Offset	Type	Length	Description
50	32	character	1	MVC Usage status: - Not usable A AUDIT status CSet CONSOLIDATE status E EXPORT status FFULL status JEJECT status NDRAIN status UUsable
51	33	character	1	MVC Retired (T or -)
52	34	character	1	MVC has invalid MIR (M or -)
53	35	character	2	ACS location of MVC
55	37	character	8	MVC was last mounted on this VTSS
63	3F	character	8	Owning VTSS name or Storage Class name

QUERY

See “DISPLAY” on page 45.

RECALL

The RECALL does demand recalls of VTVs to a VTSS.

Syntax

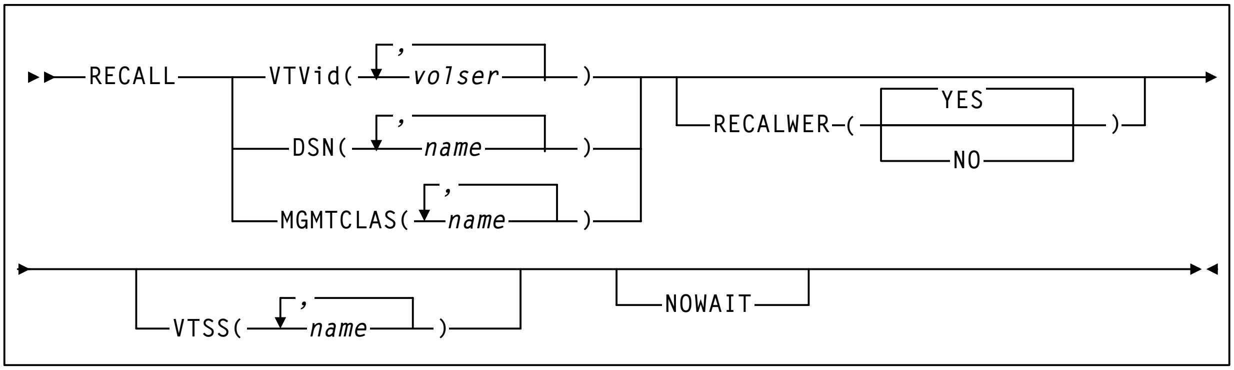


Figure 89. RECALL ***syntax***

Parameters

VTVid

specifies the VTVs that VSM recalls.

volser, vol-range, or vol-list

the volsers of one or more VTVs. You can also specify one or more ranges.

DSN

specifies data sets used to select VTVs to recall.

name

the data set name. Table 4 on page 98 describes the valid wildcard for data set names.

MGMTCLAS

specifies one or more Management Classes that determine one or more VTVs to recall.

mgmt-class-name | mgmt-class-list

the names of one or more Management Classes that you defined on the MGMTclas control statement; for more information, see “MGMTCLAS Control Statement” on page 204.

Note: The VTVid, DSN, and MGMTCLAS parameters are mutually exclusive.

`VTSS`

specifies where the VTVs are recalled as follows:

- If you do not specify a VTSS (the default), VTCS attempts to recall the VTVs to the VTSS of creation if it is accessible. Otherwise VTCS recalls the VTVs to the VTSS with the lowest DBU.
- If you specify a single VTSS, VTCS attempts to recall the VTVs to the specified VTSS if it is accessible. Otherwise, VTCS recalls the VTVs to the VTSS with the lowest DBU.
- If you specify a list of VTVs, VTCS attempts to recall the VTVs to the VTSS of creation if it is on the list and accessible, otherwise VTCS recalls the VTVs to the VTSS with the lowest DBU on the list.

`vtss-name`

the names of one or more VTSSs.

`RECALWER`

specifies whether VTCS recalls VTVs with read data checks.

`YES`

recall VTVs with read data checks (the default).

`NO`

Do not recall VTVs with read data checks.

`NOWAIT`

specifies that the utility does not wait for the operation to complete and returns after the request is submitted.

Interfaces

`SWSADMIN` utility and `VT` command.

Usage

Use the `RECALL` to do demand recalls of VTVs to a VTSS; for more information.

Command Example

To run `RECALL` to recall VTVs determined by Management Class `MCLAS1` to `VTSS1` and return immediately, enter the following:

```
.VT RECALL MGMTCLAS(MCLAS1) VTSS(VTSS1) NOWAIT
```

JCL Requirements

The following are the required and optional statements for the RECALL JCL:

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the migrate report.

SLSIN
: specifies the input to the SWSADMIN program (RECALL utility name and parameters).

JCL Example

Figure 90 shows example JCL to run RECALL to recall VTVs determined by Management Class MCLAS1 to VTSS1 and return immediately.

```
//MIGRATE     EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  RECALL MGMTCLAS(MCLAS1) VTSS(VTSS1) NOWAIT
```

Figure 90. ***Example JCL for the*** RECALL ***utility (recall by Management Class)***

RECLAIM

The `RECLAIM` does demand MVC space reclamation. The `RECLAIM` can also override the `CONFIG RECLAIM` settings for the `THRESHLD`, `MAXMVC`, and `CONMVC` parameters.

Syntax

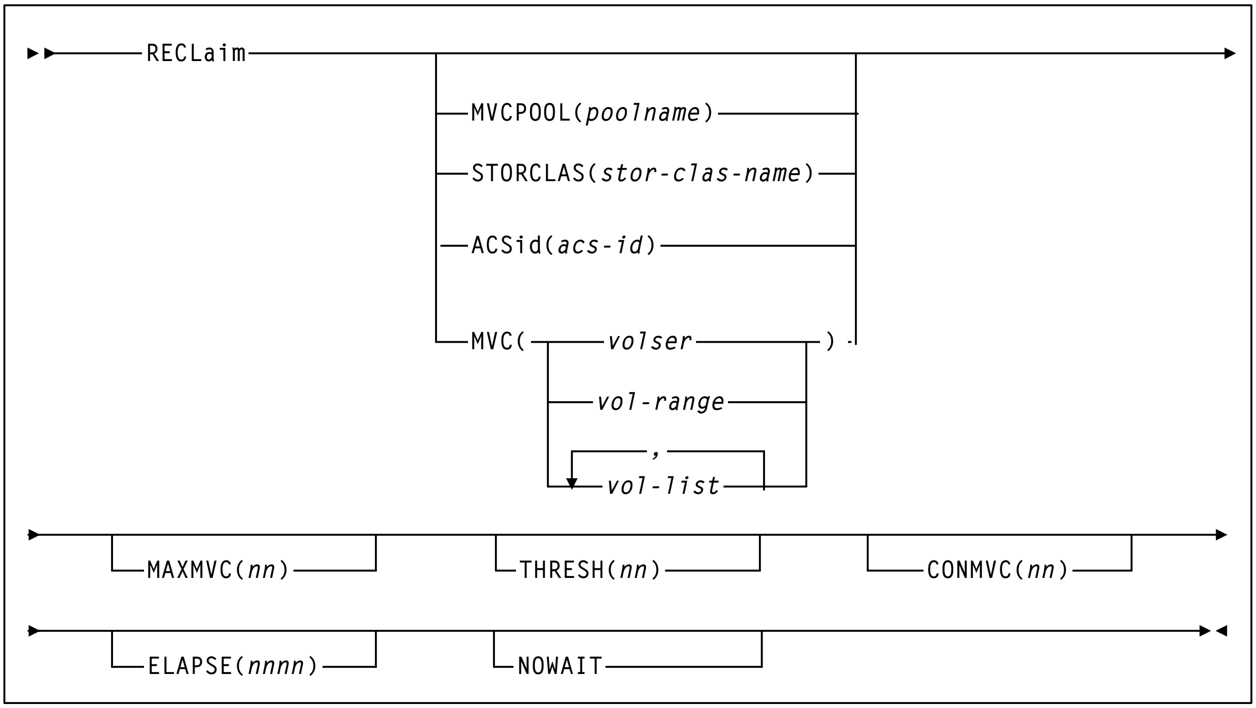


Figure 91. `RECLAIM` ***syntax***

Parameters

`MVCPOOL`

reclaim the MVCs in the specified Named MVC Pool.

`poolname`

the name of a Named MVC Pool that you defined on the `MVCPool` control statement; for more information, see “MVCPOOL Control Statement” on page 216.

`STORCLAS`

reclaim the MVCs in the specified Storage Class.

`stor-class-name`

the name of a Storage Class that you defined on the `STORclas` control statement; for more information, see “STORCLAS Control Statement” on page 219.

`ACSid`

reclaim the eligible MVCs in the specified ACS.

`acs-id`

The specified ACS.

`MVC`

reclaim the specified MVC(s).

`volser`, `vol-range` or `vol-list`

the volsers of one or more MVCs.You can also specify individual MVCs and ranges in a list. For example:

`(MVC000-MVC005,MVC010,MVC015)`

`MAXMVC(nn)`

specifies the maximum number of MVCs that will be processed by a single space reclamation task. Valid values are 1 to 98. There is no default; if not specified, the `CONFIG RECLAIM` value (or default) is used.

For automatic space reclamation to start, the number of eligible MVCs (determined by the `THRESH` parameter) must also exceed the `MAXMVC` value.

`THRESH(nn)`

specifies the percentage of fragmented space that makes an MVC eligible for demand or automatic reclamation. Valid values are 4 to 98. If not specified, the `CONFIG RECLAIM` value (or default) is used.

`NOWAIT`

specifies that the utility does not wait for the operation to complete and returns after the request is submitted.

`CONMVC(nn)`

specifies the maximum number of MVCs that VTCS concurrently processes for both drain and reclaim.

Valid values are 1 to 99. If not specified, the default is the `CONMVC` value specified on the `CONFIG RECLAIM` statement.

`ELAPSE(nnnn)`

specifies the maximum time for the reclaim in minutes. If the maximum time expires, VTCS issues message SLS6682I. If there are no MVCs currently mounted, reclaim stops when the `ELAPSE` value is reached. If any MVCs are currently mounted when the `ELAPSE` value is reached, reclaim processes the mounted MVCs and then stops.

Valid values are 1 to 1440. If not specified, there is no time limit on the reclaim process.

Interfaces

`SWSADMIN utility and VT` command.

Usage

Use the RECLAIM to do demand MVC space reclamation.

To select the MVCs to reclaim, you can specify one of the following parameters:

- MVCPOOL to reclaim the MVCs in a Named MVC Pool.
- STORCLAS to reclaim the MVCs in a Storage Class.
- ACSid to reclaim the eligible MVCs in a single ACS.
- MVC to reclaim the reclaim the specified MVC(s).

You can specify only one of the MVCPOOL, STORCLAS, ACSid, or MVC parameters. If you do not specify one of these parameter, space reclamation selects MVCs from the Named MVC Pool (if implemented) or media type (for multiple MVC media environments) most in need of free space.

VSM reclaims space on only those MVCs whose percentage of fragmentation and the resources required to move the VTVs justify the reclamation. The CONFIG RECLAIM THRESHLD parameter determines when an MVC is considered for reclamation, and free space must exist on the MVC for reclamation processing. The CONFIG RECLAIM MAXMVC parameter specifies the maximum number of MVCs that will be processed by the RECLAIM. You can use the RECLAIM to override the CONFIG RECLAIM THRESHLD, MAXMVC, and CONMVC settings.

You can also specify the maximum time for the reclaim in minutes on the ELAPSE parameter. Note that there are several limiting factors that influence reclaims (for example, MAXMVC and ELAPSE). VTCS enforces the strictest limiting factor. For example, if you run RECLAIM and specify ELAPSE equal to 5 hours and MAXMVC equal to 10 and VTCS reclaims 10 MVCs in one hour, then VTCS terminates the reclaim before the ELAPSE value expires.

☞ VTCS and HSC must be active to process a RECLAIM request.

Command Examples

To reclaim the MVCs in Storage Class STORCL1, override the CONFIG RECLAIM MAXMVC and THRESHLD settings, and return after the request is submitted, enter the following:

```
.VT RECLAIM STORCLAS(STORCL) MAXMVC(20) THRESH(70)
```

To reclaim MVCs MVC000 - MVC015, override the CONFIG RECLAIM MAXMVC and THRESHLD settings, and return after the request is submitted, enter the following:

```
.VT RECLAIM MVC(MVC000-MVC015) MAXMVC(20) THRESH(70)
```

JCL Requirements

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the RECLAIM report.

SLSIN
: specifies the input to the SWSADMIN program (RECLAIM utility name and parameters).

JCL Examples

Figure 92 shows example JCL to reclaim the MVCs in Storage Class STORCL1, override the CONFIG RECLAIM MAXMVC and THRESHLD settings, and return after the request is submitted.

```
//RECLAIM  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB       DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT      DD SYSOUT=*
//SLSIN         DD *
 RECLAIM STORCLAS(STORCL) MAXMVC(20) THRESH(70)
```

Figure 92. RECLAIM ***utility example: reclaiming MVCs by Storage Class***

Figure 93 shows example JCL to reclaim MVCs MVC000 - MVC015, override the CONFIG RECLAIM MAXMVC and THRESHLD settings, and return after the request is submitted.

```
//RECLAIM  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB       DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT      DD SYSOUT=*
//SLSIN         DD *
 RECLAIM MVC(MVC000-MVC015) MAXMVC(20) THRESH(70)
```

Figure 93. RECLAIM ***utility example: reclaiming MVCs by MVC ID***

RTV Utility

The RTV utility converts VTVs contained on MVCs to data sets on Nearline volumes (real tape volumes).

Syntax

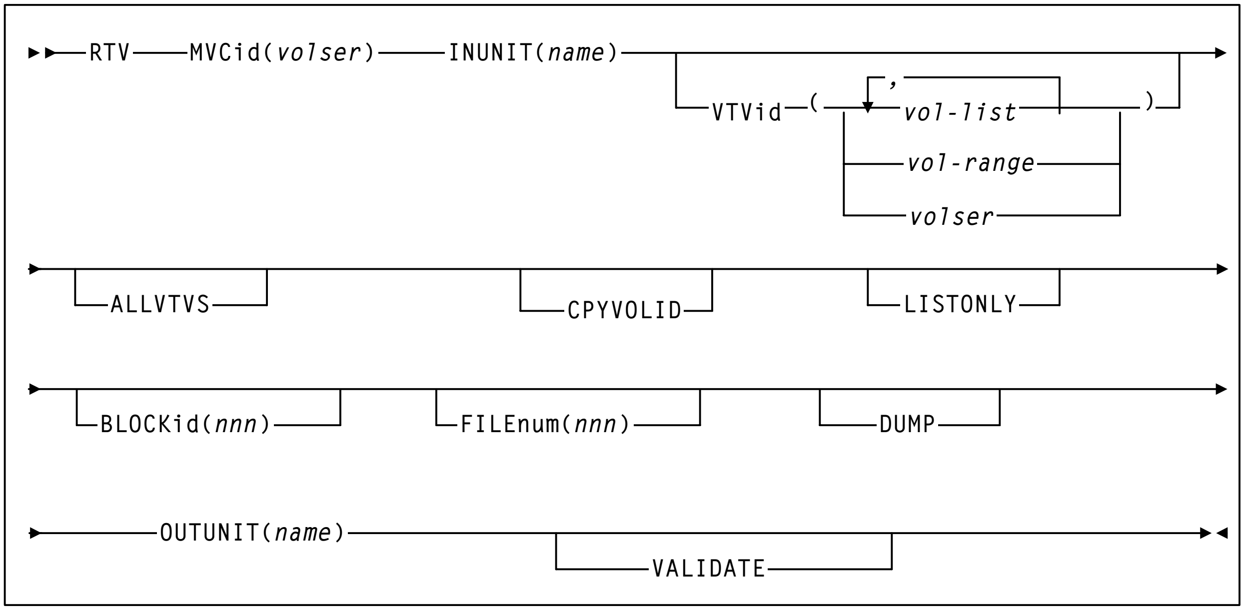


Figure 94. RTV ***utility syntax***

Parameters

MVCid

specifies the MVC that contains the VTVs that RTV converts to Nearline volume(s) which become real tape versions of the VTVs.

volser

the MVC volser.

INUNIT

the name to use to allocate the input tape unit. You can specify an MSP unit address, an esoteric name, or a generic name. The valid values are the same as for the UNIT= JCL parameter.

name

the unit name.

VTVid

specifies one or more VTVs to convert.

volser, vol-range, or vol-list

the volsers of one or more VTVs.

ALLVTVS

convert the most current copy of all VTVs on the specified MVC. That is, if there are multiple copies of a VTV on the specified MVC, RTV only converts the most current copy of the VTV.

Note: The VTVid and ALLVTVS parameters are mutually exclusive.

CPYVOLID

copy the VTV internal volser from the VTV to the output volume VOL1 record. The default is to not copy the VTV VOLID.

Caution: Use the CPYVOLID parameter carefully! The volser of the output tape will be changed to the volser of the VTV. If the output tape has an external label or if the output is directed to another VTV, this will cause label mismatches and can cause unpredictable and undesirable results.

Note:

- If the output tape is non-labelled or has a non-standard tape label, CPYVOLID will be automatically specified for this VTV decompression, and a standard label tape will be created on the output device.
- RTV supports VTVs created with standard or ANSI labels. If you do not specify CPYVOLID, RTV processes these label types as described in Table 6. Note that this only applies to the VOL1 record. The HDR1/HDR2 labels are always copied from the VTV by RTV for every VTV processed.

Table 6. RTV VTV Label Processing (CPYVOLID Not Specified)

VTV Label Type	Output Standard Label	Output ANSI Label	Output Non-Label
Standard label	VOL1 label not copied	WTOR issued	VOL1 label is copied
ANSI label	WTOR issued	VOL1 label not copied	VOL1 label is copied

In Table 6, the WTOR is as follows:

```
SWSRTV - Label mismatch - Reply RELABEL, RETRY, or CANCEL
```

The operator responses produce the following results:

RELABEL

Decompress the RTV and overwrite the volser on the output volume.

RETRY

Mount another output volume and retry the operation.

CANCEL

Do not decompress the RTV.

LISTONLY

lists (but does not convert) the VTVs on the specified MVC. For more information, see “RTV LISTONLY Listing” on page 152.

`BLOCKID`

the logical block ID where the VTV begins on the MVC.

`nnn`

the logical block ID (8 hexadecimal characters).

Hint: The `LISTONLY` parameter listing on “RTV LISTONLY Listing” on page 152 supplies a Block ID value that you can use as input to the `RTV` utility to convert a VTV to a Nearline volume.

`FILEnum`

the logical data set number of VTV on the MVC.

`nnnnn`

the logical data set number (1 to 5 decimal characters).

Note: The `LISTONLY`, `BLOCKid`, and `FILEnum` parameters are mutually exclusive. In addition, if you specify the `ALLVTVs` parameter, or if a list or range of VTVs is specified, the `FILEnum` parameter is ignored.

`DUMP`

produce a S0C3 abend dump if RTV cannot decompress a VTV. If you specify `DUMP`, create a `SYSMDUMP DD` JCL statement to capture the dump.

`OUTUNIT`

the name to use to allocate the output tape unit. You can specify an MSP unit address, an esoteric name, or a generic name. The valid values are the same as for the UNIT= JCL parameter. This parameter is required if you do not specify `LISTONLY`.

`name`

the unit name.

Interfaces

`SWSRTV` only.

Usage

Use the `RTV` utility when data is required that resides on VTVs that have been migrated to MVCs and no active VSM system is available. `RTV` takes a VTV from an MVC, decompresses the VTV, then writes the data to a single output tape (real tape volume) so the data can be read by user applications. The `RTV` utility is a stand–alone utility; that is, you can run this utility when VSM is down but the MSP system is up. Also note that the `RTV` utility is included on the VTCS product tape and is also available for download from the StorageTek Customer Resource Center (CRC). Both versions are identical.

If you downloaded the `RTV` utility from the CRC for use at a site that does not have NCS/VTCS software installed, you must do the following:

- Install the downloaded version of `RTV` per instructions on the CRC.
- APF authorize the `RTV` load library you just created.
- In “JCL Examples” on page 147, substitute the DSN you picked for the `RTV` load library for DSN `hlq.SLSLINK` on the `STEPLIB DD` statement(s).

What the RTV Utility Can Recover

The `RTV` utility can recover:

- All or specified VTVs from a specified MVC. If you do not know the location of the most current version of a VTV on the MVC, specify only the VTV volser, and `RTV` will convert the most current version of the VTV it finds on this MVC.
- A VTV at a specified block ID on a specified MVC. The `LISTONLY` parameter listing on “RTV LISTONLY Listing” on page 152 supplies a Block ID value that you can use as input to the `RTV` utility to convert a VTV to a Nearline volume. Specifying the volser and Block ID speeds positioning time.
- A VTV specified by logical data set number on a specified MVC. Specifying the volser and logical data set number will have a much longer positioning time compared to specifying volser and Block ID. Using volser and Block ID is the preferred method to access a single VTV.

Note: If more than one VTV is specified, or if no `BLOCKid` or `FILEnum` parameter is specified, the entire MVC will be read and the MVC contents displayed as part of the output. Reading of the entire MVC is necessary to insure that only the most current copy of a VTV is decompressed.

General Usage Guidelines

- The output volume that contains the converted VTV(s) must be at least 800 Mb (capacity of a 3490E cartridge) to ensure that it can contain an individual VTV.
- The VTCS MVCRPT on page 121 and VTCS VTVRPT on page 173 produce VTV and MVC reports that provide information to specify which copy of a VTV you want RTV to recover. Ensure that you have a current copy of these reports before you run the RTV utility. In addition, to help identify the VTVs you want to convert, you can use the LISTONLY parameter to produce a list of the VTVs on an MVC.

  Because multiple copies of the same VTV can exist on the same or different MVCs, **study carefully** your VTV and MVC reports and LISTONLY listings to ensure that you are using the correct MVC to convert the most current copy of a VTV!
- The RTV utility does not update the system catalog or TMC with information about the converted volumes; you must do this manually.

Security Considerations

- You must have read access both to the VTVs you want to convert and to the MVC that contains these VTVs or your system's security application cannot be running. Otherwise, the conversion will fail.
- Ensure that you APF authorize the RTV utility load library.
- RTV makes no attempts to bypass any TMS protection. All RTV tape mounts are subject to full TMS control.

JCL Requirements

The following are required or optional statements for the RTV utility JCL:

STEPLIB
: specifies the link library (SLSLINK) that contains the RTV modules.

SLSPRINT
: specifies the destination of the RTV utility report.

SLSIN
: specifies the input to the SWSRTV program (RTV utility name and parameters).

SYSMDUMP
: optional DD to capture dump.

Note: Because the RTV utility must be capable of rewriting the tape standard labels on the output unit and positioning over label information on the input unit, Dynamic Allocation is used to invoke bypass label processing (BLP) on the tape volumes. This requires that the library that contains the SWSRTV executable code be APF authorized.

JCL Examples

Listing the VTVs on an MVC

Figure 95 shows example JCL to lists the VTVs on MVC MVC001.

```
//JOBVRECJOB (account),programmer
//RUNRTV EXEC PGM=SWSRTV,PARM='MIXED'
//STEPLIB    DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT   DD SYSOUT=A
//SLSIN      DD *
  RTV  MVC(MVC001)INUNIT(/1AB4) LISTONLY
/*
//
```

Figure 95. ***Example JCL to run the*** `RTV` ***utility: LISTONLY run***

Converting a Single VTV by Specifying Its Volser

Figure 96 shows example JCL to run the `RTV` utility to convert VTV VTV200 on MVC MVC001, which will be mounted on a 3490E transport. The output (converted VTV VTV200) goes to the output volume mounted on transport 280, and `RTV` copies the VTV VOLID from the VTV to the output volume.

```
//JOBVRECJOB(account),programmer
//RUNRTV EXEC PGM=SWSRTV,PARM='MIXED'
//STEPLIB    DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT   DD SYSOUT=A
//SLSIN      DD *
  RTV  MVC(MVC001) INUNIT(3490E) VTV(VTV200) CPYVOLID OUTUNIT(280)
/*
//
```

Figure 96. ***Example JCL to run the*** `RTV` ***utility: single VTV by volser***

Converting a Single VTV by Specifying Its Volser and Block ID

Figure 97 shows example JCL to run the `RTV` utility to convert VTV VTV200 at block ID x'8EA484AB' on MVC MVC001, which will be mounted on a 3490E transport. The output (converted VTV VTV200) goes to the output volume mounted on transport 480.

```
//JOBVRECJOB   (account),programmer
//RUNRTV EXEC  PGM=SWSRTV,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=A
//SLSIN        DD *
  RTV  MVC(MVC001) INUNIT(3490E) VTV(VTV200) BLOCK(8EA484AB)
OUTUNIT(480)
```

Figure 97. ***Example JCL to run the*** `RTV` ***utility: single VTV by volser and block ID***

RTV Utility Report Messages

The RTV report displays the following messages:

`Block number too large in compressed data`

Explanation: An error was found in a compressed data record while processing a VTV. Further processing of this VTV is stopped. The VTV will be marked in the RTV Decompress Report output. The utility will continue to process other VTVs as necessary.

`Chunked record logic error`

Explanation: An error was found while processing a chunked data record for a VTV. Further processing of this VTV is stopped. The VTV will be marked in the RTV Decompress Report output. The utility will continue to process other VTVs as necessary.

`Decompress invalid length parameter`

Explanation: This indicates a program logic error. Further processing of this VTV is stopped. The VTV will be marked in the RTV Decompress Report output. The utility will continue to process other VTVs as necessary.

`Decompress invalid parameter list`

Explanation: This indicates a program logic error. Further processing of this VTV is stopped. The VTV will be marked in the RTV Decompress Report output. The utility will continue to process other VTVs as necessary.

`Decompress logic error`

Explanation: This indicates a program logic error. Further processing of this VTV is stopped. The VTV will be marked in the RTV Decompress Report output. The utility will continue to process other VTVs as necessary.

`Decompress pointer to work area is zero`

Explanation: This indicates a program logic error. Further processing of this VTV is stopped. The VTV will be marked in the RTV Decompress Report output. The utility will continue to process other VTVs as necessary.

`Dynamic allocation error. Reason Code = xxxx-xxxx`

Explanation: An error was encountered while attempting to dynamically allocate the INUNIT or OUTUNIT device. Refer to the IBM manual *MSP Authorized Assembler Services Guide* for a description of the dynamic allocation reason codes.

`FILEnum of zero is invalid`

Explanation: A `FILEnum()` value of 0 is invalid. The utility terminates with return code 12.

`I/O error on input MVC`

Explanation: An I/O error was encountered while reading a MVC. Further processing is stopped. The utility terminates with return code 12.

`I/O error on output volume`

Explanation: An I/O error was encountered while writing the output VTV. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`Invalid compressed data block read`

Explanation: This indicates that an invalid data record was found while processing this VTV. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`Invalid VTV page number encountered`

Explanation: A record sequence error was found in a compressed data record while processing a VTV. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`Invalid VTV record encountered`

Explanation: An error was found in a compressed data record while processing a VTV. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`MVC volser # does not match requested volser #`

Explanation: The volume mounted as the input MVC did not match that requested by the `MVCid()` parameter. The utility terminates with return code 12.

`MVC record length error`

Explanation: A length error was found in a compressed data record while processing a VTV. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`No algo byte found in compressed data`

Explanation: An error was found in a compressed data record while processing a VTV. Further processing of this VTV is stopped. The VTV will be marked in the RTV Decompress Report output. The utility will continue to process other VTVs as necessary.

`No HDR1 record found for requested VTV`

Explanation: Following positioning by a `BLOCKid()` statement, there was no HDR1 record located at the desired position. Remove the `BLOCKid` statement and rerun the utility. The utility terminates with return code 12.

`No HDR1 record found on input MVC`

Explanation: The volume mounted as a MVC contained no HDR1 record. The utility terminates with return code 12.

`No UHL1 record found on input MVC`

Explanation: The volume mounted as a MVC contained no UHL1 record. The utility terminates with return code 12.

`No VOL1 record found on input MVC`

Explanation: The volume mounted as a MVC contained no VOL1 record. The utility terminates with return code 12.

`NULL input buffer pointer`

Explanation: This indicates a program logic error. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`NULL output buffer pointer`

Explanation: This indicates a program logic error. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`Requested VTV not found on MVC`

Explanation: The volser requested by the `VTVid()` parameter was not found on the MVC. The utility terminates with return code 12.

`Spanned length final error`

Explanation: An error was found while processing a spanned data record for a VTV. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`Spanned length intermediate error`

Explanation: An error was found while processing a spanned data record for a VTV. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`Unexpected request on input I/O`

Explanation: This indicates a program logic error. Further processing is stopped. The utility terminates with return code 12.

Unexpected end of tape on output volume

While writing the output VTV, an end of tape indication was encountered. The VTV must be completely contained on a single output volume. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`Unexpected request on output I/O`

Explanation: This indicates a program logic error. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`Unexpected tape mark on input MVC`

Explanation: An unexpected tape mark was found on a MVC. Further processing is stopped. The utility terminates with return code 12.

`VTVid range parameter is invalid`

Explanation: An invalid range value was found in the `VTVid()` specification. The utility terminates with return code 12.

`VTV logical data check encountered`

Explanation: A data check indicator was found in a compressed data record while processing a VTV. Further processing of this VTV is stopped. The VTV will be marked in the `RTV` Decompress Report output. The utility will continue to process other VTVs as necessary.

`VTV volser # does not match requested volser #`

Explanation: Following positioning by a `BLOCKid()` or `FILEnum()` statement, the VTV volser did not match that requested by the VTVid() parameter. The utility terminates with return code 12. Remove the `BLOCKid` or `FILEnum()` statement and rerun the utility.

RTV LISTONLY Listing

Figure 98 shows an example of the listing that `RTV` produces when you specify the `LISTONLY` parameter.

```
SWSRTV                   (1.0.0)                StorageTek VTCS RTV Utility

 PAGE 0001

TIME 14:23:33                                    Control Card Image Listing

DATE 12/01/00

        RTV MVC(C83107) LISTONLY

SWSRTV                   (1.0.0)                StorageTek VTCS RTV Utility

 PAGE 0002

TIME 14:23:33                                    MVC C83107 Contents Report

DATE 12/01/00

VTV             File     Block                 <---Last Used--->                   VTV

Volser          #        ID                 Date            Time                   Status

VV6825          1        00000000         20020Nov30       12:07:56

VV6863          2        92002F0F         20030Sep27       12:57:54

VV6893          3        92002F18         20030Aug18       08:57:26

VV0403          4        92002F21         20030Aug18       08:57:26
```

Figure 98. Example `RTV LISTONLY` ***listing***

This report lists the VTV's:

- Volser
- Logical file number on the MVC
- Block ID on the MVC
- Last used time
- Status - `Not Current`, or if blank, the VTV is current

RTV Decompress Listing

Figure 99 shows an example of the listing that `RTV` produces when you do not specify the `LISTONLY` parameter (that is, you run RTV to convert VTVs to Nearline volumes).

```
SWSRTV                  (1.0.0)                StorageTek VTCS RTV Utility

 PAGE 0001

TIME 14:28:33                                   Control Card Image Listing                                 DATE 2003-1-18

            RTV MVC(C8228) VTV(VV6800-VV6900) CPYVOLID

SWSRTV                  (1.0.0)                StorageTek VTCS RTV Utility

 PAGE 0002

TIME 14:28:33                                   MVC C83223 Contents Report                                 DATE 2003-1-18

VTV             File    Block              <---Last Used--->                   VTV

Volser          #       ID             Date            Time                    Status

VV6070          1       00000000       20020Nov30      12:07:56

VV0874          2       2B001384       20030Sep27      12:57:54

VV0772          3       A3002707       20030Aug18      08:57:26

VV6828          4       9B002AB9       20030Aug18      08:57:26                Not current

VV6828          5       9B002AC2       20030Aug18      08:57:26

VV6826          6       9B002ACB       20030Aug18      08:57:26

SWSRTV                  (1.0.0)                StorageTek VTCS RTV Utility

 PAGE 0003

TIME 14:28:33                                  MVC C83223 Decompress Report                                DATE 2003-1-18

VTV                                Mounted                                  Final                       Decompress

Volser                             Volser                                   Volser                        Status

VV6826                             XX0772                                   VV6826                      Successful

VV6828                             XX0773                                   VV6828                      Successful
```

***Figure 99.* Example `RTV` Decompress Listing**

In addition to the contents fields shown in Figure 98 on page 152, the decompress listing shown in Figure 99 on page 153 lists the VTV's:

- Volser of the output Nearline volume as initially mounted
- Final volser of the output Nearline volume; if `CPYVOLID` is specified, the final volser will be identical to the VTV volser, otherwise is final volser is identical to the volser of the output Nearline volume as initially mounted
- Decompress status

SET MIGOPT

SET MIGOPT changes the following migration parameters:

- Maximum and minimum concurrent automatic migration, immediate migration, and migrate-to-threshold tasks
- High and low AMTs

Syntax

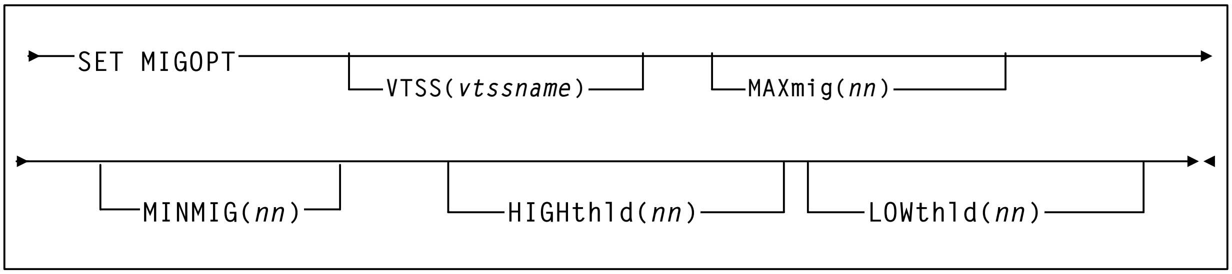


Figure 100. SET MIGOPT ***syntax***

Parameters

VTSS

the VTSS whose migration parameters you want to change. If you do not specify a VTSS, the changes affect all VTSSs.

vtssname

the VTSS identifier.

MAXMIG(nn)

specifies the maximum number of concurrent automatic migration, immediate migration, and migrate-to-threshold tasks.

Valid values are 1 to the number of RTDs on the VTSS or 8, whichever is less. There is no default; if you do not specify a value, the current value is unchanged.

MINMIG(nn)

specifies the minimum number of concurrent automatic migration, immediate migration, and migrate-to-threshold tasks.

Valid values are 1 to the MAXMIG setting. There is no default; if you do not specify a value, the current value is unchanged.

HIGHthld

specifies the new high AMT.

high-thr

the new high AMT as a percent of VTSS space. Valid values are 5 to 95 and must be greater than the LOWthld value.

`LOWthld`
: specifies the new low AMT.

  `low-thr`
  : the new low AMT as a percent of VTSS space. Valid values are 5 to 95 and must be less than the `HIGHthld` value.

Interfaces

`SWSADMIN` utility and `VT` command.

Usage

Use `SET MIGOPT` to change following:

- Maximum and minimum concurrent automatic migration, immediate migration, and migrate-to-threshold tasks.
- High and low AMTs.

.Using `SET MIGOPT`:

- You can change the LAMT, the HAMT, or both.
- Changes to the AMTs take effect immediately.
- If you try to change global values (no VTSS specified) and the values are not valid for one VTSS (for example, `MAXMIG(5)` and one VTSS only has 4 RTDs connected), VTCS will not change values for any VTSSs.

Note: You can also change the AMTs with either the `MIGrate` or the `CONFIG VTSS LOW` and `HIGH` parameters; see "MIGRATE" on page 98 and "`VTSS`" on page 5. Note, however, that a future release of VTCS will drop support of the `MIGRATE SET HIGHthld` and `LOWthld` parameters.

Command Examples

To change `MAXMIG` to 5 and `MINMIG` to 3, enter:

```
.VT SET MIGOPT MAXMIG(5) MINMIG(3)
```

To change the high AMT to 70% and the low AMT to 25%, enter:

```
.VT SET MIGOPT HIGH(70) LOW(25)
```

JCL Requirements

`STEPLIB`
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

`SLSPRINT`
: specifies the destination for the `RECLAIM` report.

`SLSIN`
: specifies the input to the `SWSADMIN` program (`RECLAIM` utility name and parameters).

JCL Examples

Figure 101 shows example JCL to change MAXMIG to 5 and MINMIG to 3.

```
//RECLAIM  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  SET MIGOPT MAXMIG(5) MINMIG(3)
```

Figure 101.RECLAIM JCL example to change MAXMIG to 5 and MINMIG to 3

Figure 102 shows example JCL change the high AMT to 70% and the low AMT to 25%.

```
//RECLAIM  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  SET MIGOPT HIGH(70) LOW(25)
```

Figure 102.RECLAIM JCL to change the high AMT to 70% and the low AMT to 25%

TRACE

TRace starts or stops event tracing for specified VTCS components.

Syntax

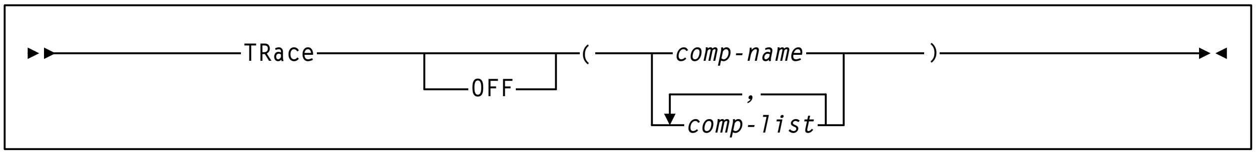


Figure 103. TRace ***syntax***

Parameters

OFF
: stops tracing for the specified components.

comp-name
: specifies one of the following components:

  VTCS
  : traces the VTCS component.

comp-list
: specifies a list of components separated by commas or blanks.

Use TRace to start or stop event tracing for specified VTCS components.

Usage

Do not run a trace unless a StorageTek software support representative asks you to do so. You then send the trace file to StorageTek for diagnosis.

Interfaces

SWSADMIN utility and VT command.

Command Example

To start tracing for the VTCS component requests, enter the following:

```
.VT TR VTCS
```

JCL Requirements

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the RECLAIM report.

SLSIN
: specifies the input to the SWSADMIN program (function name and parameters).

JCL Example

Figure 104 shows example JCL start tracing for the VTCS component requests.

```
//RECLAIM  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
  TR VTCS
```

Figure 104.JCL example to start tracing for the VTCS component requests

VARY CLINK

`Vary CLInk` changes CLINK states.

Syntax

```
►►──Vary──CLInk(clink-id)─────────VTSS(vtssname)───────────────┬─ONline──┬──────►◄
                                                               └─OFFline─┘
```

Figure 105. `Vary CLINK` ***syntax***

Parameters

`CLInk`
: the specified CLINK.

  `clink-id`
  : the link ID.

`VTSS`
: the sending VTSS in the Cluster.

  `vtssname`
  : the 1 to 8 character identifier of the sending VTSS.

`ONline`
: vary the specified CLINK online.

`OFFline`
: vary the specified CLINK offline.

Interfaces

`SWSADMIN` utility and `VT` command.

Usage

Use `Vary CLINK` to change CLINK states. For example, if a CLINK fails or requires service, you can enter a `VT Vary CLink OFFline` command to vary the CLINK offline. You enter a `VT Vary CLink ONline` command to vary the CLINK online.

Table 7 describes CLINK states.

Table 7. CLINK States

If you specify the following `Vary CLink` parameter...	The CLINK first goes to state...	And then goes to state...
`ONline`	**Online Pending** - In online pending state, the online process has started but has not completed.	**Online** - In online state, the CLINK is online and available for replication work.
`OFFline`	**Offline Pending** - In offline pending state, the offline process has started but has not completed. Current allocations continue to complete the replication of any VTVs being handled when the VARY OFF was issued. When all current allocations complete, the CLINK goes to offline state for all active hosts.	**Offline** - in offline state, The CLINK is offline to all hosts and does not accept replication work.

Command Example

To vary online CLINK `7` on the Cluster with VTSS `VTSS01` as the sending VTSS, enter:

```
.VT V CLI(7) VTSS(VTSS01) ON
```

JCL Requirements

`STEPLIB`
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

`SLSPRINT`
: specifies the destination for the `RECLAIM` report.

`SLSIN`
: specifies the input to the `SWSADMIN` program (function name and parameters).

JCL Example

Figure 106 shows example JCL to vary online CLINK `7` on the Cluster with VTSS `VTSS01` as the sending VTSS.

```
//RECLAIM  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *

       V CLI(7) VTSS(VTSS01) ON
```

Figure 106. JCL example to vary CLINK 7 on Cluster CLUSTER1 online

VARY RTD

`Vary RTD` changes RTD states.

Syntax

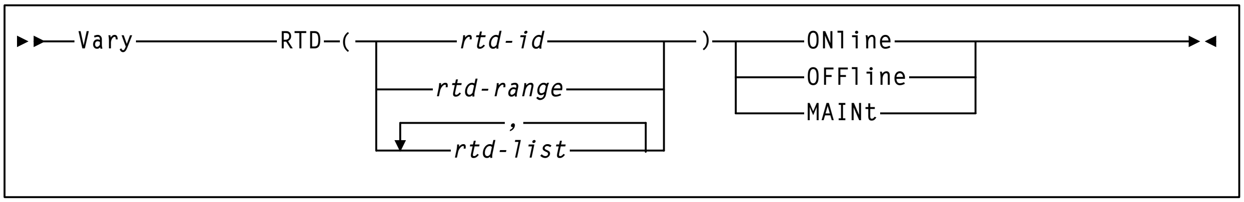


Figure 107. `Vary RTD` ***syntax***

Parameters

`RTD`
: change the state of the specified RTDs.

  `rtd-id`, `rtd-range`, or `rtd-list`
  : the unit addresses of one or more RTDs. Lists and ranges of RTDs are limited to 64 items for VSM2s and VSM3s and 256 items for VSM4s.

`ONline`
: vary the specified RTDs online to their connected VTSSs.

`OFFline`
: vary the specified RTDs offline to their connected VTSSs.

`MAINt`
: vary the specified RTDs offline (maintenance mode) to their connected VTSSs.

Interfaces

`SWSADMIN` utility and `VT` command.

Usage

Use `Vary RTD` to change RTD states. The state change applies to all VTSSs connected to the specified RTDs. That is, an RTD must be in the same state to all connected VTSSs.

Caution: If you are sharing transports with MSP, make sure to vary the transports offline to all MSP systems before varying these RTDs online to VSM and vice versa.

Command Example

To vary RTD B10 online, enter:

```
.VT V RTD(B10) ON
```

JCL Requirements

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the RECLAIM report.

SLSIN
: specifies the input to the SWSADMIN program (function name and parameters).

JCL Example

Figure 108 shows example JCL to vary RTD B10 online.

```
//RECLAIM  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
       V RTD(B10) ON
```

Figure 108.JCL example to vary RTD B10 online

VARY VTSS

`Vary VTSS` changes VTSS states on all hosts.

Syntax

```
►►──Vary──────────VTSS(───────name───────)─────────┬──OFFline──┬────────────►◄
                                                   ├──ONline───┤
                                                   └──QUIESCED─┘
```

Figure 109. `Vary VTSS` ***syntax***

Parameters

`VTSS`

change the state of the specified VTSS. For more information about VTSS states, see Table 8 on page 164.

`vtssname`

the VTSS identifier.

`ONline`

vary the specified VTSS online.

`OFFline`

vary the specified VTSS offline.

`QUIESCED`

vary the specified VTSS to quiesced state.

Note: `Vary VTSS` **does not** change the state of the VTDs or RTDs associated with the specified VTSS!

Interfaces

`SWSADMIN` utility and `VT` command.

Usage

Table 8 describes VTSS states.

Table 8. ***VTSS States***

If you specify the following `Vary VTSS` parameter...	The VTSS first goes to state...	And then goes to state...
`ONline`	**Online Pending** - In online pending state, the online process has started but has not completed on all hosts.	**Online** - In online state, the VTSS is online, available, and accepts both front-end and back-end work. If the VTSS was offline, when it goes online, VTCS issues a warning message recommending a VTSS audit.
`QUIESCED`	**Quiescing** - In quiescing state, VTCS does not direct any DD allocation to the VTSS, which still accepts pending mounts to allow those long running jobs with `unit=aff` chains to complete. When all VTDs are no longer in use (their UCBs are not allocated on MSP), the VTSS goes to quiesced state. In quiescing state, the VTSS continues to accept and process back-end work; for example, migrates, recalls, and audits.	**Quiesced** - In quiesced state, the VTSS continues to accept and process back-end work; for example, migrates, recalls, and audits. That is, you can use the recall and migrate commands and utilities to do these operations using the quiesced VTSS.
`OFFline`	**Offline Pending** - In offline pending state, the offline process has started but has not completed on all hosts. VTCS immediately shuts down the VTSS and interrupts and purges all active tasks and purges all queued tasks. The VTSS server task terminates and no longer accepts new front-end and back-end work. VTCS creates new VTVs and mounts/dismounts existing VTVs only on alternate VTSSs, if they are available.	**Offline** - in offline state, The VTSS is offline to all hosts and does not accept either front-end or back-end work. If a copy of a VTV is resident on an offline VTSS and also on an MVC and a job requires the VTV, VTCS automatically recalls the VTV to an alternate VTSS, if available.

Use `Vary VTSS` to change VTSS states as described in Table 9.

Note: `Vary VTSS` changes VTSS states on all hosts regardless of which host issued the command or ran the utility.

Table 9. `Vary VTSS` ***Usage Scenarios***

If...	And you want to...	Do the following...
A VTSS fails	Take the VTSS offline and continue processing by recalling migrated VTVs to an alternate VTSS.	Vary the failed VTSS offline. Jobs that require VTVs that have migrated from the failed VTSS are automatically recalled to the alternate VTSS.
A VTSS requires maintenance or will be taken out of service.	Take the VTSS offline and continue processing by recalling migrated VTVs to an alternate VTSS.	1. Vary the VTSS to quiesced state. 2. If the VTSS will be taken out of service or the maintenance requires a "clean" VTSS, Do a migrate-to-threshold 0%. When the migrate completes, do a VTSS audit to ensure that all VTVs were successfully migrated. 3. Vary the VTSS offline so that maintenance can be applied or it can be taken out of service. Jobs that require VTVs that have migrated from the failed VTSS are automatically recalled to the alternate VTSS. **Note:** It may also be necessary to vary VTDs and associated paths offline to MSP.

If...	And you want to...	Do the following...
An offline VTSS is ready to come back online.	Use the VTSS.	1. Dismount any VTVs still mounted in the VTSS (MSP perspective), by doing either of the following: • Use the MSP UNLOAD command to dismount the VTVs. • Use the VARY OFFLINE command to vary offline the VTD where the VTV is mounted, which will also dismount the VTV. 2. Clear any boxed VTD conditions. 3. Vary the VTSS to quiesced state. 4. Audit the VTSS. 5. Vary the VTSS online.

Command Example

To vary VTSS VTSS01 online, enter:

```
.VT V VTSS(HVTSS10) ON
```

The following shows sample output from this `VT Vary VTSS ONline` command:

```
Vary submitted to VSM system
Vary completed (0)
Vary online of VTSS HVTSS10 initiated from host ECCY
RTD task starting for device V0EE22F0
RTD task starting for device V0EE22F1
RTD task starting for device V08A22F4
VTSS HVTSS10 now online on host ECCL
VTSS HVTSS10 now online on host MSPU
VTSS HVTSS10 now online on host EC31
Vary online (local) of VTSS HVTSS10 complete
VTSS HVTSS10 has been offline; a VTSS audit is recommended
VTSS HVTSS10 server ready; state is online
VTSS HVTSS10 now online on host ECCY
Vary online (global) of VTSS HVTSS10 complete
```

JCL Requirements

`STEPLIB`
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

`SLSPRINT`
: specifies the destination for the `RECLAIM` report.

`SLSIN`
: specifies the input to the `SWSADMIN` program (function name and parameters).

JCL Example

Figure 108 shows example JCL to vary VTSS VTSS01 online.

```
//RECLAIM  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
       V VTSS(HVTSS10) ON
```

Figure 110.JCL example to vary VTSS VTSS01 online

VTVMAINT

`VTVMAINT` does the following:

- Unlinks VTVs from MVCs,
- Sets the VTV Management Class, and
- Logically dismounts specified VTVs in an offline VTSS.

Syntax

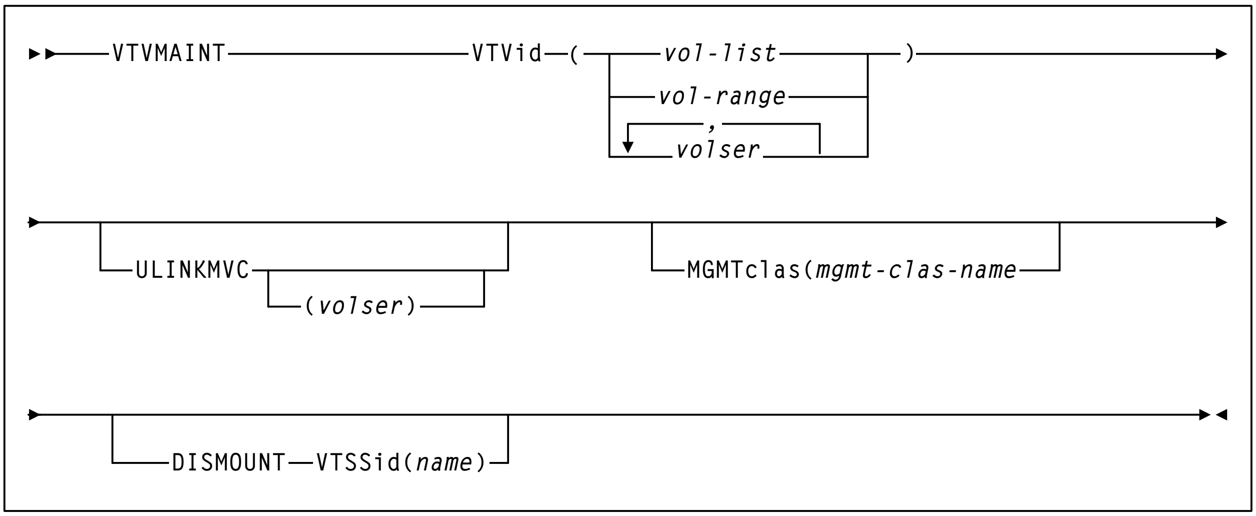


Figure 111. `VTVMAINT` ***syntax***

Parameters

`VTVid`
: specifies the VTVs.

  `volser`, `vol-range` or `vol-list`
  : the volsers of one or more VTVs.

`ULINKMVC`
: unlink the specified VTVs.

  `volser`
  : unlink the specified VTVs from the specified MVC.

`MGMTclas`
: set the Management Class of the VTVs.

  `mgmt-class-name`
  : the Management Class name that you specified on the `MGMTclas` control statement. For more information, see “MGMTCLAS Control Statement” on page 204.

`DISMOUNT VTSSid`

logically dismount the specified VTVs in the specified VTSS.

`name`

the VTSS name.

Interfaces

`SWSADMIN` utility only.

Usage

Use `VTVMAINT` to do the following:

- Unlink VTVs from MVCs,
- Set the VTV Management Class, and
- Logically dismount specified VTVs in an offline VTSS.

Note: Running `VTVMAINT` also produces an MVC report of the volumes affected by the `VTVMAINT` job.

Changing VTV Management Class and Unlinking VTVs from MVCs

You can use `VTVMAINT` to change a VTV's Management Class. If the new Management Class specifies a different Storage Class, the VTV's current location on 1 or 2 MVCs is incorrect. The following procedure tells how to use `VTVMAINT` to change a VTV's Management Class and Storage Class.

To change a VTV's Management Class and Storage Class:

1. **Recall the VTV.**

   The VTV must be VTSS-resident for the unlink to succeed in Step 2.

2. **Use `VTVMAINT ULINKMVC` to unlink the VTV from the MVC(s) where it is located.**
3. **Use `VTVMAINT MGMTclas` to assign a new management class.**
4. **Remigrate the VTV to place it on the correct MVCs.**

Logically Dismounting VTVs in an Offline VTSS

If a VTV is mounted when a VTSS goes offline and a copy of the VTV exists on an MVC, VTCS will not recall the migrated VTV to an alternate VTSS because the VTV is in mounted status on the offline VTSS. In this situation, you can use the `VTVMAINT` to logically dismount VTVs in the offline VTSS (turn off the “mounted” bit in the CDS), then recall the VTV to an alternate VTSS. VTCS records each successful VTV dismount in the `SMF14STA` field of the SMF Subtype 14 record. For more information, see “SLSSMF14 - VTCS SMF Subtype 14 Record” on page 266. The VTVRPT (UNAVAIL) option reports the status of unavailable VTVs in an offline VTSS. For more information, see “VTVRPT” on page 173.

Warning: Don’t dismount an unavailable VTV in an offline VTSS unless you are absolutely sure that the MVC copies, if any, of the VTV, are identical in content to the unavailable VTV! Otherwise, you risk recalling a VTV with back-level data to an alternate VTSS! For example, a VTV mounted for read is probably safe to dismount for recall to an alternate VTSS. A VTV mounted for write, however, is probably not safe to dismount because it has probably been updated and the MVC copies are therefore back-level.

The following procedure provides the general steps you use to logically dismount a VTV and access that VTV from a different VTSS.

To logically dismount a VTV and access that VTV from a different VTSS:

1. **Vary the VTSS offline to VTCS with the following command:**

   ```
   VT VARY VTSS(name) OFFLINE
   ```

   If I/O was active and the VTSS failed, MSP should box the VTDs and dismount any mounted VTVs *from the MSP perspective*. However, if communication with the VTSS failed before the VTSS actually dismounted any mounted VTVs, they may still be online to VTCS. Therefore, you first need to vary the VTSS offline to VTCS.

   If MSP boxed the VTDs and dismounted any mounted VTVs, go to Step 3. Otherwise, continue with Step 2.

2. **Dismount the VTV (MSP perspective).**

   You cannot remount the VTV on a VTD in another VTSS if MSP still considers it mounted in the offline VTSS. Do either of the following:

   - Use the `MSP UNLOAD` command to dismount the VTV.
   - Use the `VARY OFFLINE` to vary offline the VTD where the VTV is mounted, which will also dismount the VTV.

3. **Run** `VTVMAINT`**, specifying the offline VTSS and VTV(s) you want to logically dismount.**

   For example, to logically dismount VTVs VV6823, VV6825, and VV6688 in offline VTSS01, code the following `SLSIN DD` statement in your JCL:

   `VTVMAINT DISMOUNT VTV(VV6823,VV6825,VV6688) VTSS(VTSS01)`

   If migrated copies of the dismounted VTVs exist that an online VTSS can access, you can now use this VTSS to access the VTVs.

Caution: If the VTV copy mounted in the offline VTSS was modified and not migrated, the MVC copy that you recall to an alternate VTSS is not current! Therefore, StorageTek strongly recommends that you do not recall these non-current MVC copies!

Hint: When the offline VTSS is ready to be brought back online, StorageTek strongly recommends that you audit the VTSS before running production jobs that use the VTSS. Also ensure that you clear any boxed VTD conditions before issuing the `VTSS VARY ONLINE` command.

JCL Requirements

`STEPLIB`
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

`SLSPRINT`
: specifies the destination for the `VTVMAINT` report.

`SLSIN`
: specifies the input to the `SWSADMIN` program (`VTVMAINT` utility name and parameters).

JCL Examples

Figure 75 shows example JCL to run the `VTVMAINT` utility.

```
//VTVMAINT  EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 VTVMAINT DISMOUNT VTV(VV6823,VV6825,VV6688) VTSS(VTSS01)
```

Figure 112. `VTVMAINT` ***utility example***

VTVMAINT Report

Figure 88 shows an example of a `VTVMAINT` report for the following command:

```
VTVMAINT VTV(X00000-X00002) ULINKMVC MGMTCLAS(M1)
```

```
SWSADMIN (5.1.0)                 STORAGETEK VTCS SYTEM UTILITY                                          PAGE 0001
TIME 06:32:03                         VTV MAINTENANCE                                                   DATE 2003-04-19
VTV             RC
X00000          04
X00001          04
X00002          04
VTVMAINT EXCEPTION REPORT
VTV X00000 IS ALREADY IN MGMTCLAS M1
VTV X00001 IS ALREADY IN MGMTCLAS M1
VTV X00002 IS ALREADY IN MGMTCLAS M1
SLS1315I SWS500.V5.CDS WAS SELECTED AS THE PRIMARY CONTROL DATA SET
SWSADMIN (5.1.0)                 STORAGETEK VTCS SYTEM UTILITY                                          PAGE 0002
TIME 06:32:03                         VTCS VTV REPORT                                                   DATE 2003-04-19
VTV        SIZE  COMP% <----CREATION-----> <----LAST USED----> MIGR SCRT RESD REPL MGMT  MVC1   MVC2   MVC3   MVC4   MAX    VTSSNAME
VOLSER     (MB)              DATE     TIME      DATE     TIME                      CLASS                              VTV
X00000     0.01      0  2003MAY19  05:02:08  2003MAY19  05:22:08  -    -   R    -    M1    022550 022551 022552 022553 800
X00001     0.01      0  2003MAY19  05:02:08  2003MAY19  05:22:08  -    -   R    -    M1    033550 033551 033552 033553 800
X00002     0.01      0  2003MAY19  05:02:08  2003MAY19  05:22:08  -    -   R    -    M1    044550 044551 044552 044553 800
       3 INITIALIZED VTVS PROCESSED
       0 NON-INITIALIZED VTVS PROCESSED
```

Figure 113.VTVMAINT Report

As shown in Figure 113, the `VTVMAINT` report shows:

- Status of VTVs processed - volser and return code (0 - all updates completed, 4 - some updates completed, 8 - no updates completed).
- An exception report of the reason for all uncompleted updates.

A VTV report; for more information, see “VTV Report” on page 177.

VTVRPT

The VTVRPT reports the status of your VSM system's VTVs.

Syntax

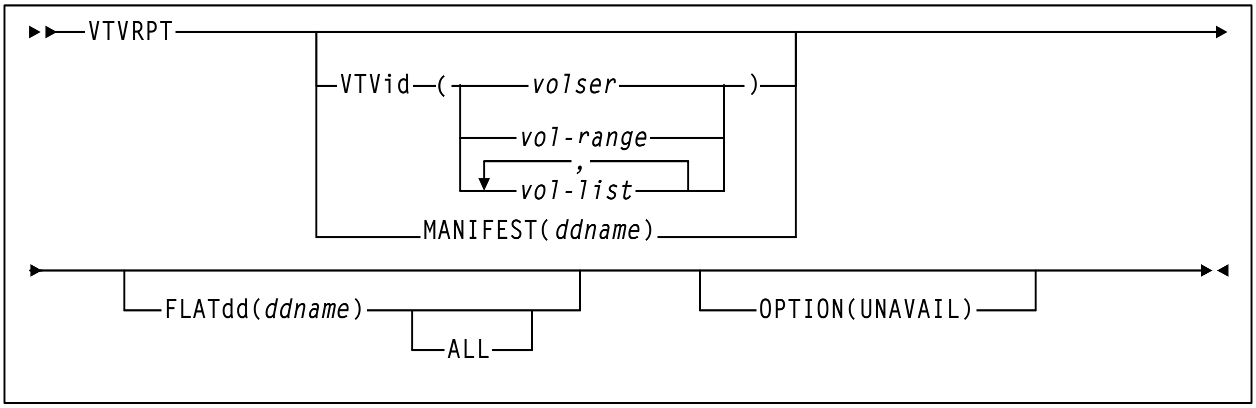


Figure 114. VTVRPT ***syntax***

Parameters

VTVid

specifies the VTVs for the report. If you do not specify the VTVs, the report includes all initialized VTVs in your VSM system. A VTV is initialized when VTCS has used it at least once.

volser, vol-range, or vol-list

the volsers of one or more VTVs.

MANIFEST

specifies the input ddname of the manifest file used to generate the report.

ddname

ddname of the manifest file. **Note that** you can specify a merged manifest file or multiple manifest files. Figure 59 on page 93 shows example JCL to run the MVCMAINT utility where the MANIFEST parameter specifies importing multiple generations of a file.

FLATdd

specifies the output destination ddname if a flat file is required.

ddname

the ddname of the flat file included in the JCL.

ALL

specifies to report on all VTVs (including non-initialized volumes). If you do not specify ALL, the flat file reports only initialized VTVs.

OPTION(UNAVAIL)

specifies to report only on unavailable VTVs (VTVs in an offline VTSS).

Interfaces

SWSADMIN utility only.

Usage

Use the VTVRPT to report the status of your VSM system's VTVs. The report includes all VTVs that contain current data and all scratched VTVs. For more information, see "VTV Report" on page 177. Figure 115 on page 174 shows example JCL to run VTVRPT to produce a report and Figure 120 on page 177 shows the report format. VTV reports only list VTVs that have been used, not VTVs that are defined but have not been not used. Note that you can specify the MANIFEST parameter to generate a report from an export manifest file instead of from the HSC CDS.

You can also report on only unavailable VTVs (VTVs in offline VTSSs). For more information, see Figure 117. on page 175 and "VTV Report" on page 177.

To produce reports for VSM, ExPR requires a flat file format of the VTV report as input. Figure 117 on page 175 shows example JCL to run VTVRPT to produce flat file format of the report for ExPR. Table 10 on page 181Table 10 shows the record format of the flat file produced by VTVRPT. shows the flat file record format.

You can produce VTV reports in structured XML format. You can then process the XML output in a programming language of your choice. For information on the VTCS sample REXX execs, see "Sample REXX Execs" on page 313.

JCL Requirements

The following are the required and optional statements for the VTVRPT JCL:

STEPLIB
: specifies the link library (SLSLINK) that contains the VTCS and HSC modules.

SLSPRINT
: specifies the destination for the VTV report.

SLSIN
: specifies the input to the SWSADMIN program (VTVRPT utility name and parameters).

SLSXML
: specifies the output destination for XML output. Allocate this file as RECFM=VB, LRECL=255.

JCL Examples

Figure 115 shows example JCL to run VTVRPT to produce a report of all VTVs in your VSM system.

```
//VTVR   EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB    DD DSN=hlq.SLSLINK,DISP=SHR
//SLSPRINT   DD SYSOUT=*
//SLSIN      DD *
  VTVRPT
```

Figure 115. Example JCL for the VTVRPT utility

Figure 116 shows example JCL to run VTVRPT to produce a report using manifest file REMOTE1 as input.

```
//VTVR   EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB    DD DSN=hlq.SLSLINK,DISP=SHR
//REMOTE1      DD DSN=FEDB.VSMLMULT.REMOTE1,DISP=SHR
//SLSPRINT   DD SYSOUT=*
//SLSIN      DD *
  VTVRPT MANIFEST(REMOTE1)
```

Figure 116. Example **JCL** ***for the*** VTVRPT ***utility (manifest file input)***

Figure 117 shows example to run VTVRPT to produce a flat file report format of all VTVs for input to ExPR.

```
//VTVR   EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//VTVOUT       DD DSN=FEDB.FLAT,UNIT=SYSALLDA,DISP=OLD
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 VTVRPT FLATDD(VTVOUT)
```

Figure 117. Example **JCL** ***for the*** VTVRPT ***utility (flat file output for ExPR)***

Figure 118 shows example to run VTVRPT to report only on unavailable VTVs (VTVs in an offline VTSS).

```
//VTVR   EXEC PGM=SWSADMIN,PARM='MIXED'
//STEPLIB      DD DSN=hlq.SLSLINK,DISP=SHR
//VTVOUT       DD DSN=FEDB.FLAT,UNIT=SYSALLDA,DISP=OLD
//SLSPRINT     DD SYSOUT=*
//SLSIN        DD *
 VTVRPT OPTION(UNAVAIL)
```

Figure 118. Example **JCL** ***for the*** VTVRPT ***utility (unavailable VTVs only)***

Figure 119 shows example to run VTVRPT to report only on unavailable VTVs (VTVs in an offline VTSS) in structured XML format.

```
//VTVREPT   JOB  x,CHRIS,CLASS=A,MSGCLASS=E
//*
//*
//*
//VTVRPT     EXEC PGM=SWSADMIN
//STEPLIB     DD DSN=ncs610.LINKLIB,DISP=SHR
//SLSCNTL     DD DSN=hlq.V61.CLUSTER.CDS,DISP=SHR
//SLSPRINT    DD SYSOUT=*
//SYSOUT      DD SYSOUT=*
//SLSXML      DD DISP=(,CATLG),DSN=hlq.VTVREPT.XML,
//            DCB=(RECFM=VB,LRECL=255,BLKSIZE=4096),
//            UNIT=SYSDA,SPACE=(CYL,(x,y))
//SLSIN       DD *
 VTVRPT
/*                                  //*

//TSO         EXEC PGM=IKJEFT01,DYNAMNBR=128
//SYSPROC     DD DSN=ncs610.SLSSAMP,DISP=SHR
//SYSTSPRT    DD SYSOUT=*
//SLSPRINT  DD SYSOUT=*

//INFILE      DD  DISP=SHR,DSN=hlq.VTVREPT.XML
//OUTFILE     DD  DISP=(,CATLG),DSN=hlq.VTVREPT.CSV,
//            DCB=(RECFM=VB,LRECL=1024,BLKSIZE=4096),
//          UNIT=SYSDA,SPACE=(CYL,(x,y))
//SYSTSIN     DD *
 %VTVXML
/*
//
```

Figure 119. ***Example* JCL *for the* VTVRPT *utility (unavailable VTVs only, structured XML format)***

Note: XML output can consume **considerable** space. You may want to consider routing your XML output to a VTV so that the output is compressed.

VTV Report

Figure 120 shows an example of a VTV report and Figure 121. on page 178 shows an example of a VTV report with the UNAVAIL option. This report lists only unavailable VTVs in three sections--unavailable mounted on a VTD, unavailable VTSS-resident, and unavailable VTSS-resident and fenced.

```
SWSADMIN (5.1.0)                STORAGETEK VTCS SYTEM UTILITY                                          PAGE 0002

TIME 06:32:03                        VTCS VTV REPORT                                                   DATE 2003-03-20

VTV        SIZE  COMP% <----CREATION-----> <----LAST USED----> MIGR SCRT RESD REPL MGMT  MVC1   MVC2   MVC3   MVC4   MAX    VTSSNAME

VOLSER     (MB)             DATE      TIME       DATE      TIME                        CLASS                              VTV

X00T00     0.04     84  2003JUL16  05:02:08  2003JUL19  05:41:00  M    -    R    -    VCL4  022550 022551 022552 022553 800    VTSS16

X00002  <MOUNT>         2003JUL14  06:54:35  2003JUL19  07:43:46  M    -    R    -    VCL4  033550 033551 033552 033553 800    VTSS17

X00003    15.60     84  2003JUL14  10:05:05  2003JUL19  05:41:28  M    -    R    -    VCL2  044550 044551               400    VTSS16

X00004     0.36     84  2003MAY28  08:51:20  2003JUL19  05:41:30  M    S    R    -    VCL3  022550 022551 022552        800    VTSS16

X00005    15.60     84  2003JUL14  10:05:14  2003JUL19  05:41:31  M    -    R    -    VCL1  033550                      400    VTSS16

X00006    15.60     84  2003JUL14  10:08:23  2003JUL19  08:45:31  C    -    -    -    VCL4  044550 044551 044552 044553 800    VTSS17
```

Figure 120. **Example output from** VTVRPT

```
SWSADMIN (5.1.0)                 STORAGETEK VTCS SYTEM UTILITY                                         PAGE 002
TIME 06:59:03                     UNAVAIL MOUNTED VTV REPORT                                           DATE 2003-03-20
VTV        SIZE  COMP% <----CREATION-----> <----LAST USED----> MIGR SCRT RESD REPL MGMT  MVC1   MVC2   MVC3   MVC4   MAX    VTSSNAME
VOLSER     (MB)              DATE     TIME      DATE     TIME                      CLASS                               VTV
Y09053  <MOUNT>        2003MAR19  09:34:14  2003MAR20  05:55:44   -    -    R    -    M9                               800    HBVTSS1

SWSADMIN (5.1.0)                 STORAGETEK VTCS SYTEM UTILITY                                         PAGE 003
TIME 06:59:03                     UNAVAIL RESIDENT VTV REPORT                                          DATE 2003-03-20
VTV        SIZE  COMP% <----CREATION-----> <----LAST USED----> MIGR SCRT RESD REPL MGMT  MVC1   MVC2   MVC3   MVC4   MAX    VTSSNAME
VOLSER     (MB)              DATE     TIME      DATE     TIME                      CLASS                               VTV
X01007   156.24     89  2003JAN10  03:00:02  2003MAR01  04:51:47   -    S    R    -                                           HBVTSS1
X01010     3.90      0  2003MAR01  09:10:37  2003MAR01  09:10:37   -    -    R    -                                           HBVTSS1
X01014     3.90      0  2003MAR01  09:11:08  2003MAR01  09:11:08   -    -    R    -                                           HBVTSS1
X01021     3.90      0  2003MAR01  09:21:11  2003MAR01  09:21:11   -    -    R    -                                           HBVTSS1

SWSADMIN (5.1.0)                 STORAGETEK VTCS SYTEM UTILITY                                         PAGE 004
TIME 06:59:03                     UNAVAIL FENCED VTV REPORT                                            DATE 2003-03-20
VTV        SIZE  COMP% <----CREATION-----> <----LAST USED----> MIGR SCRT RESD REPL MGMT  MVC1   MVC2   MVC3   MVC4   MAX    VTSSNAME
VOLSER     (MB)              DATE     TIME      DATE     TIME                      CLASS                               VTV
X01280  <FENCED>                                                   -    -    -    -
X04762  <FENCED>                                                   -    -    -    -
X04776  <FENCED>                                                   -    -    -    -
X02019  <FENCED>                                                   -    -    -    -
X10066  <FENCED>                                                   -    -    -    -
X10068  <FENCED>                                                   -    -    -    -
```

Figure 121. ***Example output from*** `VTVRPT` ***(UNAVAIL option)***

VTVRPT Report Fields

The following list describes the VTV report fields.

VTV Volser
the VTV volser.

Size (MB)
the uncompressed size of the VTV (MB). **<MOUNT>** indicates that the VTV was mounted when the report ran. **<FENCED>** indicates that the VTV's state is unknown. If **<FENCED>** appears, contact StorageTek software support.

Comp %
the VTV compression percentage achieved. This is the difference between the uncompressed and compressed VTV size expressed as a percentage of the uncompressed VTV size. For example if a 100MB VTV compresses to 40MB then the compression% will be given as 60%. A compression of 0% indicates that no compression was possible on the VTV.

Creation Date and Time
the date and time that the VTV was created.

Last Used Date and Time
the date and time that the VTV was last used. This date and time value is updated by successful completion of a VTV mount, migrate, recall, or scratch.

Migr
indicates whether the VTV has been migrated (M) or consolidated (C). If the VTV is both migrated and consolidated, a 'C' appears in this field. If the VTV has not been migrated, it is either VTSS resident or non-existent (not created or used, scratched, and deleted).

Scrt
indicates whether the VTV has been scratched.

Resd
indicates whether the VTV is resident in a VTSS.

Repl

one of the following VTV replication statuses:

-

the VTV has no replication requirements or the VTV has replication requirements but is not resident.

R

replication is required but has not started.

S

replication has started.

C

replication has completed.

Note: The **Repl** column displays R, S, or C only if the VTV is resident, otherwise a '-' appears.

MGMT Class

the name of the Management Class for the VTV specified.

MVC1, MVC2, MVC3, MVC4

the MVC(s) that contain the VTV (for both migration and consolidation). If all of these fields are empty, the VTV has not been migrated or consolidated. If 1 or more of these fields list an MVC volser, the VTV was migrated to each of these MVCs.

MAX VTV

the maximum size of the VTV in Mb (400 or 800).

VTSSNAME

the VTSS where the VTV resides, or, if the VTV is migrated, the VTSS where the VTSS was last resident. If this field is empty, the VTV is non-existent (not created or used, scratched, and deleted).

VTV Flat File Record Format

Table 10 shows the record format of the flat file produced by `VTVRPT`.

Table 10. VTVRPT flat file record format

Decimal Offset	Hexadecimal Offset	Type	Length	Description
0	0	start of record		start of VTV flat file record
0	0	integer	4	record length
4	4	character	1	character set type of text fields
		X’61’		ASCII
		X’6E’		EBCDIC
5	5	character	1	record type ’V’ (indicates VTV report)
6	6	character	6	VTV volser
12	C	character	8	VTSS where the VTV resides
20	14	integer	4	uncompressed VTV size (MB)
24	18	character	1	VTV migrated? (Y, N, or C)
25	19		1	reserved
26	1A	character	6	MVC volser (first copy)
32	20	character	6	MVC volser (second copy)
38	26	character	1	always set to N
39	27	character	1	VTV scratched? (Y or N)
40	28	time_t	4	date VTV created (time_t format)
44	2C	time_t	4	date VTV last referenced (time_t format)
48	30		8	reserved
56	38	integer	2	compression percentage for VTV
58	3A	character	1	replicate indicator: C, R, S, or - For more information, see “VTV Report” on page 177.
59	3B	character	1	fenced indicator (Y or N)
60	3C	character	1	mounted indicator (Y or N)
61	3D	character	8	Management Class name
69	45	character	6	MVC volser (third copy)
75	4B	character	6	MVC volser (fourth copy)
81	51			reserved
82	52	integer	2	maximum VTV size (400 or 800)

Chapter 2. SMC Enhancements and Additions for VSM

Note: For NCS 6.1:

- The TAPEREQ statement (and the accompanying TREQDEF command) has been moved from HSC and MVS/CSC to SMC. For more information, see "TAPEREQ Control Statement" on page 184 and *SMC Configuration and Administration Guide*.
- The UNITATTR statement has been moved from HSC to SMC and is **no longer required** for VTDs. An SMC UNITATTR statement is required **only** to set the real transport model type for non-library transports (which are not supported for VSM). For more information, see *SMC Configuration and Administration Guide*.
- The HSC ALLOC command has been moved from HSC to SMC and renamed the ALLOCDEF command. For more information, see "SMC ALLOCDEF Command Enhancements" on page 190 and *SMC Configuration and Administration Guide*.

TAPEREQ Control Statement

The TAPEREQ control statement can route tape data sets to VSM and pass a Management Class to VSM.

Syntax

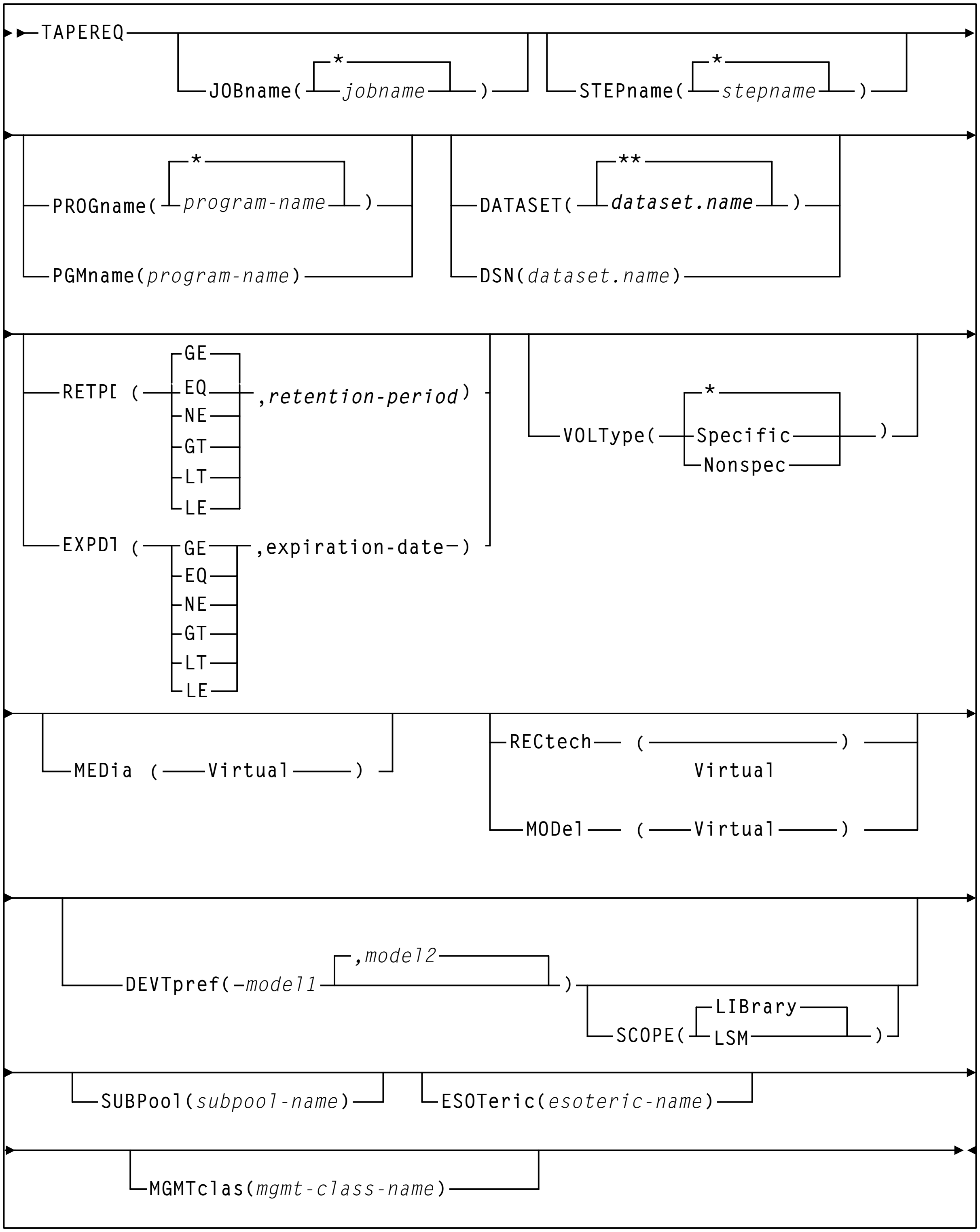


Figure 122. TAPEREQ ***Control Statement Syntax***

Parameters

Unchanged TAPEREQ Parameters

The following `TAPEREQ` parameters are unchanged but apply to VSM. Figure 122. on page 185 shows valid values for these parameters; see *SMC Configuration and Administration Guide* for more information.

- `JOBname`
- `STEPname`
- `PROGram`
- `PGMname`
- `DATASET`
- `DSN`
- `RETPD`
- `EXPDT`
- `VOLType`
- `DEVTpref`
- `SCOPE`

New or Enhanced TAPEREQ Parameters

The following `TAPEREQ` parameters apply to both VSM and NCS without VSM installed.

`SUBPool`

specifies the scratch subpool that contains volumes used to satisfy nonspecific requests. For more information about scratch subpool management, see *SMC Configuration and Administration Guide*.

`poolname`

the subpool name.

`ESOTeric`

specifies the esoteric that defines the list of eligible transports to be used to satisfy a tape request.

To route a data set to a VTD, specify one of the esoteric names that you defined during configuration. For more information on creating and using VSM esoterics for `TAPEREQ` statements, see *VTCS Installation and Configuration Guide*.

For more information on using esoteric substitution to route a data set to Nearline transports that are *not* RTDs, see Chapter 9, “User Exits” in *HSC System Programmer’s Guide for MSP.*

`esoteric-name`

the esoteric name.

TAPEREQ Parameters for VSM

The `MEDia`, `RECtech`, and `MODel` parameters have a value of `Virtual` for VSM only. Specifying `Virtual` on any of these three parameters will route the data set to a VTV mounted on a VTD. `Virtual` does not apply to NCS without VSM installed. The `MGMTclas` parameter does not apply to NCS without VSM installed.

Caution: If you specify a Management Class on the `MGMTclas` parameter, you must specify both `MEDia(V)` and `RECtech(V)`.

`MEDia`

specifies the volume media.

`Virtual`

specifies that VSM will route the data set to a VTV mounted on a VTD.

`RECtech`

specifies the recording technique.

`Virtual`

specifies that VSM will route the data set to a VTV mounted on a VTD.

`MODel`

specifies the transport model.

`Virtual`

specifies that VSM will route the data set to a VTV mounted on a VTD.

`MGMTclas`

specifies a Management Class you defined on the `MGMTclas` control statement; for more information, see “MGMTCLAS Control Statement” on page 204.

`mgmt-class-name`

the Management Class name.

Note: NCS does not support the `DUPlex` parameter on `TAPEREQ` statements, only on `MGMTclas` statements. For more information, see “MGMTCLAS Control Statement” on page 204.

Usage

To route data sets to VSM with TAPEREQ statements, do one of the following:

- Specify Virtual on the MEDia, MODel, or RECtech parameter. If you specify Virtual, VSM selects an available VTD in your system and routes the data set to that VTD.

  In a multi-VTSS environment, therefore, specifying Virtual does *not* direct the VTD allocation to a specific VTSS, but lets the allocation occur in any VTSS in the configuration.

- Specify an esoteric that represents VTDs on the ESOTeric parameter.

  For more information on creating and using VSM esoterics for TAPEREQ statements, see *VTCS Installation and Configuration Guide*.

You can also specify a Management Class on the MGMTclas parameter. You define Management Classes with the MGMTclas statement; for more information, see "MGMTCLAS Control Statement" on page 204. You use the MGMTDEF command to load MGMTclas statements from a specified definition data set; for more information, see "MGMTCLAS Control Statement" on page 204.

Note: If you specify a Management Class on a TAPEREQ statement and an SMS routine, the Management Class on the SMS routine takes precedence.

You must use the TREQDEF command to load TAPEREQ control statements from a specified definition data set; see *SMC Configuration and Administration Guide* for more information.

Note: Multiple TAPEREQ statements that specify the same or overlapping selection criteria (such as jobname, stepname, or data set) can cause undesirable results (such as assignment of MEDia Virtual *and* an esoteric). For example:

```
TAPEREQ DSN(AA22*.*) MEDIA(V) RECT(V)
TAPEREQ DSN(**) MEDIA(LONGITUD) RECT(36) ESOT(ACS0)
```

In this case, because the second TAPEREQ statement is a "catchall" statement with a wildcard for *any* data set, SMC will actually attempt to assign real media and esoteric ACS0 to any data set that meets the data set mask of AA22*.* instead of routing it to VSM. Let's say that what you really want is to route all data sets selected by mask of AA22*.* to VSM, but you want to route all data sets selected by a mask of PROD17.* to 36-track tape in ACS0 (represented by esoteric ACS0). To do this, delete the catchall TAPEREQ statement and instead code the following:

```
TAPEREQ DSN(AA22*.*) MEDIA(V) RECT(V)
TAPEREQ DSN(PROD17.*) MEDIA(LONGITUD) RECT(36) ESOT(ACS0)
```

Examples

To route all data sets with the HLQ of PAYROLL to a VTV mounted on a VTD that VSM selects, create the following TAPEREQ statement:

```
TAPEREQ DSN(PAYROLL.**) MED(VIRTUAL) RECT(VIRTUAL)
```

To route all data sets with the HLQ of PAYROLL to a VTV mounted on one of the VTDs represented by the VTSS1 esoteric, create the following TAPEREQ statement:

```
TAPEREQ DSN(PAYROLL.**) ESOT(ESOVTSS1)
```

To route volumes created by jobs with jobname PRODSL2 to a VTV mounted on a VTD that VSM selects and specify Management Class MGMTCLS1:

```
TAPEREQ JOB(PRODSL1) RECT(VIRTUAL) MGMT(MGMTCLS1)
```

SMC ALLOCDEF Command Enhancements

As described in *SMC Configuration and Administration Guide*, the `ALLOCDEF` command sets or changes NCS device allocation options.

In the JES environment, regardless of the value you specify, the `DEFER` parameter is always set to `ON` for VTVs. That is, for VTVs, deferred mount processing is enabled, which overrides the mount processing specified in the user's JCL. The VTV mount is deferred until the JCL job step opens a data set on the VTV. This value helps minimize VTV recalls. If a data set resides on a migrated VTV, VSM does not recall the VTV until the job actually opens the data set on the VTV.

Note that if a unit affinity chain includes a mixture of incompatible drives (including VTDs), NCS SMC device exclusion always ensures that the chain will be broken.

Chapter 3. HSC Enhancements and Additions for VSM

Note: For NCS 6.1:

- The `TAPEREQ` statement (and the accompanying `TREQDEF` command) has been moved from HSC and MVS/CSC to SMC. For more information, see "TAPEREQ Control Statement" on page 184 and *SMC Configuration and Administration Guide*.
- The `UNITATTR` statement has been moved from HSC to SMC and is **no longer required** for VTDs. An SMC `UNITATTR` statement is required **only** to set the real transport model type for non-library transports (which are not supported for VSM). For more information, see *SMC Configuration and Administration Guide*.
- The HSC `ALLOC` command has been moved from HSC to SMC and renamed the `ALLOCDEF` command. For more information, see "SMC ALLOCDEF Command Enhancements" on page 190 and *SMC Configuration and Administration Guide*.

This chapter contains reference information about the following enhancements and additions to HSC to support VSM:

- "DISPLAY Command" on page 193, that accepts the following parameters:
  - `FEATures`, that displays the HSC features set by the `FEATures` PARMLIB control statement.
  - `MGMTDEF`, that displays the data set and date and time loaded if `MGMTclas` control statements are active.
  - `MVCDEF`, that displays the data set and date and time loaded if `MVCPool` control statements are active.
- "FEATURES Control Statement" on page 194, that specifies which VSM features are enabled.
- "MERGECDS Utility" on page 195, that can update a CDS or merge CDSs with VSM volume records.
- "MGMTCLAS Control Statement" on page 204, an HSC control statement that defines VSM Management Classes.
- "MGMTDEF Command" on page 213, a new HSC command that loads the `MGMTclas` statements from a specified definition data set.
- "MOUNT Command" on page 215, that can mount a specific or scratch VTV on a VTD and optionally assign a Management Class to the VTV.
- "MVCPOOL Control Statement" on page 216, an HSC control statement to define a pool of HSC volumes as MVCs
- "STORCLAS Control Statement" on page 219, an HSC control statement that defines VSM Storage Classes.

- “STORLST Control Statement” on page 224, an HSC control statement that specifies a list of Storage Classes and their corresponding preferencing.
- “STORSEL Control Statement” on page 226, an HSC control statement that defines a Storage Class usage rule that applies to the Storage Class list and its preferencing specified on a referenced `STORLST` control statement.
- “VTSSLST Control Statement” on page 231, an HSC control statement that specifies a list of VTSSs and their corresponding preferencing.
- “VTSSSEL Control Statement” on page 233, an HSC control statement that defines a VTSS usage rule that applies to the VTSS list and its preferencing specified on a referenced `VTSSLST` control statement.
- “VOLATTR Control Statement” on page 229, that specifies VTV attributes, including the volser and media type (virtual).
- “HSC Programmatic Interface Enhancements” on page 237
- “HSC User Exit Enhancements” on page 238
- “HSC Batch API Enhancements” on page 238

Note: The following HSC enhancements are available via PTF as described below:

  - The `HSC MOUNT` command can now mount a scratch or specific VTV on a VTD and optionally assigns a Management Class to the VTV. For more information, see “MOUNT Command” on page 215.
  - The Programmatic Interface `MOUNT` request now supports an additional parameter of `MGMTCLAS` that can assign a VSM Management Class to the VTV. For more information, see “HSC Programmatic Interface Enhancements” on page 237.

- “HSC Operator Command Enhancements” on page 243
- “HSC DELDISP Parameter Enhancements” on page 249

DISPLAY Command

For VSM, the HSC `Display` command displays the data set and date and time loaded if `MGMTclas` and/or `MVCPool` control statements are active and the HSC features set by the `FEATures` PARMLIB control statement.

Syntax

```
►►─Display───────────FEATures───────────MGMTDEF───────────────MVCDEF─────────────►◄
```

Figure 123. `Display` ***Command***

Parameters

`FEATures`
: displays the HSC features set by the `FEATures` PARMLIB control statement.

`MGMTDEF`
: displays the data set and date and time loaded if `MGMTclas` control statements are active.

`MVCDEF`
: displays the data set and date and time loaded if `MVCPool` control statements are active.

Usage

Use the HSC `Display` command to display the data set and date and time loaded if `MGMTclas` and or `MVCPool` control statements are active and the HSC features set by the `FEATures` PARMLIB control statement.

Examples

To display the data set and date and time loaded of an active `MGMTclas` statement, enter:

```
D MGMTDEF
```

To display the data set and date and time loaded of an active `MVCPool` statement, enter:

```
D MVCDEF
```

To display the HSC features set by the `FEATures` PARMLIB control statement, enter:

```
D FEAT
```

FEATURES Control Statement

The HSC `FEATures` control statement specifies which VSM features are enabled.

Syntax

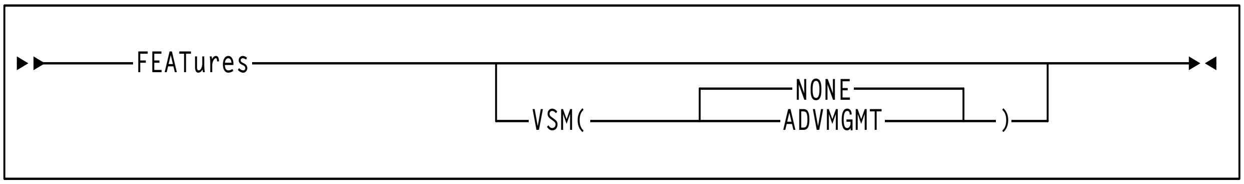


Figure 124. `FEATures` **Control Statement**

Parameters

`VSM`

specifies which VSM Management Features are enabled.

`NONE`

Basic Management only is enabled; the Advanced Management Feature is not enabled (the default). `STORclas` statements, the `MGMTclas` statement `MIGpol`, `RESTIME`, `CONSRC`, `CONTGT` and `REPLICAT` parameters, and `EXPORT` and `IMPORT` are disabled. For more information about the `MGMTclas` parameters that are valid for Basic Management, see "Parameters - Basic Management Feature" on page 205.

`ADVMGMT`

Both Basic and the following Advanced Management Features are enabled:

- `STORclas` statements; for more information, see "STORCLAS Control Statement" on page 219.
- `MGMTclas` statement `MIGpol`, `RESTIME`, `CONSRC`, `CONTGT`, and `REPLICAT` parameters; for more information, see "Additional Parameters - Advanced Management Feature" on page 209.
- `EXPORT` and `IMPORT`; for more information, see "EXPORT" on page 82 and "IMPORT" on page 86.

If the `FEATures` PARMLIB control statement is not specified, Basic Management only is enabled.

Usage

Use the HSC `FEATures` control statement to specify which VSM features are enabled.

Example

To enable the Advanced Management Feature, create the following statement:

```
FEAT VSM(ADVMGMT)
```

MERGECDS Utility

The enhanced `MERGEcds` utility can reconfigure a CDS or merge CDSs with VSM volume records.

Syntax

```
►►──────MERGEcds──┬─────┬───┬──────────────┬───┬──────────────┬──►
                  └─ALL─┘   └───VALIDate───┘   ├───VIRTonly───┤
                                               └───REALonly───┘

►──┬──────────────┬───┬─────────────┬──────────────────────────►◄
   └───DELVirt────┘   └───NOMSG─────┘
```

Figure 125. `MERGEcds` ***Utility Syntax***

Parameters

`VALIDate`

specifies to only validate that the configurations to be reconfigured or merged are compatible, but not do the operation. `MERGEcds VALIDate` reports any duplicate, in-transit, and errant volumes.

`ALL`

specifies to copy volume information for all ACSs and VTSSs from the "from" CDS to the "to" CDS. For a CDS merge, the ACS ID and LSM IDs, and VTSS identifiers must match.

If you do not specify `ALL`, `MERGEcds` reads the parameters specified in the `SLSMERGE DD` statement, which specify the ACSs, LSMs, and VTSSs whose volume information you want to merge or reconfigure. `ALL` and `SLSMERGE DD` are mutually exclusive.

You also specify the `ALL` parameter to convert a CDS to extended format.

`VIRTonly`

specifies to use only VSM volume records.

`REALonly`

specifies to use only real Nearline volume records.

As shown in Figure 125, `VIRTonly` and `REALonly` are mutually exclusive. See Table 11. on page 199 for more information about the `MERGEcds` parameter interactions.

`DELVirt`

specifies that VTV and MVC volume information is not copied to the “to” CDS if **both** of the following are true:

- The VTVs and MVCs defined in the "from" CDS are either uninitialized or empty. An empty VTV is not VTSS resident and has no current MVC copies. An empty MVC contains no current VTVs.
- The uninitialized or empty VTVs and MVCs in the "from" CDS are not defined in the "to" CDS. That is, no duplicate volsers exist.

`NOMSG`

suppresses message `SLS4245I`, which displays the volser of an MVC or VTV that was not copied to the “to” CDS. `NOMSG` has no effect if you do not also specify `DELVirt`.

Usage

Use the enhanced `MERGEcds` utility to update a CDS or merge CDSs with VSM volume records. For more information, see the following sections:

- “MERGEcds Procedure for VSM Environments”
- “MERGEcds Parameter Interactions” on page 199

You also specify the `ALL` parameter to convert a CDS to extended format.

MERGEcds Procedure for VSM Environments

This procedure is for using `MERGEcds` in a VSM environment. For information about using `MERGEcds` in a Nearline environment without VSM installed, see “MERGEcds Utility” in Chapter 5 of *HSC System Programmer's Guide*.

To run MERGEcds in a VSM environment:

1. **Before running `MERGEcds`, do the following:**

   a. Create VTV and MVC reports for each CDS that contains VSM records for comparison after the merge is complete.

   b. Ensure that the “to” CDS contains enough space to hold the VSM volume records from all CDSs to be merged **and** any new VTV and/or MVC ranges to be created. You may need to expand the “to” CDS if it does not contain enough space. If you are deleting VTV and/or MVC ranges, you must create a new “to” CDS, because you cannot to remove existing records from an existing CDS.

   c. Quiesce all real and virtual tape activity on the “from” CDS by shutting down HSCs that reference the “from” CDS to avoid losing active real and/or virtual operations during the merge. The “from” CDS is defined in the JCL for `MERGEcds` as shown in Figure 128 on page 202 and Figure 129 on page 203.

d. You run `MERGEcds` on the MSP system that contains the "to" CDS, which must already be defined. You can run `MERGEcds` without stopping HSC on the MSP system that contains the "to" CDS, but HSC on this system should be running at base level (no tape activity) until `MERGEcds` completes and until you have verified the results of the merge and backed up the "to" CDS.

Note: If you want rename the "to" and "from" CDSs, StorageTek recommends that you shut down HSC on **both** the "to" and "from" systems. See the *HSC System Programmer's Guide* ("Control Statements and Start Procedure" in Chapter 3 and "Renaming Control Data Sets" in Chapter 2).

e. Before running `MERGEcds` in a VSM environment, you must run VTCS `CONFIG` **even if there are no changes** to the VSM configuration to prepare for merging VTCS volume records.

Note: With `CONFIG`, you can only add MVC and VTV volsers, you cannot remove them. You can, however, use the `DELVirt` parameter to not copy MVC or VTV volsers to the "to" CDS; for more information, see "`DELVirt`" on page 196 and "MERGEcds Example: Merging CDSs with Duplicate VTSS Identifiers, Do Not Copy Volume Information for MVCs and VTVs" on page 203.

f. If you also run `MERGEcds` to merge CDSs with Nearline volume information (for example, merging two ACSs where the "from" ACS contains MVCs), you may also need to do some or all of the tasks described in:

- "Define and Select Nearline Volumes" in *VTCS Installation and Configuration Guide*.
- "MERGECDS Utility" in Chapter 3, "Control Statements and Start Procedure" in *HSC System Programmer's Guide for MSP.*
- "Renaming Control Data Sets" in Chapter 2, "Host Software Component Functions" in *HSC System Programmer's Guide for MSP.*

2. **Run `MERGEcds` as follows:**

   - To merge CDSs, both of which contain *different* VSM volume records, specify `MERGE ALL` or `MERGE ALL VIRTonly`. You would typically do this, for example, if the "from" CDS defines VTVs and MVCs for `VTSS01` and `VTSS02` and the "to" CDS defines VTVs and MVCs for `VTSS05` and `VTSS06` and there are no duplicate MVC or VTV volsers; see Figure 127 on page 202.

   - To merge CDSs which have duplicate VTSS identifiers:

     1. Specify `MERGE` or `MERGE VIRTonly`.
     2. Rename the duplicate VTSS identifier by specifying a new identifier on the `SLSMERGE TVTSS` subparameter.

     You would typically do this, for example, if the "from" CDS defines VTVs and MVCs for `VTSS01` and `VTSS02` and the "to" CDS defines VTVs and MVCs for "populated" VTSSs `VTSS01` and `VTSS02` and there are no duplicate MVC or VTV volsers; see Figure 128 on page 202.

Note: Before running `MERGEcds` to rename a VTSS, you must set the identifier to 99999999 at the LOP.

The virtual part of a CDS merge fails if either of the following occurs:

- There are duplicate MVC or VTV volsers or duplicate volsers between MVCs or VTVs and standard Nearline volumes in the "from" and "to" CDS.
- You specify DELVirt and the affected VTVs and MVCs do not meet the conditions described in "`DELVirt`" on page 196.

Note: If you also specify a merge of real volumes, it will complete even if the virtual merge fails.

3. **After running `MERGEcds`, do the following:**

   a. Produce VTV and MVC reports for the "to" CDS and compare with the reports you produced before the merge.

   b. If the "from" CDS contained MVC records or if you deleted one or more MVC ranges, you need to update the `VOLATTR` statements.

   c. Restart HSC on all LPARs where you shut down HSC and return HSC to full service level on all LPARs running at base service level.

MERGEcds Parameter Interactions

Table 11 describes the interactions of the MERGEcds parameters.

Table 11. MERGEcds ***Parameter Interactions***

If you specify...	The SLSMERGE DD file is...	And MERGEcds...
MERGE ALL	not read	uses both real Nearline volume records and VSM volume records but does not allow renaming the VTSS.
MERGE ALL REALonly	not read	uses only real Nearline volume records (current MERGEcds behavior).
MERGE ALL VIRTonly	not read	uses only VSM volume records but does not allow renaming the VTSS.
MERGE	read	uses both real Nearline volume records and VSM volume records and allows renaming the VTSS.
MERGE REALonly	read and MERGEcds honors the FACS/TACS and FLSM/TLSM subparameters.	uses only real Nearline volume records (current MERGEcds behavior).
MERGE VIRTonly	read and MERGEcds honors the FVTSS/TVTSS subparameters.	uses only VSM volume records and allows renaming the VTSS.
MERGE REALonly VIRTonly	not read	operation fails, REALonly and VIRTonly are mutually exclusive.

JCL Requirements

The following are the required and optional statements for the MERGEcds JCL:

SLSFCNTL
: specifies the current primary copy of the “from” HSC CDS.

SLSFCTL2
: specifies the current secondary copy of the “from” HSC CDS. This is only required if HSC has been set up to run with a secondary copy.

SLSFSTBY
: specifies the current standby copy of the “from” HSC CDS. This is also only required if HSC has been set up to run with a standby copy.

SLSIN
: specifies the input to the SLUADMIN program (MERGEcds utility name and parameters).

SLSMERGE
: specifies the "from" and "to" ACSs, LSMs, or VTSSs to use for a merge. This parameter is optional and is mutually exclusive with the MERGEcds ALL parameter.

Syntax

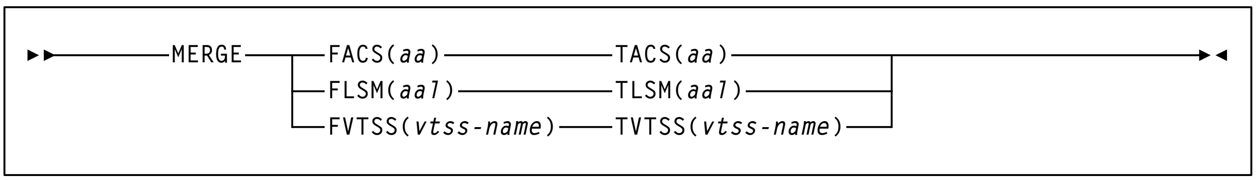


Figure 126. SLSMERGE ***DD Statement Syntax***

ACS and LSM IDs are valid hexadecimal values for Nearline systems. The `vtss-name` is a VTSS identifier.

`FACS=acs-id`
: specifies the “from” ACS.

`TACS=acs-id`
: specifies the “to” ACS.

`FLSM=lsm-id`
: specifies the “from” LSM.

`TLSM=lsm-id`
: specifies the “to” LSM.

`FVTSS=vtss-name`
: specifies the “from” VTSS.

`TVTSS=vtss-name`
: specifies the “to” VTSS.

Caution: Note that the “real” parameters (`FACS`, `TACS`, `FLSM`, `TLSM`) act as selection criteria, while the “virtual” parameters (`FVTSS`, `TVTSS`) do not act as selection criteria, they act only as rename criteria as follows:

- If you specify `FACS/FLSM`, only volume records for real tapes in the specified ACS/LSM are copied to the new CDS.
- If you specify `FVTSS/TVTSS`, all VTV records in the old CDS are copied to the new CDS. For example:

  ```
  MERGE FVTSS(VTSS18) TVTSS(VTSS17)
  ```

  In this example, all VTV records are copied to the new CDS, but the `Resident VTSS` field is changed from `VTSS18` to `VTSS17`.

Also note that you **cannot** specify the same `TOVTSS` statement multiple times in a single `MERGECDS` job. For example, if you want to change the `Resident VTSS` field to `VTSS17` for VTVs where the current field is set to `VTSS18` **and** those where the current field is set to `VTSS19`, you must run two separate `MERGECDS and the corresponding SLICREAT(e)` jobs for each rename operation.

Example:

1. Run `SLICREAT(E)` to build a temporary CDS
2. Run `MERGECDS FVTSS(VTSS18) TVTSS(VTSS17)` using the old CDS as input.
3. Run `SLICREAT(E)` to build a new permanent CDS.
4. Run `MERGECDS FVTSS(VTSS19) TVTSS(VTSS17)` using the temporary CDS from Step 1. as input.

JCL Examples

MERGEcds Example: Reconfiguring a CDS with VSM Volume Information or Merging CDSs with Different VTSS Identifiers

Figure 127 shows example MERGEcds JCL to update a CDS with VSM volume information, such as for an initial VSM configuration. You can also use JCL such as shown in this example to merge CDSs with different VTSS identifiers.

```
//UPDATEVSM   EXEC PGM=SLUADMIN,PARM='MIXED'
//SLSFCNTL    DD DSN=FEDB.VSMLMULT.DBASEPRM,DISP=SHR
//SLSFCTL2    DD DSN=FEDB.VSMLMULT.DBASESEC,DISP=SHR
//SLSFSTBY    DD DSN=FEDB.VSMLMULT.DBASESBY,DISP=SHR
//SLSPRINT    DD   SYSOUT=*
//SLSIN       DD   *
 MERGE ALL VIRT
/*
//
```

Figure 127. MERGEcds ***example: updating a CDS or merging CDSs with different VTSS identifiers***

MERGEcds Example: Merging CDSs with Duplicate VTSS Identifiers

Figure 128 shows example MERGEcds JCL to merge CDS with duplicate VTSS identifiers, VTSS01 and VTSS02, that are renamed to VTSS03 and VTSS04 in the “to” CDS.

```
//MERGEVSM    EXEC PGM=SLUADMIN,PARM='MIXED'
//SLSFCNTL    DD DSN=FEDB.VSMLMULT.DBASEPRM,DISP=SHR
//SLSFCTL2    DD DSN=FEDB.VSMLMULT.DBASESEC,DISP=SHR
//SLSFSTBY    DD DSN=FEDB.VSMLMULT.DBASESBY,DISP=SHR
//SLSPRINT    DD   SYSOUT=*
//SLSIN       DD   *
 MERGE VIRT
//SLSMERGE    DD   *
 MERGE FVTSS(VTSS01) TVTSS(VTSS03)
 MERGE FVTSS(VTSS02) TVTSS(VTSS04)
/*
//
```

Figure 128. MERGEcds ***example: merging CDSs with different VTSS identifiers***

MERGEcds Example: Merging CDSs with Duplicate VTSS Identifiers, Do Not Copy Volume Information for MVCs and VTVs

Figure 129 shows example MERGEcds JCL to merge CDS with duplicate VTSS identifiers, VTSS01 and VTSS02, that are renamed to VTSS03 and VTSS04 in the “to” CDS. This example also specifies the:

- DELVirt parameter so that volsers of uninitialized or empty VTVs and MVCs are not copied to the "to" CDS.
- NOMSG parameter that suppresses message SLS4245I.

```
//MERGEVSM     EXEC PGM=SLUADMIN,PARM='MIXED'
//SLSFCNTL     DD DSN=FEDB.VSMLMULT.DBASEPRM,DISP=SHR
//SLSFCTL2     DD DSN=FEDB.VSMLMULT.DBASESEC,DISP=SHR
//SLSFSTBY     DD DSN=FEDB.VSMLMULT.DBASESBY,DISP=SHR
//SLSPRINT     DD   SYSOUT=*
//SLSIN        DD   *
 MERGE VIRT DELV NOMSG
//SLSMERGE     DD   *
 MERGE FVTSS(VTSS01) TVTSS(VTSS03)
 MERGE FVTSS(VTSS02) TVTSS(VTSS04)
/*
//
```

Figure 129. MERGEcds ***example: merging CDSs with different VTSS identifiers***

MGMTCLAS Control Statement

The `MGMTclas` control statement defines a VSM Management Class. As shown in the following sections, the VSM feature you enable determines which `MGMTclas` parameters are valid; for more information, see “FEATURES Control Statement” on page 194.

Syntax - Basic Management Feature

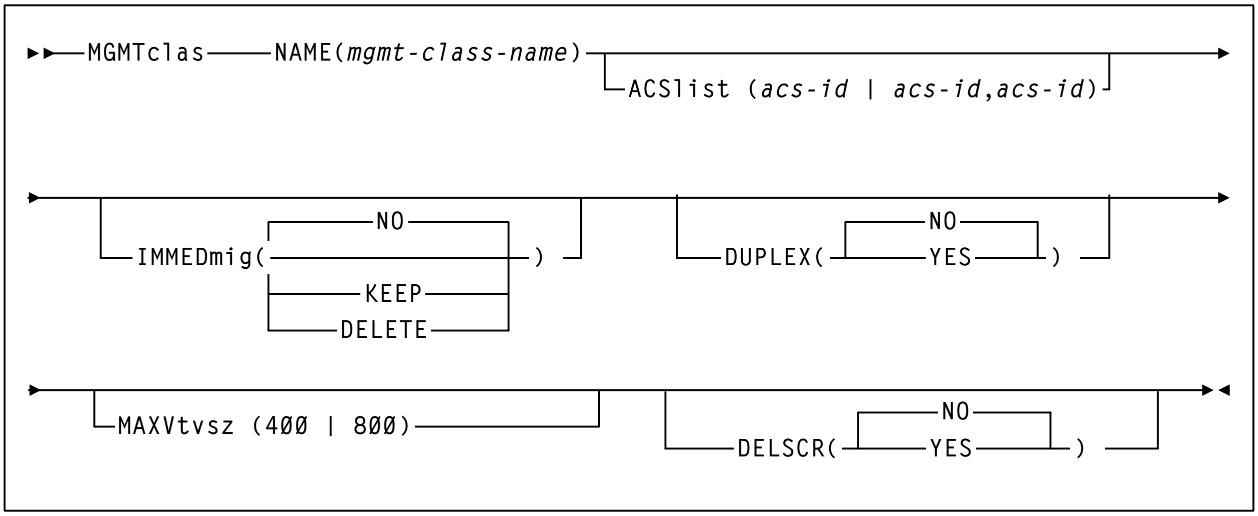


Figure 130. `MGMTclas` ***Control Statement Syntax - Basic Management Feature***

Parameters - Basic Management Feature

NAME

specifies the name of the Management Class.

mgmt-class-name

the Management Class name. This name must be 1 to 8 alphanumeric characters beginning with an alpha character and must follow SMS naming conventions.

ACSlist

specifies the ACSs from which RTDs and MVCs are selected.

ACSlist is optional; if not specified, the default is the ACS specified on the CONFIG DEFLTACS parameter; for more information, see "DEFLTACS=acs-id" on page 21.

The ACSlist and MIGpol parameters are mutually exclusive.

See Table 12. on page 212 for information about using the DUPlex and ACSlist parameters.

Note: Regardless of the number of host to ACS connections in your configuration, VSM supports a maximum of two ACSs connected to each VTSS.

acs-id | acs-id,acs-id

Specify either one or two ACS IDs. An ACS ID has a hexadecimal value from 00 through FF.

IMMEDmig

specifies whether VSM immediately migrates a VTV after dismounting it.

NO

specifies that VSM does not immediately migrate the VTV, but migrates it according to standard VSM migration criteria (the default).

KEEP

specifies that VSM immediately migrates a VTV and keeps a copy resident on the VTSS until the VTV become eligible for deletion.

DELETE

specifies that VSM immediately migrates the VTV and then deletes it from the VTSS.

Note: IMMEDmig KEEP and IMMEDmig DELETE are mutually exclusive with CONFIG HOST NOMIGRAT. If you specify both, the IMMEDmig value overrides NOMIGRAT, and VTCS does not issue a message about this override.

`DUPlex`

specifies whether VSM will migrate two copies of the VTV to two MVCs.

The `DUPlex` and `MIGpol` parameters are mutually exclusive.

See Table 12. on page 212 for information about using the `DUPlex` and `ACSlist` parameters.

`NO`

Do not duplex the VTV (the default).

`YES`

Duplex the VTV.

`MAXVtvsz`

specifies the maximum size for VTVs in this Management Class. `MAXVtvsz` requires you to specify `CDSLEVEL(V6ABOVE) or CDSLEVEL(V61ABOVE)` on the `CONFIG` statement. For more information, see “CONFIG Statement” on page 13.

`400`

400Mb (the default, if not specified).

`800`

800 Mb.

Note:

- The `MAXVtvsz` parameter applies to **only** VSM3s and VSM4s with the following microcode levels:
  - For VSM3s: microcode level N01.00.69.04 **or** microcode level N01.00.71.00 and above
  - For VSM4s: microcode level D01.00.04.03 **or** microcode level D01.00.06.03 and above
- The size of a VTV will only change once it goes through a scratch cycle. Therefore, if you change the Management Class and `DISP=MOD`, then it will still retain the original size.

DELSCR

specifies whether VSM deletes scratched VTVs.

This parameter is optional.

NO

do not delete scratched VTVs (the default).

YES

delete scratched VTVs.

Warning: When you scratch a VTV with DELSCR YES attribute, **VSM erases the VTV data at scratch synchronization time**, which eliminates the ability "unscratch" a VTV to recover data!

Also note that when using HSC to perform scratch synchronization, **it is possible that a volume that is scratch** in the TMC at the beginning of scratch synchronization run and also scratch in the CDS from the previous scratch update run (and thus is in the list for HSC to scratch in the CDS) is accessed by a job during the scratch update run and written to and **made non-scratch** by the TMS in the TMC. **In this case, it is still possible for HSC to scratch the volume** because it was in the originally extracted list of volumes to be scratched. Therefore, **StorageTek strongly recommends** that you **do not** run any jobs that use scratches during HSC scratch synchronization. For more information about HSC scratch synchronization with the Scratch Conversion Utility (SLUCONDB), see Chapter 5, "Utility Functions" of *HSC System Programmer's Guide for MSP*.

For more information about ExLM scratch synchronization with the SYNCVTV function, see "Using ExLM with VTCS (All Versions)" in Chapter 2, "Using ExLM to Manage Nearline and VTCS Resources" of *ExLM System Administrator's Guide.*

Syntax - Advanced Management Feature

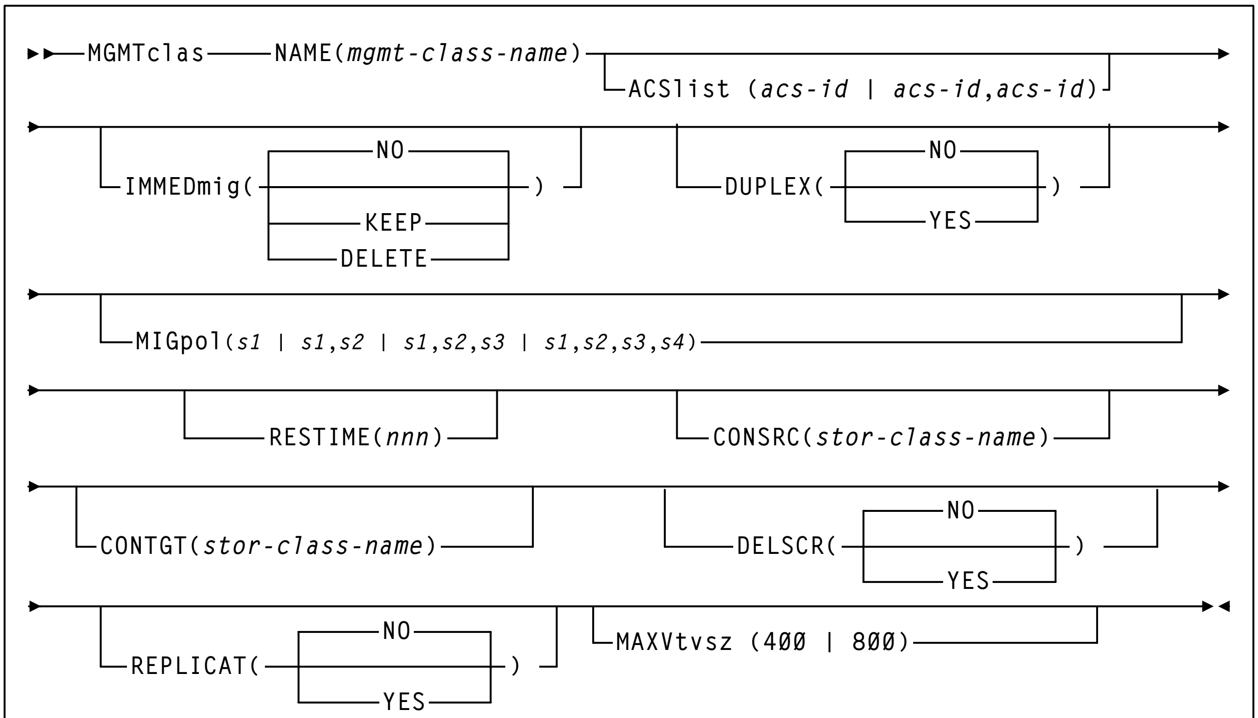


Figure 131. `MGMTclas` ***Control Statement Syntax - Advanced Management Feature***

Additional Parameters - Advanced Management Feature

The following `MGMTclas` parameters are valid for the Advanced Management Feature in addition to the Basic Management Feature parameters described in “Parameters - Basic Management Feature” on page 205.

`MIGpol`

specifies up to four Storage Classes that specify the ACS and media type of migration MVCs. If you specify:

- One Storage Class, VTCS migrates one copy of a VTV.
- Multiple Storage Classes (with different `ACS` values, different `MEDIA` values, or both), VTCS makes multiple copies the VTV to different MVCs in different ACSs.
- Multiple Storage Classes with identical `ACS` and `MEDIA` values, VTCS makes multiple copies of the VTV to the same ACS and media type but to different MVCs.

Note: Multiple Storage Classes on `MIGpol` also affects how:

- VTV recall works.
- MVC space reclamation works.
- How VTV consolidation works

The `DUPlex` and `MIGpol` parameters are mutually exclusive.

This parameter is optional; there is no default value.

`s1 | s1,s2 | s1,s2,s3 | s1,s2,s3,s4`

the names of either up to 4 Storage Classes that you defined on the `STORclas` control statement; for more information, see “STORCLAS Control Statement” on page 219 and “Using the STORclas MEDIA Parameter for MVC Media Preferencing” on page 222. Greater than two copies requires you to specify `CDSLEVEL(V6ABOVE)` or `CDSLEVEL(V6ABOVE)` on the `CONFIG` statement. For more information, see “CONFIG Statement” on page 13.

`RESTIME`

specifies how long VTCS attempts to keep a VTV as VTSS-resident before becoming a preferred automatic migration candidate.

This parameter is optional; there is no default value. Valid values are 1 to 9999.

The `RESTIME` and `IMMEDmig(DELETE)` parameters are mutually exclusive.

`nnnn`

the residency time in hours.

`CONSRC`

specifies the Storage Class that species a preference for the source MVC ACS and media for consolidation of VTVs that are migrated and copied to multiple different MVC locations or media types. If the MVC in the specified Storage Class is unavailable, and the specified Storage Class is not the last (in order specified in the migration policy), VTCS will use the MVC associated with the last Storage Class. If the MVC in the specified Storage Class is unavailable and the specified Storage Class is the last (in order specified in the `MIGpol` parameter), VTCS will use the MVC associated with the previous Storage Class (in order specified in the `MIGpol` parameter).

This parameter is optional; there is no default value.

`stor-class-name`

the name of a Storage Class that you defined on the `STORclas` control statement; for more information, see "STORCLAS Control Statement" on page 219.

`CONTGT`

specifies the Storage Class that determines the output MVC ACS and media for VTV consolidation. Note that the media preferencing is in the opposite order of the list of media types specified on the Storage Class.

This parameter is optional; there is no default value. If you do not specify a value for `CONTGT`, VTCS selects the output MVC as follows:

- For single-ACS and dual-ACS configurations, the media selection order for VTV consolidation.
- For dual-ACS systems, VTCS selects MVCs from the default ACS specified by the `CONFIG DEFLTACS` parameter; for more information, see "`DEFLTACS=acs-id`" on page 21.

`stor-class-name`

the name of a Storage Class that you defined on the `STORclas` control statement; for more information, see "STORCLAS Control Statement" on page 219.

`REPLICAT`

specifies whether VSM replicates the VTV.

`NO`

Do not replicate the VTV (the default).

`YES`

Replicate the VTV.

Usage

Use the `MGMTclas` control statement to specify the name of a VSM Management Class and define policies for that class.

You can also specify a Storage Class on the `MIGpol`, `CONSRC`, and `CONTGT` parameters of the `MGMTclas` control statement. The `MIGpol`, `RESTIME`, `CONSRC`, `CONTGT`, and `REPLICAT` parameters are only valid if `FEATures VSM(ADVMGMT)` is specified; for more information, see "FEATURES Control Statement" on page 194.

You use the `MGMTDEF` command to load `MGMTclas` and `STORclas` control statements, which must reside in the same data set for cross-validation; for more information, see "MGMTDEF Command" on page 213.

Storage Classes are never mixed on the same MVC. If you do not specify a Storage Class on the `MGMTclas` statement, the Storage Class defaults to the VTSS name.

For more information, see:"STORCLAS Control Statement" on page 219.

☞ If you specify the `ACSlist` parameter of the `MGMTclas` statement and the `CONFIG VTSS DEFLTACS` parameter, VTCS ignores the value on the `DEFLTACS` parameter.

Using the DUPlex parameter

Table 12 describes possible scenarios using the `DUPlex` and `ACSlist` parameters.

Table 12. ***MGMTclas ACSlist/DUPlex Scenarios***

If DUPlex is set to...	And ACSlist specifies...	Then VSM...
YES	two ACSs	migrates the VTVs to two MVCs, one in each ACS. (This scenario is the normal one for duplexing to two ACSs.)
YES	one ACS	migrates the VTVs to two MVCs in the ACS specified
NO	two ACSs	ignores the DUPlex policy and migrates the VTVs to two MVCs, one in each ACS.
NO	one ACS	migrates the VTVs to one MVC in the ACS specified

Examples

In the following example, Management Class `MGMTCLS1` specifies VTV duplexing to two MVCs, one in ACS 00 and one in ACS 01 with an immediate migration of the VTV on dismount and a copy of the VTV kept in the VTSS.

```
MGMTclas NAME(MGMTCLS1) ACSL(00,01) IMMED(KEEP) DUP(YES)
```

In the following example, Management Class `MGMT4CP` specifies VTV duplexing to four MVCs with an immediate migration of the VTV on dismount and a copy of the VTV kept in the VTSS.

```
MGMTclas NAME(MGMT4CP) MIGPOL(LOCAL1,LOCAL2,REM1,REM2) IMMED(KEEP)
```

MGMTDEF Command

The MGMTDEF command loads the MGMTclas, STORclas, VTSSLST, VTSSSEL, STORLST, and STORSEL statements from a specified definition data set.

Syntax

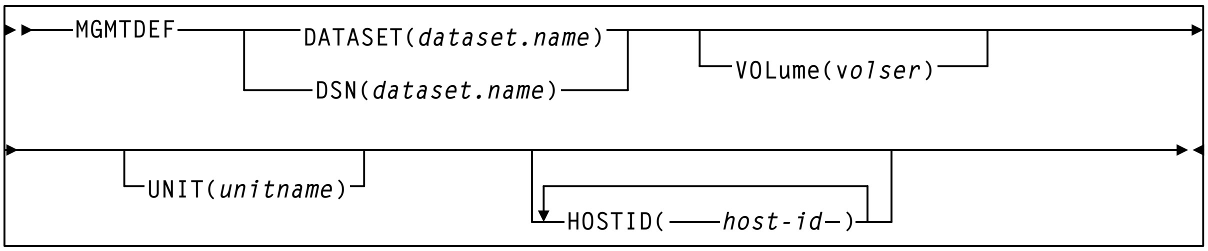


Figure 132. MGMTDEF ***Command***

Parameters

DATASET or DSN

specifies the definition data set that contains the MGMTclas and STORclas statements to load.

dataset.name

the data set name.

VOLume

specifies the DASD volume where the definition data set resides. This parameter is optional, unless the data set is not cataloged, or the data set resides on a volume other than the volume indicated by the catalog.

volser

the DASD volser.

UNIT

specifies the DASD device where the definition data set resides.

unitname

the DASD unit name. If the definition data set is not cataloged and this parameter is omitted, the unit name defaults to SYSALLDA.

HOSTID

specifies the host for execution of the MGMTDEF command. This parameter is only valid when MGMTDEF is specified as a PARMLIB control statement.

host-id

specifies the name of one or more hosts from which to execute the MGMTDEF command. Multiple hosts must be separated by commas.

Usage

Use the `MGMTDEF` command to load `MGMTclas`, `STORclas`, `VTSSLST`, `VTSSSEL`, `STORLST`, and `STORSEL` statements, which must reside in the same data set for cross-validation.

For more information, see

- "MGMTCLAS Control Statement" on page 204
- "STORCLAS Control Statement" on page 219
- "STORLST Control Statement" on page 224
- "STORSEL Control Statement" on page 226
- "VTSSLST Control Statement" on page 231
- "VTSSSEL Control Statement" on page 233

The `MGMTDEF` command is valid for base and full service levels of HSC. You can enter `MGMTDEF` as an operator command or specify `MGMTDEF` as a statement in the HSC PARMLIB. If you specify a `MGMTDEF` statement in the PARMLIB, HSC startup loads the specified `MGMTDEF` statements. Note that the `HOSTID` parameter is only valid when you specify `MGMTDEF` as a statement in the HSC PARMLIB. After HSC startup, you can enter a `MGMTDEF` command to dynamically reload another set of `MGMTclas` statements from a different definition data set. If you restart HSC, it reloads the `MGMTclas` statements specified by the `MGMTDEF` command in the PARMLIB.

Examples

To load the `MGMTclas` and `STORclas` statements in data set `hsc.parms` on a host, enter:

```
MGMTDEF DSN(hsc.parms)
```

MOUNT Command

The enhanced MOUNT command mounts a scratch or specific VTV on a VTD and optionally assigns a Management Class to the VTV.

Syntax

```
►►─Mount──┬─volser──┬──devaddr──┬──────────────────────────────────┬─►◄
          └─SCRTCH──┘           └─MGMTclas(──mgmt-clas-name──)─────┘
```

Parameters

volser | SCRTCH
: specifies a specific VTV volser or the scratch VTV attribute (SCRTCH).

  volser
  : the volser of a specific VTV.

devaddr
: specifies the MSP device address of the VTD to use to mount the VTV.

MGMTclas
: specifies a Management Class you defined on the MGMTclas control statement; for more information, see "MGMTCLAS Control Statement" on page 204.

  mgmt-class-name
  : the Management Class name.

Usage

Use the enhanced MOUNT command to mount a scratch or specific VTV on a VTD and optionally assign a Management Class to the VTV.

Note: For a non-scratch VTV, the MOUNT command assigns a Management Class **only** if:

- The CONFIG GLOBAL statement specifies VTVattr=ALLmount as described in "GLOBAL Statement" on page 15, and
- The VTV is either resident in or can be recalled to an online VTSS.

Examples

To mount a scratch VTV on the VTD at device address 8900 and assign Management Class VSMLOCAL, enter:

```
M SCRTCH 8900 MGMT(VSMLOCAL)
```

MVCPOOL Control Statement

The MVCPool control statement defines your system's MVC pool and, optionally, Named MVC Pools within that pool.

Syntax

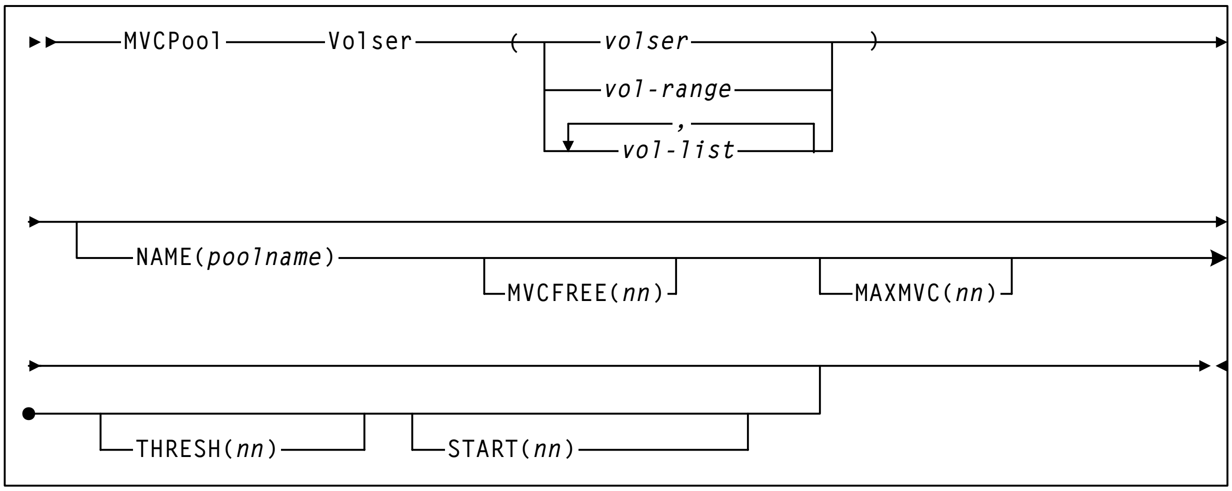


Figure 133. `MVCPool` ***Control Statement Syntax***

Parameters

`Volser`

defines the MVCs.

`volser`, `vol-range`, or `vol-list`

the volsers of one or more MVCs. If you specify multiple volume ranges, do not overlap them.

`NAME`

defines a Named MVC Pool. If you do not specify the `MVCPOOL NAME` parameter, VTCS does not create a Named MVC Subpool and assigns the specified volumes to the default pool (`DEFAULTPOOL`). You cannot create Named MVC Pools with the reserved names `DEFAULTPOOL` and `ALL`.

`poolname`

the MVC Pool name (up to 13 characters).

Note: You can use the optional `MVCFREE`, `MAXMVC`, `THRESH`, and `START` parameters to specify values for the Named MVC Pool that override the global values specified on `CONFIG`

MVCFREE(nnn)

specifies the minimum number of free MVCs in the MVC pool. A free MVC has 100% usable space and does not contain any migrated VTVs. Valid values are 0 to 255. There is no default; if not specified, the CONFIG RECLAIM value (or default) is used.

If free MVCs is equal or less than this value, VTCS issues message SLS6616I and starts an automatic space reclamation.

Note: If you set MVCFREE=0, VTCS actually uses the default value (40).

MAXMVC(nn)

specifies the maximum number of MVCs that will be processed in a single space reclamation run. Valid values are 1 to 98. There is no default; if not specified, the CONFIG RECLAIM value (or default) is used.

For automatic space reclamation to start, the number of eligible MVCs (determined by the THRESH parameter) must also exceed the MAXMVC value.

THRESH(nn)

specifies the percentage of fragmented space that makes an MVC eligible for demand or automatic reclamation. Valid values are 4 to 98. There is no default; if not specified, the CONFIG RECLAIM value (or default) is used.

START(nn)

specifies the level at which automatic space reclamation starts for each ACS (not globally for all ACSs) or, if specified, for a Named MVC Pool. Specify a percentage value, which is equal to:

(MVCs eligible for reclamation/Total available MVCs) * 100

Where:

MVCs eligible for reclamation

is the number of eligible MVCs determined by the THRESHLD parameter.

Total available MVCs

equals the number of eligible MVCs *plus* the number of free MVCs. A free MVC has 100% usable space and does not contain any migrated VTVs.

Valid values are 1 to 98. There is no default; if not specified, the CONFIG RECLAIM value (or default) is used.

Usage

Use the `MVCPool` control statement to define your system's MVC pool.

You use the `VT MVCDEF` command on page 102 to load the `MVCPool` control statements. For more information about using the `VT MVCDEF` command and the `MVCPool` statement.

You can use the `NAME` parameter to define a Named MVC Pool. You can use the optional `MVCFREE`, `MAXMVC`, `THRESH`, and `START` parameters to specify values for the Named MVC Pool that override the global values specified on `CONFIG`.

Example

The following `MVCPool` statement defines volsers 900000 - 909999 as MVCs:

```
MVCP V(900000 - 909999)
```

STORCLAS Control Statement

The `STORclas` control statement defines a VSM Storage Class.

Note: The `STORclas` control statement is valid only if `FEATures VSM(ADVMGMT)` is specified; for more information, see “FEATURES Control Statement” on page 194.

Syntax

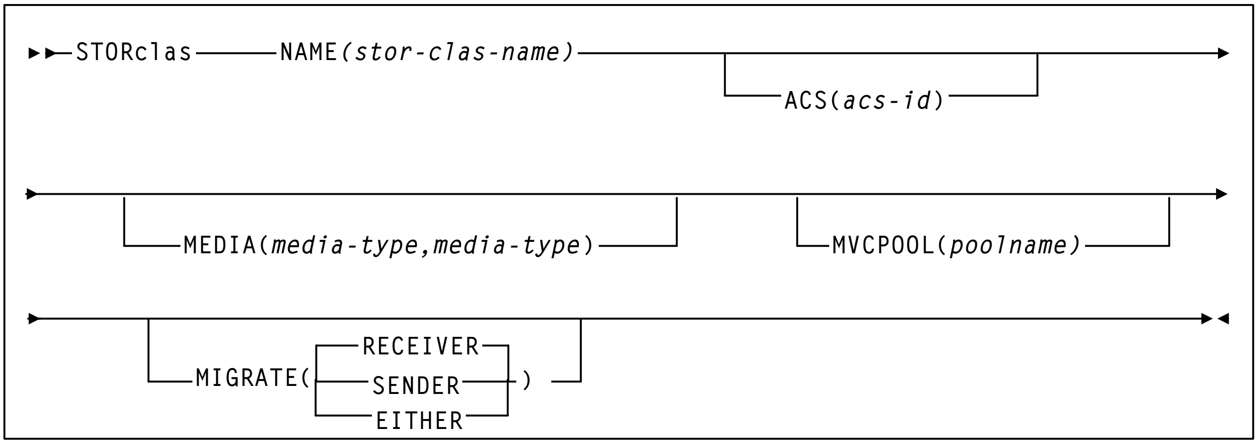


Figure 134. `STORclas` ***Control Statement Syntax***

Parameters

`NAME`

specifies the name of the Storage Class.

`stor-clas-name`

the Storage Class name. This name must be 1 to 8 alphanumeric characters beginning with an alpha character and must follow SMS naming conventions.

`ACS`

specifies the ACSs from which RTDs and MVCs are selected.

`acs-id`

Species the ACS ID. An ACS ID has a hexadecimal value from 00 through FE.

`MEDIA`

specifies a preference list of MVC media types. This list supersedes the default media selection list.

For migration and consolidation, VSM attempts to select MVC media in the specified order. For reclamation, VTVs are read from MVCs in the specified order, and written back to MVCs in the reverse of the specified order.

`media-type`

Species a media type.

MVCPOOL

specifies the Named MVC Pool from which volumes are selected. If you do not specify an MVC Pool name, the volumes are selected from the default pool (DEFAULTPOOL).

poolname

the name of an MVC Pool that you defined on the MVCPool control statement; for more information, see “MVCPOOL Control Statement” on page 216.

MIGRATE

for Management Classes with REPLICAT(YES) that reference this Storage Class, specifies the source VTSS (in a Cluster) for VTV migration.

RECEIVER

VTSS that receives the replicated VTV (the default), which is the Secondary VTSS in a Primary-Secondary Cluster.

SENDER

VTSS that sends the replicated VTV, which is the Primary VTSS in a Primary-Secondary Cluster.

EITHER

Either VTSS in a Peer-to-Peer Cluster.

Usage

Use the STORclas control statement to define a VSM Storage Class. The STORclas control statement is only valid if FEATures VSM(ADVMGMT) is specified; for more information, see “FEATURES Control Statement” on page 194. You can define a Storage Class for the Named MVC Pool specified on the MVCPOOL parameter.

The MGMTclas control statement can specify a Storage Class on the MIGpol, CONSRC, and CONTGT parameters. For more information, see “MGMTCLAS Control Statement” on page 204. You use the MGMTDEF command to load MGMTclas and STORclas control statements, which must reside in the same data set for cross-validation; for more information, see “MGMTDEF Command” on page 213.

Storage Classes are never mixed on the same MVC. If you do not specify a Storage Class on the MGMTclas statement, the Storage Class defaults to the VTSS name.

Caution: Note the following when using Named MVC pools:

- If you specify the MVCPOOL parameter, ensure that specified Named MVC Pool exists. Otherwise, the Storage Class is invalid, and VTCS reverts to the default Storage Class (the name of the last VTSS that wrote to the MVC for reclamation or migration) and a a ‘no MVCs available’ condition will occur. Either correctly define the Named MVC Pool or add Storage Class definitions for all VTSSs and associated them with the appropriate Named MVC pool(s).
- If you specify the MEDIA and MVCPOOL parameters, ensure that the Named MVC Pool contains MVCs of the specified media type(s); otherwise, a ‘no MVCs available’ condition will occur.

Note: If you do not explicitly assign a Storage Class, an MVC’s default Storage Class is the name of the last VTSS that wrote to the MVC for reclamation or migration and this class has the VTCS default media selections. To change these defaults, create a Storage Class with the VTSS name and specify the desired media selection order.

Caution: You cannot use the default Storage Class to group or segregate workloads.

For more information, see “MGMTCLAS Control Statement” on page 204.

Using the STORclas MEDIA Parameter for MVC Media Preferencing

- By default, in mixed-media VSM systems, VTV automatic and demand migrations (and consolidations) go to MVCs by media type in this order:
  1. STANDARD
  2. ECART
  3. ZCART
  4. STK1RAB
  5. STK1RC
  6. STK2PA
  7. STK2PB
- By default, for automatic and demand space reclamations, VSM writes VTVs to output MVCs by media type in this order:
  1. STK2PB
  2. STK2PA
  3. STK1RC
  4. STK1RAB
  5. ZCART
  6. ECART
  7. STANDARD

  Note that the preceding default media orders are composed of values from **either** the MEDIA **or** RECTECH value (but **not** both) of the VOLATTR statement for the media as shown in Table 13.

Table 13. MVC Media Types

Media	Description	VOLATTR Parameter
STANDARD	standard length 3480 cartridge	MEDIA
ECART	3490E cartridge	MEDIA
ZCART	3490EE cartridge	MEDIA
STK1RAB	T9840A/B cartridge	RECTECH
STK1RC	T9840C cartridge	RECTECH
STK2PA	T9940A cartridge	RECTECH
STK2PB	T9940B cartridge	RECTECH

- The `MEDIA` parameter of the `STORclas` statement (which defines Storage Classes) specifies a preference list of MVC media types. This list supersedes the default media selection list. **Note that** for reclamation, VTVs are written back to MVCs in the **reverse** of the order specified on the `MEDIA` parameter.

  For example, if you specify the following on the `MEDIA` parameter of the `STORclas` statement...

  `MEDIA(STK1RAB,STK1RC,STK2PB)`

  - ...to select an MVC for migration to this Storage Class, VTCS searches for a usable MVC in the order `STK1RAB`, `STK1RC`, `STK2PB`.
  - ...to select an MVC for the output of reclaim to this Storage Class, VTCS searches for a usable MVC in the order `STK2PB`, `STK1RC`, `STK1RAB`.

- You can specify the media and ACS preferencing via the Storage Class(es) specified on the `MIGpol` parameter of the `MGMTclas` control statement; for more information, see "MGMTCLAS Control Statement" on page 204.
- To optimize recall processing in mixed-media systems, ensure that your MVC pool has at least one media type compatible with each RTD type.

Examples

See *VTCS Installation and Configuration Guide* and *VTCS Administrator's Guide*.

STORLST Control Statement

The `STORLST` control statement specifies a list of Storage Classes and their corresponding preferencing.

Note: The `STORLST` control statement is valid only if `FEATures VSM(ADVMGMT)` is specified; for more information, see “FEATURES Control Statement” on page 194.

Syntax

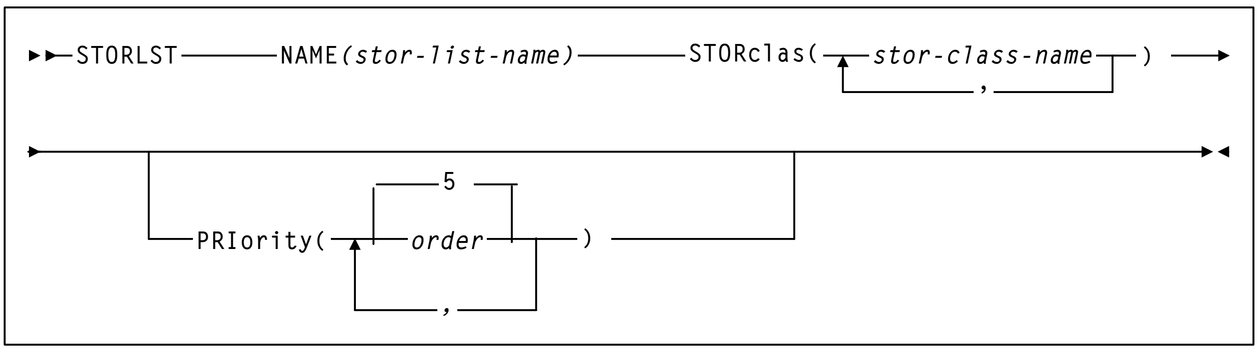


Figure 135. `STORLST` ***Control Statement Syntax***

Parameters

`NAME`

specifies the name of the Storage Class list.

`stor-list-name`

the list name (a maximum of 8 alphanumeric characters).

`STORclas`

specifies one to four Storage Classes on the Storage Class list.

`stor-clas-name`

the name of a Storage Class that you defined on the `STORclas` control statement; for more information, see “STORCLAS Control Statement” on page 219.

`PRIority`

a list of priorities corresponding to the Storage Classes specified on the `STORclas` parameter.

`order`

the specified priority. Valid values are 0 to 9 (highest priority), and the default is 5. You can assign the same priority to multiple Storage Classes. For example, if two Storage Classes both have a priority of 9, VTCS selects randomly from the two. A 0 (zero) priority specifies that VTCS selects the Storage Class only if all other Storage Classes are unavailable (for example, no free MVCs available for write).

Note: The Storage Class list is further qualified by the criteria specified by the `MGMTclas` and `VTSS` parameters of the `STORSEL` statement.

Usage

Use the STORLST control statement to specify a list of Storage Classes and their corresponding preferencing. The STORSEL statement, which defines a Storage Class usage rule, specifies a Storage Class preferencing list. For more information, see “STORSEL Control Statement” on page 226.

You use the MGMTDEF command to load the following statements, which must all reside in the same data set for cross-validation:

- MGMTclas
- STORclas
- VTSSLST
- VTSSSEL
- STORLST
- STORSEL

For more information, see “MGMTDEF Command” on page 213.

The STORLST control statement is only valid if FEATures VSM(ADVMGMT) is specified; for more information, see “FEATURES Control Statement” on page 194.

Examples

The following STORLST statement specifies list of local Storage Classes with MVCs in ACSs connected to a local VTSS with a priority of 1 for Storage Class SACS00 and a priority of 9 for Storage Class SACS01:

```
STORLST NAME(LOCALACS) STOR(SACS00,SACS01) PRI(1,9)
```

See also *VTCS Installation and Configuration Guide*.

STORSEL Control Statement

The `STORSEL` control statement defines a Storage Class usage rule that applies to the Storage Class list and its preferencing specified on a referenced `STORLST` control statement.

Note: The `STORSEL` control statement is valid only if `FEATures VSM(ADVMGMT)` is specified; for more information, see “FEATURES Control Statement” on page 194.

Syntax

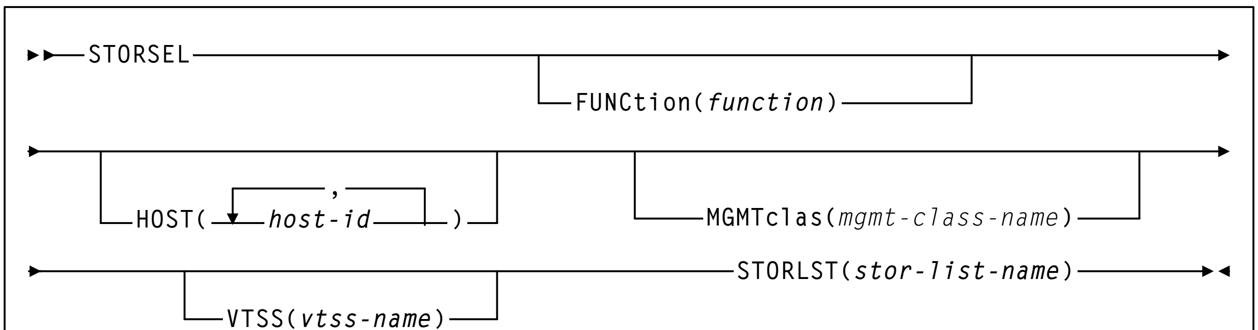


Figure 136. `STORSEL` ***Control Statement Syntax***

Parameters

FUNCtion

specifies the VSM function to which the rule applies.

function

the function name, as described in Table 2.

Table 14. ***STORSEL Functions***

Function	Explanation
SPECIFIC	Applies to automatic recall of a specific VTV for mounting. The list of Storage Classes is determined by the specified STORLST statement. This list influences the list of RTDs eligible to mount the MVC in the Storage Class to recall the VTV.
RECALL	Applies to demand recall of a specific VTV for mounting. The list of Storage Classes is determined by the specified STORLST statement. This list influences the list of MVC copies of a VTV to select the optimal MVC for recall of the VTV.
EXPORT	Applies to export. The list of Storage Classes is determined by the specified STORLST statement. This list influences the list of MVC copies of a VTV to select the optimal MVC for export of the VTV.
CONSOLID	Applies to consolidate. The list of Storage Classes is determined by the specified STORLST statement. This list influences the list of MVC copies of a VTV to select the optimal MVC for consolidation of the VTV.

Note: The HOST, MGMTclas, and VTSS parameters are optional and have no default values.

HOST

specifies one or more hosts to which the rule applies. Any hosts not specified on this parameter ignore the rule.

host-id

a host identifier (maximum 8 characters).

Note: The Storage Class list specified on the STORLST parameter is further qualified by the criteria specified by the MGMTclas and VTSS parameters.

MGMTclas

specifies a Management Class.

mgmt-class-name

the name of a Management Class that you defined on the MGMTclas control statement; for more information, see “MGMTCLAS Control Statement” on page 204.

VTSS

specifies a VTSS as follows:

- For automatic recalls, the VTSS where the recall is performed.
- For all other functions, the VTSS where the VTV previously resided.

`STORLST`

specifies a list of Storage Classes and their corresponding preferencing.

`stor-list-name`

the name of a Storage Class list that you defined on the `STORLST` control statement; for more information, see “STORLST Control Statement” on page 224.

Usage

Use the `STORSEL` statement to define a Storage Class usage rule that applies to the Storage Class list and its preferencing specified on a referenced `STORLST` control statement. For more information, see “STORLST Control Statement” on page 224.

You use the `MGMTDEF` command to load the following statements, which must all reside in the same data set for cross-validation:

- `MGMTclas`
- `STORclas`
- `VTSSLST`
- `VTSSSEL`
- `STORLST`
- `STORSEL`

For more information, see “MGMTDEF Command” on page 213.

The `STORSEL` control statement is only valid if `FEATures VSM(ADVMGMT)` is specified; for more information, see “FEATURES Control Statement” on page 194.

Examples

The following `STORSEL` statement VTV mounts for automatic recall for host `MSP8` to a preference list (`LOCALACS`) of ACSs connected to local VTSSs.

```
STORSEL FUNC(RECALL) HOST(MSP8) STORLST(LOCALACS)
```

See also *VTCS Installation and Configuration Guide*.

VOLATTR Control Statement

For VSM, the enhanced HSC VOLATTR control statement specifies VTV attributes, including the volser and media type (virtual).

Syntax

```
►►──VOLATTR──SERial──(──┬──volser─────────┬──)──┬──────────────────┬──►◄
                        ├──vol-range──────┤     ├──MEDia(Virtual)──┤
                        │  ┌─────,─────┐  │     └──RECtech(Virtual)┘
                        └──▼─vol-list──┴──┘
```

*Figure 137.*VOLATTR ***Control Statement Syntax***

Parameters

Unchanged VOLATTR Parameters

The SERial VOLATTR parameter is unchanged but applies to VSM. Figure 1 shows valid values for this parameter; see Chapter 3, "Control Statements and Start Procedure" of *HSC System Programmer's Guide for MSP* for more information.

Hint: When you create VOLATTR statements for VTVs, you use the SERial parameter to specify the VTV volsers.

Caution: On VOLATTR statements for VTVs, do *not* specify duplicate volsers or overlapping volser ranges.
In addition, after you define an initial set of VTV volsers, you can add more volsers but you should not change your initial set of VTV volsers, which wastes HSC CDS space. For example, if you initially define VTVs V00000 - V99999, you can later add VTVs W00000 - W99999 by specifying both volser ranges when you update the VOLATTR statement that specifies your system's VTVs. If you update the VOLATTR statement to change the volser range from V00000 - V99999 to W00000 - W99999, hosts can still access the original range (V00000 - V99999). If a host scratches a VTV in the original range, however, the VTV cannot be reused, but continues to take up space in the CDS.

VOLATTR Parameters Enhanced for VSM

The following `VOLATTR` parameters have a new *required* value of `Virtual` for VTVs only. `Virtual` does *not* apply to physical HSC volumes.

`MEDia`

specifies the volume media.

`Virtual`

specifies that VSM will route data sets to a VTV mounted on a VTD.

`RECtech`

specifies the recording technique.

`Virtual`

specifies that VSM will route the data set to a VTV mounted on a VTD.

Note: To define a volume as virtual, you must specify the `Virtual` keyword for either the `MEDia` or `RECtech`. You can also specify both `MEDia` or `RECtech`; the keyword must be `Virtual` for both parameters to define the volume as virtual.

Usage

For VSM, use the enhanced `HSC VOLATTR` control statement to specify VTV attributes, including the volser and media type (virtual).

You must use the `VOLDEF` command to load `VOLATTR` control statements from a specified definition data set; see Chapter 3, “Control Statements and Start Procedure” of *HSC System Programmer’s Guide for MSP* for more information.

Example

To define VTVs to be all volumes whose volsers begin with VT, create the following `VOLATTR` statement:

```
VOLATTR SERIAL(VT*) MEDIA(Virtual)
```

VTSSLST Control Statement

The `VTSSLST` control statement specifies a list of VTSSs and their corresponding preferencing.

Note: The `VTSSLST` control statement is valid only if `FEATures VSM(ADVMGMT)` is specified; for more information, see “FEATURES Control Statement” on page 194.

Syntax

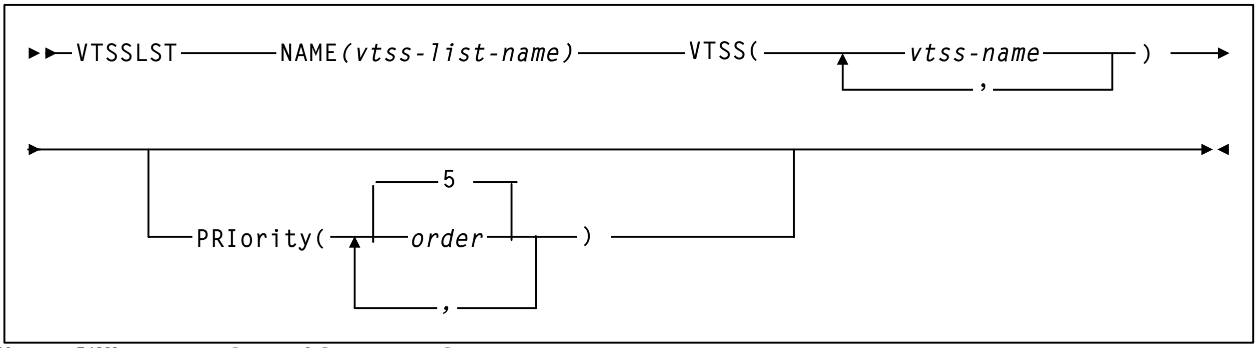


Figure 138. `VTSSLST` ***Control Statement Syntax***

Parameters

`NAME`

specifies the name of the VTSS list.

`vtss-list-name`

the list name (a maximum of 8 alphanumeric characters).

`VTSS`

specifies one to eight VTSSs on the VTSS list.

`vtss-name`

a VTSS identifier.

`PRIority`

a list of priorities corresponding to the VTSS identifiers specified on the `VTSS` parameter.

`order`

the specified priority. Valid values are 0 to 9 (highest priority), and the default is 5. You can assign the same priority to multiple VTSSs. For example, if two VTSSs both have a priority of 9, VTCS selects randomly from the two (according to factors such as DBU and VSM model). A 0 (zero) priority specifies that VTCS selects the VTSS only if all other VTSSs are unavailable (for example, unavailable due to DBU > 95%, VTSS offline, or RTDs offline).

Note: The VTSS list specified on the `VTSSLST` parameter is further qualified by:

- The function specified on `VTSSSEL` statement.
- The criteria specified by the `MGMTclas`, `VTSS`, `STORclas`, and `MVCpool` parameters of the `VTSSSEL` statement.
- Other factors such as RTD connectivity.

For example, in scratch allocation, the list of VTSSs is reduced to the VTSSs that can meet Management Class policies (such as `REPLICAT(YES)`). If the list of VTSSs is reduced to zero, the request fails.

Usage

Use the `VTSSLST` control statement to specify a list of VTSSs and their corresponding preferencing. The `VTSSSEL` statement, which defines a VTSS usage rule, specifies a VTSS preferencing list. For more information, see “VTSSSEL Control Statement” on page 233.

You use the `MGMTDEF` command to load the following statements, which must all reside in the same data set for cross-validation:

- `MGMTclas`
- `STORclas`
- `VTSSLST`
- `VTSSSEL`
- `STORLST`
- `STORSEL`

For more information, see “MGMTDEF Command” on page 213.

The `VTSSLST` control statement is only valid if `FEATures VSM(ADVMGMT)` is specified; for more information, see “FEATURES Control Statement” on page 194.

Examples

The following `VTSSLST` statement specifies list of local VTSSs with a priority of 1 for VTSS01 and a priority of 9 for VTSS02:

```
VTSSLST NAME(LOCAL) VTSS(VTSS01,VTSS02) PRI(1,9)
```

See also *VTCS Installation and Configuration Guide*.

VTSSSEL Control Statement

The `VTSSSEL` control statement defines a VTSS usage rule that applies to the VTSS list and its preferencing specified on a referenced `VTSSLST` control statement.

Note: The `VTSSSEL` control statement is valid only if `FEATures VSM(ADVMGMT)` is specified; for more information, see “FEATURES Control Statement” on page 194.

Syntax

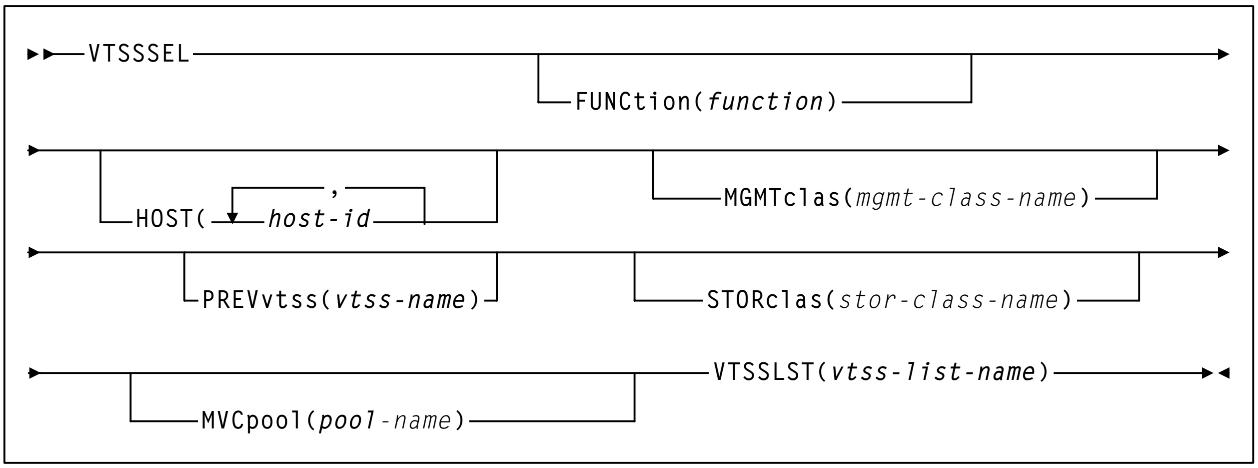


Figure 139. `VTSSSEL` ***Control Statement Syntax***

Parameters

`FUNCtion`

specifies the VSM function to which the rule applies.

`function`

the function name, as described in Table 15.

Table 15. VTSSSEL Functions

Function	Explanation
SCRATCH	Applies to non-specific (scratch) VTV allocation. The list of eligible VTDs is determined by the specified VTSSLST statement. The PREVVTSS, STORclas and MVCpool parameters do not apply.
SPECIFIC	Applies to specific VTV allocation. The list of eligible VTDs is determined by the specified VTSSLST statement. The STORclas and MVCpool parameters do not apply.
RECALL	Applies to demand recall. The list of eligible VTSSs for recall is determined by the specified VTSSLST statement. The VTSS list also determines the search order for an RTD to service the MVCs selected for recall (derived from the VTVs selected). The MGMTclas parameter does not apply.
RECLAIM	Applies to reclaim. The list of eligible VTSSs for reclaim is determined by the specified VTSSLST statement. The VTSS list also determines the search order for an RTD to service the MVCs selected for reclaim. The MGMTclas parameter does not apply.
DRAIN	Applies to drain. The list of eligible VTSSs for drain is determined by the specified VTSSLST statement. The VTSS list also determines the search order for an RTD to service the MVCs selected for drain. The MGMTclas parameter does not apply.
AUDIT	Applies to MVC audit. The list of eligible VTSSs for audit is determined by the specified VTSSLST statement. The VTSS list also determines the search order for an RTD to service the MVCs selected for audit. The MGMTclas parameter does not apply.
EXPORT	Applies to export. The list of eligible VTSSs for export is determined by the specified VTSSLST statement. VTSS list also determines the search order for an RTD to service the MVCs selected for export. The MGMTclas parameter does not apply.
CONSOLID	Applies to consolidate. The list of eligible VTSSs for consolidation is determined by the specified VTSSLST statement. TTSS list also determines the search order for an RTD to service the MVCs selected for consolidation. The MGMTclas parameter does not apply.

`HOST`

specifies one or more hosts to which the rule applies. Any hosts not specified on this parameter ignore the rule.

`host-id`

a host identifier (maximum 8 characters).

Note: The VTSS list specified on the `VTSSLST` parameter is further qualified by the criteria specified by the `MGMTclas`, `VTSS`, `STORclas`, and `MVCpool` parameters.

`MGMTclas`

specifies a Management Class.

`mgmt-class-name`

the names of a Management Class that you defined on the `MGMTclas` control statement; for more information, see “MGMTCLAS Control Statement” on page 204.

`PREVvtss`

specifies a VTSS where a VTV:

- Is or resident.
- Was migrated from.

`STORclas`

specifies a Storage Class and applies only when MVCs are used to select VTSSs.

`stor-clas-name`

the name of a Storage Class that you defined on the `STORclas` control statement; for more information, see “STORCLAS Control Statement” on page 219.

`MVCpool`

specifies a Named MVC Pool that contains and MVC and applies only when MVCs are used to select VTSSs.

`poolname`

the name of an MVC Pool that you defined on the `MVCPool` control statement; for more information, see“MVCPOOL Control Statement” on page 216.

`VTSSLST`

specifies a list of VTSSs and their corresponding preferencing.

`vtss-list-name`

the name of a VTSS list that you defined on the `VTSSLST` control statement; for more information, see “VTSSLST Control Statement” on page 231.

Usage

Use the VTSSSEL statement to define a VTSS usage rule that applies to the VTSS list and its preferencing specified on a referenced VTSSLST control statement. For more information, see "VTSSLST Control Statement" on page 231.

You use the MGMTDEF command to load the following statements, which must all reside in the same data set for cross-validation:

- MGMTclas
- STORclas
- VTSSLST
- VTSSSEL
- STORLST
- STORSEL

For more information, see "MGMTDEF Command" on page 213.

The VTSSSEL control statement is only valid if FEATures VSM(ADVMGMT) is specified; for more information, see "FEATURES Control Statement" on page 194.

Examples

The following VTSSEL statement restricts scratch VTV mounts for host MSP8 to a preference list (LOCAL) of VTSSs.

```
VTSSSEL FUNC(SCRATCH) HOST(MSP8) VTSSLST(LOCAL)
```

See also *VTCS Installation and Configuration Guide*.

HSC Programmatic Interface Enhancements

The MOUNT, QDRLIST, and SELSCR requests support an additional value of VIRTUAL (to specify a VTV) for the MEDIA and RECtech parameters.

For these requests for VSM:

- For scratch requests, within a VTSS, VSM selects the first available VTD with the lowest device address.

  In a multi-VTSS system, VSM will determine which VTSS is optimal for the request, then select the first available VTD with the lowest device address in that VTSS. For a given request, VSM limits VTD selection to the first 8 VTSSs, but will select from all VTSSs for a series of requests.
- For specific requests, depending on the level of VTCS, VSM will either return a list of all VTDs or preferred VTDs.
- The SCRPOOL parameter is invalid; specify SUBPOOL instead.
- The MOUNT request supports an additional parameter of MGMTclas that can assign a VSM Management Class to the VTV.

The volume information element returned for a QVOLUME request or a MOUNT request for a virtual volume includes a value of VIRTUAL for media type (SLXVMED) and an x'01' for volume status (SLXVSTA).

For more information on these requests, see Appendix E, “Programmatic Interface” of the *HSC System Programmer’s Guide for MSP*.

HSC User Exit Enhancements

Use return code UX02VIRT (32) in register 15 in HSC User Exit SLSUX02 (JES), which you use to control transport allocation for scratch mounts. To satisfy a scratch mount request, return code UX0xVIRT causes VSM to select an available VTD in your system and routes the job to a VTV mounted on that VTD.

For more information about HSC User Exits, see Chapter 9, “User Exits” of *HSC System Programmer’s Guide for MSP*.

HSC Batch API Enhancements

The HSC Batch API supports bulk reading of CDS VTV and MVC records. For more information about the Batch API, see Appendix F, “Batch Application Interface (API)” of *HSC System Programmer’s Guide for MSP*.

Hint: Appendix F, “Batch Application Interface (API)” of *HSC System Programmer’s Guide for MSP* provides an example of a QCDS request that retrieves VTV records.

For more information about the HSC Batch API enhancements for VSM, see the following sections:

- “Batch API Mapping Macros” on page 239
- “SLSUREQ QCDS Request” on page 242
- “Library Element Mapping” on page 242

The HSC Batch API provides additional data in the following records returning from a Batch API Query CDS request:

- The VTV record now provides the compressed and uncompressed size of the VTV. For more information, see “SLUVTDAT Macro Record Format” on page 241.
- The MVC record now provides an MVC status indicator. For more information, see “SLUVMDAT Macro Record Format” on page 239.

Batch API Mapping Macros

The following sections described the macros to support VSM:

- “SLUVMDAT Macro Record Format”
- “SLUVTDAT Macro Record Format” on page 241

SLUVMDAT Macro Record Format

Table 16 describes the SLUVMDAT macro record format.

Table 16. SLUVMDAT Macro Record Format

Dec	Hex	Type	Length	Label	Description
SLUVMDAT - FLAT FILE MVC DATA DSECT FUNCTION: DESCRIBES THE MVC DATA WHICH IS GENERATED TO THE FLAT FILE BY THE BATCH API					
0	(0)	STRUCTURE		MDREC	FLAT FILE RECORD
0	(0)	SIGNED-FWORD	4	MDRECRDW	RECORD DESCRIPTOR WORD
4	(4)	SIGNED-FWORD	4	MDRECL	LENGTH
8	(8)	CHARACTER	1	MDRECC	CHARACTER EBCDIC/ASCII
9	(9)	CHARACTER	1	MDRECT	TYPE M - MVC
10	(A)	CHARACTER	6	MDRECM	MVC VOLSER
16	(10)	SIGNED-FWORD	4	MDRECVC	VTV COUNT
20	(14)	SIGNED-FWORD	4	MDRECPU	PERCENT USED
24	(18)	SIGNED-FWORD	4	MDRECPA	PERCENT AVAILABLE
28	(1C)	SIGNED-FWORD	4	MDRECPW	PERCENT WASTED
32	(20)	SIGNED-FWORD	4	MDRECMC	MOUNTED COUNT
36	(24)	SIGNED-FWORD	4	MDRECTL	TIME LAST USED HIGH ORDER WORD FROM STACK INSTRUCTION
40	(28)	SIGNED-FWORD	4	MDRECMS	MEDIA SIZE
44	(2C)	LENGTH		MDRECLEN	LENGTH OF RECORD WHEN USING VERSIONS 1 AND 2 OF SLSUREQM
44	(2C)	BITSTRING	1	MDRECERR	MVC STATUS INDICATOR
		X’80’		MDINITD	MVC INITIALIZED FROM A MIGRATE
		X’40’		MDMOUNT	MVC IS MOUNTED ON AN RTD
		X’20’		MDBROKE	MVC HAS AN ERROR

Table 16. SLUVMDAT Macro Record Format

Dec	Hex	Type	Length	Label	Description
		X’10’		MDFULL	MVC CANNOT CONTAIN ANY MORE VTVS
		X’08’		MDDRAIN	MVC IS BEING DRAINED
		X’04’		MDLOST	MVC IS LOST (LAST MOUNT TIMED OUT)
		X’02’		MDDATCK	MVC SWAPPED (NOT IN RECOVERY)
		X’01’		MDREADO	MVC IS READONLY
45	(2D)	RESERVED	3		
48	(30)	LENGTH		MDRECLEN3	LENGTH OF RECORD LENGTH OF RECORD WHEN USING VERSIONS 3 OF SLSUREQM

SLUVTDAT Macro Record Format

Table 17 describes the SLUVTDAT macro record format.

Table 17. SLUVTDAT Macro Record Format

Dec	Hex	Type	Length	Label	Description
SLUVMTDAT - FLAT FILE VTV DATA DSECT FUNCTION: DESCRIBES THE VTV DATA WHICH IS GENERATED TO THE FLAT FILE BY THE BATCH API					
0	(0)	STRUCTURE		VDREC	FLAT FILE RECORD
0	(0)	SIGNED-FWORD	4	VDRECRDW	RECORD DESCRIPTOR WORD
4	(4)	SIGNED-FWORD	4	VDRECL	LENGTH
8	(8)	CHARACTER	1	VDRECC	CHARACTER EBCDIC/ASCII
9	(9)	CHARACTER	1	VDRECT	TYPE V - VTV
10	(A)	CHARACTER	6	VDRECV	VTV VOLSER
16	(10)	CHARACTER	8	VDRECVT	VTSS
24	(18)	SIGNED-FWORD	4	VDRECSZ	SIZE (MB)
28	(1C)	CHARACTER	1	VDRECM	MIGRATED Y/N
29	(1D)	CHARACTER	1	VDRECD	DUPLEX Y/N
30	(1E)	CHARACTER	6	VDRECM1	MVC VOLSER OF FIRST/ONLY COPY
36	(24)	CHARACTER	6	VDRECM2	MVC VOLSER OF SECOND COPY
42	(2A)	CHARACTER	1	VDRECI	INVALID Y/N
43	(2B)	CHARACTER	1	VDRECS	SCRATCH Y/N
44	(2C)	SIGNED-FWORD	4	VDRECTC	HIGH ORDER WORD OF TOD CLOCK (GMT) RETURNED BY STCK INSTRUCTION
48	(30)	SIGNED-FWORD	4	VDRECTL	HIGH ORDER WORD OF TOD CLOCK (GMT) RETURNED BY STCK INSTRUCTION
52	(34)	CHARACTER	8	VDRECMC	MANAGEMENT CLASS
60	(3C)	LENGTH		VDRECLEN	LENGTH OF RECORD WHEN USING SLSUREQM VERSIONS 1 AND 2
60	(3C)	SIGNED-FWORD	4	VDUCMPSZ	VTV UNCOMPRESSED SIZE (BYTES)
64	(40)	SIGNED-FWORD	4	VDCOMPSZ	VTV COMPRESSED SIZE (BYTES)
68	(44)	CHARACTER	6		RESERVED

Table 17. SLUVTDAT Macro Record Format

Dec	Hex	Type	Length	Label	Description
74	(4A)	CHARACTER	6		RESERVED
80	(50)	LENGTH		VDRECLEN	LENGTH OF RECORD WHEN USING SLSUREQM VERSION 3

SLSUREQ QCDS Request

The following TYPE= values are valid on the SLSUREQ QCDS request:

`MVC`
: specifies the VTCS MVC record area of the CDS.

`VTV`
: specifies the VTCS VTV record area of the CDS.

Library Element Mapping

Table 18 describes the Library Element Record Mapping additions.

Table 18. Library Element Record Mapping Additions

Request	Records Returned
READ MVC	VTCS MVC records mapped by the SLUVMDAT macro
READ VTV	VTV records mapped by the SLUVTDAT macro

HSC Operator Command Enhancements

With HSC:

- You can dynamically reload SCRPOOL statements via the SCRPDEF command. For more information, see “Scratch Subpool Definition (SCRPDEF) Command and Control Statement” in Chapter 3, “HSC Control Statements and HSC Start Procedure” of *HSC System Programmer’s Guide for MSP*.
- The Warn SCRatch, Display SCRatch, and Display THReshld commands are enhanced to let you manage and monitor scratch VTVs. For more information, see:
  - “HSC WARN SCRATCH Command Enhancements” on page 244
  - “HSC DISPLAY THRESHLD Command Enhancements” on page 245
  - “HSC DISPLAY SCRATCH Command Enhancements” on page 247
- You can expand the CDS using the CDS EXPAND command.
- You can use the TRACELKP command to trace HSC definition data sets, including the following:
  - VOLATTR
  - LMUPDEF
  - MVCPOOL
  - MGMTCLAS
  - STORCLAS

  For more information, see Chapter 2, “Commands, Control Statements, and Utilities,” in *HSC Operator’s Guide for MSP*.

HSC WARN SCRATCH Command Enhancements

For VSM, the enhanced HSC `Warn SCRatch` command sets scratch warning thresholds for HSC subpools that contain VTVs.

Syntax

```
►►──Warn SCRatch──VSM──┬─────────────────────────┬──THReshld(threshold-value)──►◄
                       └──SUBpool(subpool-name)──┘
```

Figure 140. Warn SCRatch VSM ***Command Syntax***

Parameters

`VSM`

Specifies that the command applies to VTVs only.

`SUBpool`

Specifies an HSC subpool that contains VTVs. If not specified, the command applies to the total VSM scratch count.

`subpool-name`

The specified subpool.

`THReshld`

Specifies the scratch warning threshold. If the number of scratches goes below the specified threshold, HSC issues a warning message. Valid values are 0 to 9999.

`threshold-value`

The specified threshold.

Usage

Use `Warn SCRatch` to set scratch warning thresholds for HSC subpools that contain VTVs.

Example

To set a threshold of 50 VTVs for subpool VIRT01, enter the following:

```
W SCR VSM SUB(VIRT01) THR(50)
```

HSC DISPLAY THRESHLD Command Enhancements

For VSM, the enhanced HSC `Display THReshld` command displays scratch thresholds for HSC subpools that contain VTVs.

Syntax

```
►►─Display THReshld───VSM──────────────────────────────────────────►◄
                                  └─SUBpool(subpool-name)─┘
```

Figure 141. `Display THReshld VSM` ***Command Syntax***

Parameters

`VSM`
: Specifies that the command applies to VTVs only.

`SUBpool`
: Specifies an HSC subpool that contains VTVs. If not specified, the command applies to the total VSM scratch count.

  `subpool-name`
  : The specified subpool.

Usage

Use `Display THReshld` to display scratch thresholds for HSC subpools that contain VTVs.

Example

To display the scratch threshold for subpool `VIRT01`, enter the following:

```
D THR VSM SUB(VIRT01)
```

Output

Figure 144 shows an example of HSC `Display THR VSM` output.

ACS/LSM		SUBPOOL	NAME	MEDIA	RECTECH	COUNT	THRESH
VSM		SUBPOOL	VIRT01	TOTAL		20	10
VSM		SUBPOOL	VIRT02	TOTAL		20	10
VSM		NON-SUBPOOL		TOTAL		60	50
VSM	TOTAL					100	100

Figure 142. ***Example output from HSC*** `Display THR VSM`

Note: VSM information is displayed if:

- You explicitly specify `Display THReshld VSM`.
- You specify `Display THReshld` (no keywords) and VSM is active.

When there are no subpools defined for VSM, the VSM summary includes only the “Non-Subpool” and “Total” lines. If VSM is not active when an explicit request is made for VSM thresholds, the will indicates “Nothing to Display”.

HSC DISPLAY SCRATCH Command Enhancements

For VSM, the enhanced HSC `Display SCRatch` command displays scratch counts for HSC subpools that contain VTVs.

Syntax

```
►►─Display SCRatch───────VSM──────────────────────────────────────────────────►◄
                                    └─SUBpool(subpool-name)─┘
```

Figure 143. Warn SCRatch VSM ***Command Syntax***

Parameters

`VSM`

Specifies that the command applies to VTVs only.

`SUBpool`

Specifies an HSC subpool that contains VTVs. If not specified, the command applies to the total VSM scratch count.

`subpool-name`

The specified subpool.

Usage

Use `Display SCRatch` to display scratch counts for HSC subpools that contain VTVs. Scratch counts are reported as follows:

- Scratch counts are reported for all subpools that have a warning threshold set, even if the scratch count is zero.
- If a calculation of the “non-subpool” count results in a negative number, a count of zero is displayed.

Example

To display the scratch count for subpool `VIRT01`, enter the following:

```
D SCR VSM SUB(VIRT01)
```

Output

Figure 144 shows an example of HSC `Display SCR VSM` output.

ACS/LSM		SUBPOOL	NAME	MEDIA	RECTECH	COUNT
VSM		SUBPOOL	VIRT01	TOTAL		20
VSM		SUBPOOL	VIRT02	TOTAL		20
VSM		NON-SUBPOOL		TOTAL		60
VSM	TOTAL					100

Figure 144.**Example output from HSC** `Display SCR VSM`

Note: VSM information is displayed if:

- You explicitly specify `Display SCRatch VSM`.
- You specify `Display SCRatch` (no keywords) and VSM is active.

When there are no subpools defined for VSM, the VSM summary includes only the "Non-Subpool" and "Total" lines. If VSM is not active when an explicit request is made for VSM scratch counts, the will indicates "Nothing to Display".

HSC DELDISP Parameter Enhancements

HSC provides two new settings for the LIBGEN DELDISP parameter of the SLILIBRY macro:

ASCRTCH
: (All scratch). Both real tape volumes and VTVs are made scratch if they were mounted scratch and the disposition on the dismount message is delete ('D').

VSCRTCH
: (Virtual scratch). Only VTVs are made scratch if they were mounted scratch and the delete disposition on the dismount message is delete ('D').

The current DELDISP settings (SCRTCH and NOSCRTCH) define scratch handling at dismount **only** for real volumes. If DELDISP is set to either of these values, VTVs are **never** scratched at dismount.

On pre-5.0 HSC systems, acting against a CDS where the DELDISP is set to ASCRTCH, the setting is handled as SCRTCH. Similarly, if the DELDISP has been set to VSCRTCH, the setting is handled as NOSCRTCH.

Note: A LIBGEN and MERGECDS or RECONFIG utility is not required to change the DELDISP setting in an existing system. The DELDISP setting can be changed with the HSC SET utility; it now accepts the two new settings. Active systems must be recycled to affect the change for DELDISP. Once set, the DELDISP setting is persistent across HSC initializations.

Chapter 4. LibraryStation Enhancements and Additions for VSM

This chapter contains reference information about the following enhancements and additions to LibraryStation to support VSM:

- "SPNUM Statement" on page 252, a new LibraryStation `LSDEF` file statement that which defines a LibraryStation subpool that corresponds to an HSC subpool.
- "VIRTACS Statement" on page 255, a new LibraryStation `LSDEF` file statement that defines a virtual ACS that maps to a VTSS to let clients connect to VSM.
- "SLGDIAG VIRTUAL_DRIVE Parameter" on page 256, which you can use to verify VSM and HSC operation with LibraryStation in the same `SLGDIAG` job.

SPNUM Statement

The enhanced SPNUM statement, which defines a LibraryStation subpool that corresponds to an HSC subpool, lets MVS/CSC and non-MVS/CSC clients request VTV mounts and pass a Management Class to VSM.

Syntax

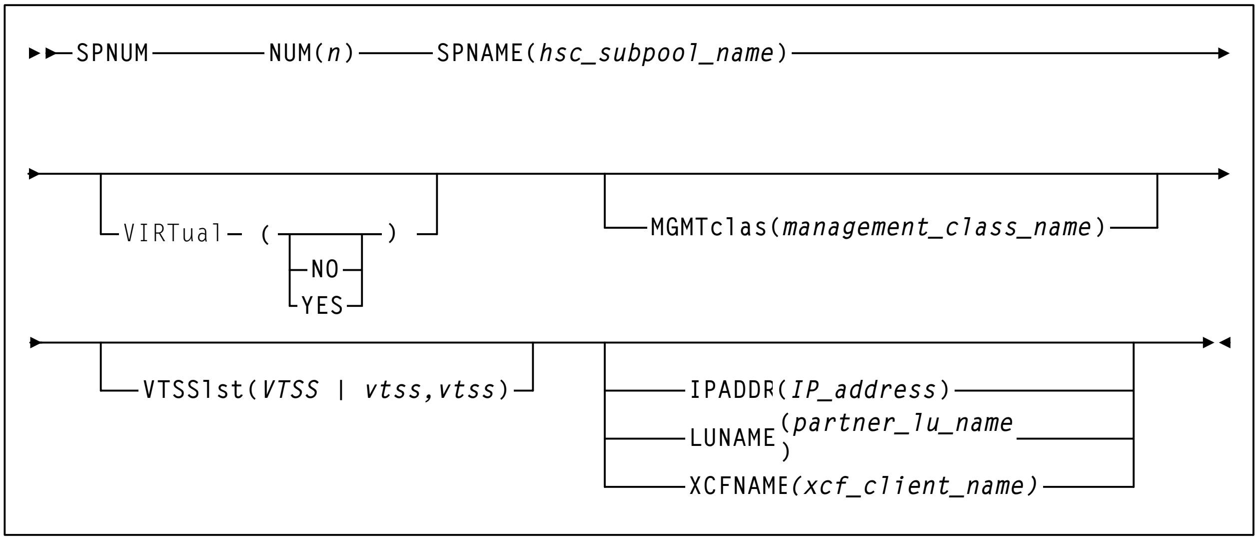


Figure 145. SPNUM ***Statement Syntax***

Parameters

Unchanged SPNUM Parameters

The following SPNUM parameters are unchanged but apply to VSM. Figure 145 shows valid values for these parameters; see Chapter 10, “Configuring the LSDEF Data Set” of *LibraryStation Configuration Guide* for more information.

- NUM
- SPNAME
- IPADDR
- LUNAME
- XCFNAME

New SPNUM Parameters

The following new SPNUM parameters apply to LibraryStation in VSM environments. These parameters allow MVS/CSC and non-MVS/CSC clients to request VTV mounts.

VIRTual

specifies whether the subpool contains VTVs.

NO

the subpool does not contain VTVs.

YES

the subpool contains VTVs.

Note: If VIRTual is not specified, the subpool can contain both real volumes and VTVs.

MGMTclas

specifies the name of a Management Class you defined on the HSC MGMTclas control statement; for more information, see "MGMTCLAS Control Statement" on page 204.

mgmt-class-name

the Management Class name.

Caution: For MVS/CSC clients, if you specify a Management Class when routing data to VSM (on a TAPEREQ statement, for example), StorageTek recommends that you do **not** specify a Management Class on the SPNUM statement.

VTSS

specifies one or two VTSSs used to satisfy the mount request.

vtssname| vtssname, vtssname

the names of one or two VTSSs.

Note: Each VTSS name must correspond to a VTSS name specified on a VIRTACS statement; for more information, see "VIRTACS Statement" on page 255.

Usage

Use the enhanced SPNUM statement to define a LibraryStation VTV subpool that corresponds to an HSC subpool. This subpool lets MVS/CSC and non-MVS/CSC clients request VTV mounts and pass a Management Class to VSM. You can also use the VTSSLST parameter to specify one or two VTSSs used to satisfy the mount request. You use the LibraryStation VIRTACS statement to define a virtual ACS that maps to a VTSS to let clients connect to VSM; for more information, see "VIRTACS Statement" on page 255.

Note: LibraryStation subpools can contain mixtures of VTVs and real Nearline volumes. If your client allows it, however, to simplify volume management, StorageTek recommends that you define subpools that contain only VTVs or real Nearline volumes, not mixtures of them.

You specify a Management Class on the `MGMTclas` parameter to pass this Management Class to VSM for each VTV mount. You define Management Classes with the `MGMTclas` statement; for more information, see "MGMTCLAS Control Statement" on page 204. You use the `MGMTDEF` command to load `MGMTclas` statements from a specified definition data set (which resides on the HSC/LibraryStation server); for more information, see "MGMTDEF Command" on page 213.

Examples

For example, for non-MVS/CSC clients, create the following `SPNUM` statement to:

- Define VTV subpool 7 that corresponds to HSC subpool `LSVIRT1`
- Pass Management Class `MGMTCLS7` to VSM when a VTV is mounted
- Specify VTSSs `VTSS01` and `VTSS02` are used to satisfy VTV mounts
- Restrict VTV mount requests to the client at IP address 129.80.57.16

```
SPNUM NUM(07) SPNAME(LSVIRT1) VIRT(YES) MGMT(MGMTCLS7)
VTSSL(VTSS01,VTSS02) IPADDR(129.80.57.16)
```

VIRTACS Statement

The `VIRTACS` statement defines a virtual ACS that maps to a VTSS to let clients connect to VSM.

Syntax

```
►►──────VIRTACS─ID·(acsid)────────VTSSNAME·(vtss-name)────────────────────────────►◄
```

Figure 146. `VIRTACS` ***Statement Syntax***

Parameters

`ID`
: specifies a virtual ACS ID.

  `acsid`
  : a decimal virtual ACS ID.

`VTSSNAME`
: specifies the VTSS identifier that maps to the virtual ACS ID.

  `vtss-name`
  : a VTSS identifier.

Usage

Use the `VIRTACS` statement to define a virtual ACS that maps to a VTSS to let clients connect to VSM.

Examples

To define virtual ACS 126 and map it to VTSS VTSS02, create the following `VIRTACS` statement:

```
VIRTACS ID(126) VTSSNAME(VTSS02)
```

SLGDIAG VIRTUAL_DRIVE Parameter

The `SLGDIAG` utility now provides `VIRTUAL_DRIVE` parameter that verifies LibraryStation operation with VSM in the following format (all decimal numbers):

```
=VIRTUAL_DRIVE=ascid,lsmid,panelid,driveid
```

See V*TCS Installation and Configuration Guide* for more information on VTD drive addresses for LibraryStation and NCS clients.

You can use `SLGDIAG` to verify LibraryStation operation with VSM in either of the following ways:

- To verify LibraryStation operation with only VSM (but not with HSC), specify the `=VIRTUAL_DRIVE=` parameter to query the specified VTD.
- To verify LibraryStation operation with VSM and HSC in the same batch job, specify the `=VIRTUAL_DRIVE=` parameter and also the existing `=DRIVE=` and `=VOLUME=` parameters (which request a mount/dismount on the specified Nearline transport).

For more information on the `SLGDIAG` utility, see Chapter 5, “Administration and Maintenance” in *LibraryStation Operator and System Programmer’s Guide*.

Chapter 5. MVS/CSC Enhancements and Additions for VSM

This chapter contains reference information about the following enhancements and additions to MVS/CSC to support VSM:

- “MVS/CSC Startup Parameter Enhancements” on page 258 for the DEFER and FETCH parameters.
- “MVS/CSC DISPLAY Command Enhancements” on page 258.
- “MVS/CSC User Exit Enhancements” on page 258.
- “MVS/CSC Programmatic Interface Enhancements” on page 259.
- “MVS/CSC DELDISP Parameter Enhancements” on page 259

Note: For NCS 6.1, the TAPEREQ statement (and the accompanying TREQDEF command) has been moved from HSC and MVS/CSC to SMC. For more information, see “TAPEREQ Control Statement” on page 184 and *SMC Configuration and Administration Guide*.

Caution: If you are using MVS/CSC to request mounts of scratch VTVs from an HSC scratch subpool, if the subpool is empty, MVS/CSC will not issue a message indicating that the subpool is empty. Instead, VTCS issues a message SLS6671E and continues to retry the scratch mount until you make scratch VTVs available!

MVS/CSC Startup Parameter Enhancements

The following sections describe the MVS/CSC startup parameter enhancements for VSM. For more information about MVS/CSC startup parameters, see Chapter 3, “Defining MVS/CSC Startup Parameters” in *MVS/CSC Configuration Guide*.

DEFER

In the JES2 environment, regardless of the value you specify, the `DEFER` parameter is always set to `ON` for VTVs. That is, for VTVs, deferred mount processing is enabled, which overrides the mount processing specified in the user’s JCL. The VTV mount is deferred until the JCL job step opens a data set on the VTV. This value helps minimize VTV recalls. If a data set resides on a migrated VTV, VSM does not recall the VTV until the job actually opens the data set on the VTV.

In the JES3 environment, regardless of the value you specify, the `DEFER` parameter is always set to `JES3` for VTVs, which causes all mounts to be JES3 deferred. A volume is not mounted until a step begins execution.

Note that if a unit affinity chain includes a mixture of incompatible drives (including VTDs), NCS SMC device exclusion always ensures that the chain will be broken.

FETCH

`FETCH` specifies whether JES3 operator fetch message `IAT5110` is issued during VTD allocation.

MVS/CSC DISPLAY Command Enhancements

In a VSM configuration, the `DISPLAY LIBUNITS` command displays `VIRTUAL` for VTDs in the `Model` column. For more information about the `DISPLAY LIBUNITS` command, see Chapter 3, “Issuing MVS/CSC Operator Commands” of *MVS/CSC Operator’s Guide*. Note that you can use VTCS commands and reports on another host to produce additional VSM information.

MVS/CSC User Exit Enhancements

MVS/CSC User Exit SCSUX02 (JES2 and JES3 without TAPE setup environments), which you use to control transport allocation for scratch mounts, now supports return code `UX02VIRT` in register 15. SCSUX04 (JES3 with TAPE setup environment) also supports return code `UX04VIRT` in register 15. To satisfy a scratch mount request, these return codes cause VSM to select an available VTD in your system and route the data set to a VTV mounted on that VTD. In a multi-VTSS environment, therefore, these return codes do *not* direct the VTD allocation to a specific VTSS, but let the allocation occur in any VTSS in the configuration.

Information returned from SCSUX09 (JES2 and JES3 without TAPE setup environments) and SCSUX11 (JES3 with TAPE setup environment) applies to real transports only and is ignored for VTDs. VTD mounts are automatically deferred. For more information about MVS/CSC User Exits, see Chapter 8 or Chapter 9 in *MVS/CSC System Programmer’s Guide*.

MVS/CSC Programmatic Interface Enhancements

The SCSXREQM macro mappings are updated to support VSM as follows:

- The SCXVMED field can now display a value of VIRTUAL for VTVs.
- In the Volume Information Element, the formerly reserved field at decimal offset 24 is now an 8 byte character field with label SCXVTSSN. If SCXVMED is VIRTUAL, the volume is VTSS-resident, and MVS/ CSC controls the VTD in the VTSS in which the VTV resides, SCXVTSSN displays the VTSS name. If the VTV is migrated, SCXVTSSN is blank.
- The field SCXVLC is hexadecimal zero for a VTV.

For more information about the SCSXREQM macro, see Appendix A, “SCSXREQM Macro Mappings” of *MVS/CSC System Programmer’s Guide*.

MVS/CSC DELDISP Parameter Enhancements

MVS/CSC provides two new settings for the DELDISP startup parameter which is specified in a sequential file (usually a PDS member) at initialization:

ASCRTCH
: (All scratch). Both real tape volumes and VTVs are made scratch if they were mounted scratch and the disposition on the dismount message is delete ('D').

VSCRTCH
: (Virtual scratch). Only VTVs are made scratch if they were mounted scratch and the delete disposition on the dismount message is delete ('D').

The current DELDISP settings (SCRTCH and NOSCRTCH) define scratch handling at dismount **only** for real volumes. In an MVS/CSC system, if DELDISP is set to either of these values, VTVs are **never** scratched at dismount.

Each MVS/CSC system can define its own startup parameter file and can have different settings for DELDISP. A recycle of an MVS/CSC system is not necessary to change the DELDISP setting. The MVS/CSC ALTER command can change the setting for DELDISP; it accepts the two new settings. When changing the DELDISP setting via the ALTER command, it goes into affect immediately for that MVS/CSC system. However, if the MVS/CSC is recycled, the DELDISP setting is set to the value defined in the startup parameter file; if omitted it defaults to NOSCRTCH.

Appendix A. VTCS SMF Record Format

This appendix describes the formats of the HSC SMF record subtypes for VTCS events.

Note: In the record descriptions in this appendix, all generated timestamps, regardless of whether they are ttime or TOD values, are based on GMT time, not local time.

SLSSMF10 - VTCS SMF Subtype 10 Record

Function Records a VTSS subsystem performance request.

Table 19. SLSSMF10 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF10	VTCS SMF record subtype 10
0	0	character	8	SMF10VTS	VTSS ID
8	8	hexstring	2	SMF10BCH	base cache size (MB), where base cache is system space reserved for VTSS processing
10	A	hexstring	2	SMF10CCH	customer cache size (MB)
12	C	hexstring	4	SMF10OCH	offline cache size
16	10	hexstring	4	SMF10PCH	pinned cache size
20	14	hexstring	2	SMF10NSZ	nvs size (MB)
22	16	hexstring	8	SMF10TCT	reserved
30	1E	hexstring	8	SMF10TCP	total back end capacity
38	26	hexstring	8	SMF10FCT	reserved
46	2E	hexstring	8	SMF10FCP	total free back end capacity
54	36	hexstring	8	SMF10CFT	reserved
62	3E	hexstring	8	SMF10CFP	collected free back end capacity
70	46	hexstring	8	SMF10BRT	reserved
78	4E	hexstring	8	SMF10BRP	bytes read for free space collection
86	56	hexstring	8	SMF10SCT	reserved
94	5E	hexstring	8	SMF10SCP	total amount of free space collection
102	66	hexstring	2	SMF10RGC	redundancy group count
104	68	hexstring	8	SMF10CDT	reserved
112	70	hexstring	8	SMF10CDP	standard capacity defined
120	78	hexstring	4	SMF10EMP	count of ECAM-T messages processed
124	7C	hexstring	4	SMF10EBS	count of ECAM-T messages bypassed because no buffer space available
128	80	hexstring	4	SMF10EBC	count of ECAM-T messages bypassed because configuration was busy
132	84	hexstring	4	SMF10ECP	number of ECAM-T channel programs

SLSSMF11 - VTCS SMF Subtype 11 Record

Function Records a VTSS channel interface performance request.

Table 20. SLSSMF11 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF11	VTCS SMF record subtype 11
0	0	character	8	SMF11VTS	VTSS ID
8	8	hexstring	2	SMF11CNT	count of entries in this record the following fields repeat for each interface in this record
10	A	data		SMF11ENT	start of entry
10	A	character	8	SMF11INM	channel interface name
18	12	bitstring	2	SMF11CI	channel interface installed (y/n)
		X’0000’		SMF11CIN	no
		X’0001’		SMF11CIY	yes
20	14	bitstring	2	SMF11CE	channel interface enabled (y/n)
		X’0000’		SMF11CEN	no
		X’0001’		SMF11CEY	yes
22	16	hexstring	2	SMF11NAT	number of addresses trapped
24	18	hexstring	2	SMF11CSP	channel speed: 200 = 20 MB/sec ESCON channel
26	1A	hexstring	8	SMF11NIO	number of I/Os
34	22	hexstring	8	SMF11CUB	control unit busy (in υ -seconds)
42	2A	bitstring	2	SMF11TOL	type of link
		X’0000’		SMF11TLH	host
		X’0001’		SMF11TLR	RTD
44	2C	length		SMF11ENL	length of each entry

SLSSMF13 - VTCS SMF Subtype 13 Record

Function Records a VTV mount request.

Table 21. SLSSMF13 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF13	VTCS SMF record subtype 13
0	0	character	8	SMF13VTS	VTSS ID
8	8	character	6	SMF13VID	VTV volser ID
14	E	character	2	SMF13DID	VTD device ID
16	10	bitstring	2	SMF13RWS	read/write state (thumbwheel)
		X’0001’		SMF13RRO	read only
		X’0002’		SMF13RRW	read/write
18	12	bitstring	2	SMF13VMT	virtual mount type
		X’0001’		SMF13EXT	mount existing VTV
		X’0002’		SMF13SSL	mount sl scratch VTV
		X’0003’		SMF13SNL	mount existing VTV as scratch
		X’0004’		SMF13SAL	mount ANSI label scratch VTV
20	14	hexstring	4	SMF13TIM	VTV timestamp (ttime format, seconds since 1/1/70)
24	18	bitstring	2	SMF13RCI	recall indicator
		X’0001’		SMF13MNR	mounted without a recall
		X’0002’		SMF13MRC	mounted after a recall
26	1A	bitstring	2	SMF13CTP	cartridge type
		X’0000’		SMF13SCT	S-cart (max 400 Mb size)
		X’0001’		SMF13ECT	E-cart (max 800 Mb size)

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
28	1C	character	10		reserved
38	26	character	8	SMF13JNM	MSP jobname
46	2E	character	8	SMF13SNM	MSP stepname
54	36	character	44	SMF13DSN	MSP data set name
98	62	hexstring	8	SMF13MST	mount start timestamp (TOD), where mount start occurs when VTCS receives a mount request from HSC (or VTCS generates the request), generates a new thread to handle the mount request, then determines whether the request is for an existing, new, or scratch VTV
106	6A	hexstring	8	SMF13MET	mount end timestamp (TOD), where mount end occurs when VTSS generates a successful response to the ECAM-T request to mount the VTV on the selected RTD
114	72	character	8	SMF13MGT	VTV Management Class
122	7A	character	8	SMF13HST	Originating host name

SLSSMF14 - VTCS SMF Subtype 14 Record

Function Records a VTV dismount request.

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF14	VTCS SMF record subtype 14
0	0	character	8	SMF14VTS	VTSS ID
8	8	character	6	SMF14VID	VTV volser ID
14	E	bitstring	2	SMF14STA	VTV state
		X'0001'		SMF14MNT	VTV mounted
		X'0002'		SMF14DSM	VTV dismounted
		X'0003'		SMF14NON	VTV does not exist
		X'0004'		SMF14MIG	VTV is being migrated
		X'0005'		SMF14REC	VTV is being recalled
		X'0006'		SMF14VTM	VTV logically dismounted by `VTVMAINT`
16	10	hexstring	2	SMF14DID	MSP device address
18	12	hexstring	4	SMF14VSZ	uncompressed size of the VTV in bytes
22	16	hexstring	4	SMF14MSZ	the number of virtual tape pages in 32K increments required to migrate the VTV to an RTD
26	1A	hexstring	4	SMF14TIM	the last time the VTV was successfully mounted on a VTD (ttime format, seconds since 1/1/70)
30	1E	hexstring	2	SMF14UL#	number of MVCs to unlink
32	20	bitstring	2	SMF14CTP	cartridge type
		X'0000'		SMF14SCT	S-cart (max 400 Mb size)
		X'0001'		SMF14ECT	E-cart (max 800 Mb size)
34	22	bitstring	2	SMF14VMT	virtual mount type
		X'0001'		SMF14EXT	mount existing VTV
		X'0002'		SMF14SSL	mount sl scratch VTV
36	24	character	8	SMF14JNM	MSP jobname
44	2C	character	8	SMF14SNM	MSP stepname
52	34	character	44	SMF14DSN	MSP data set name
96	60	character	8	SMF14MGT	VTV Management Class
104	68	character	8	SMF14HST	Originating host name

Table 22. SLSSMF14 Record Format

SLSSMF15 - VTCS SMF Subtype 15 Record

Function Records a delete VTV request.

Table 23. SLSSMF15 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF15	VTCS SMF record subtype 15
0	0	character	8	SMF15VTS	VTSS ID (blanks if migrated)
8	8	character	6	SMF15VID	virtual volser ID
14	E	character	4	SMF15TIM	VTV creation time (ttime format, seconds since 1/1/70)
18	12		4		reserved
22	16	character	4	SMF15LTR	time VTV last referenced (high order TOD value)
26	1A	bitstring	2	SMF15CTP	cartridge type
		X’0000’		SMF15SCT	S-cart (max 400 Mb size)
		X’0001’		SMF15ECT	E-cart (max 800 Mb size)
28	1C		2		reserved
30	1E	bitstring	2	SMF15RSN	VTV delete reason code
		X’0001’		SMF15NMM	VTV migrated then deleted
		X’0002’		SMF15MPR	VTV previously migrated
		X’0003’		SMF15SPR	VTV reclaimed
		X’0004’		SMF15CON	VTV consolidated
		X’0005’		SMF15OLD	invalid VTV version found
		X’0006’		SMF15DSC	VTV deleted on scratch
		X’0007’		SMF15IMP	VTV deleted by import
32	20	character	8	SMF15MGT	VTV Management Class
40	28	bitstring	2	SMF15LRI	last residency indicator
		X’0001’		SMF15LRR	last residency was for recall
		X’0002’		SMF15LRC	last residency was for create

SLSSMF16 - VTCS SMF Subtype 16 Record

Function Records an RTD mount request.

Table 24. SLSSMF16 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SMF16VTS	VTCS SMF record subtype 16
0	0	character	8	SMF16VTS	VTSS ID
8	8	hexstring	2	SMF16RID	RTD ID (0-F)
10	A	character	6	SMF16MID	MVC volser ID
16	10	character	6	SMF16AID	actual volser from VOL1 label
22	16	bitstring	2	SMF16RWS	read/write state (thumbwheel)
		X’0001’		SMF16RRO	read only state
		X’0002’		SMF16RRW	read/write state
24	18	bitstring	2	SMF16MT	mount request type
		X’0001’		SMF16MTM	migrate
		X’0002’		SMF16MTR	recall
		X’0003’		SMF16MTL	reclaim
		X’0004’		SMF16MTD	drain
		X’0005’		SMF16MTA	audit
		X’0006’		SMF16MTC	consolidate
		X’0007’		SMF16MTX	export
26	1A	hexstring	32	SMF16SNS	RTD sense data (all zeros or all X’FF’s unless RTD errors occur)
58	3A	hexstring	8	SMF16MST	mount start timestamp (TOD), where mount start occurs when HSC receives a successful request to load the requested MVC
66	42	hexstring	8	SMF16MET	mount end timestamp (TOD), where mount end occurs when the VTSS receives a successful ECAM-T request to mount the requested MVC on an RTD
74	4A	character	8	SMF16SCL	MVC Storage Class

SLSSMF17 - VTCS SMF Subtype 17 Record

Function Records an RTD dismount request.

Table 25. SLSSMF17 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF17	VTCS SMF record subtype 17
0	0	character	8	SMF17VTS	VTSS ID
8	8	hexstring	2	SMF17RID	RTD ID (0-F)
10	A	hexstring	64	SMF17BLD	RTD buffered log data
74	4A	hexstring	32	SMF17SNS	RTD sense data (all zeros or all X'FF's unless RTD errors occur)
106	6A	character	8	SMF17SCL	MVC Storage Class
114	72	character	6	SMF17MVC	MVC volser

SLSSMF18 - VTCS SMF Subtype 18 Record

Function Records a migrate VTV request.

Table 26. SLSSMF18 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF18	VTCS SMF record subtype 18
0	0	character	8	SMF18VTS	VTSS ID
8	8	hexstring	2	SMF18RID	RTD ID (0-F)
10	A	character	6	SMF18VID	VTV volser ID
16	10	character	6	SMF18MID	MVC volser ID
22	16	hexstring	4	SMF18VPO	VTV position on this MVC (block ID)
26	1A	character	6	SMF18AID	actual volser from VOL1 label
32	20	hexstring	4	SMF18MSZ	uncompressed size of the VTV in bytes
36	24	hexstring	4	SMF18BCM	the number of virtual tape pages in 32K increments required to migrate the VTV to an RTD
40	28	hexstring	4	SMF18TIM	the last time the VTV was successfully mounted on a VTD (ttime format, seconds since 1/1/70)
44	2C	bitstring	2	SMF18MT	migrate request type
		X’0001’		SMF18MTA	auto
		X’0002’		SMF18MTI	immediate
		X’0003’		SMF18MTD	demand
		X’0004’		SMF18MTR	reclaim
		X’0005’		SMF18MTC	consolidate
		X’0006’		SMF18MTX	export
46	2E	bitstring	2	SMF18CTP	cartridge type
		X’0000’		SMF18SCT	S-cart (max 400 Mb size)
		X’0001’		SMF18ECT	E-cart (max 800 Mb size)
48	30	hexstring	4	SMF18NPO	next MVC position (block ID)
52	34	hexstring	32	SMF18SNS	RTD sense
84	54	hexstring	8	SMF18MST	migrate start timestamp (TOD)
92	5C	hexstring	8	SMF18MET	migrate end timestamp (TOD)

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
100	64	character	8	SMF18MGT	VTV Management Class
108	6C	character	8	SMF18SCL	MVC Storage Class

SLSSMF19 - VTCS SMF Subtype 19 Record

Function Records a recall VTV request.

Table 27. SLSSMF19 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SMF19VTS	VTCS SMF record subtype 19
0	0	character	8	SMF19VTS	VTSS ID
8	8	hexstring	2	SMF19RTD	RTD ID (0-F)
10	A	character	6	SMF19VID	VTV volser ID
16	10	character	6	SMF19MID	MVC volser ID
22	16	hexstring	4	SMF19VPO	VTV position on this MVC (block ID)
26	1A	bitstring	2	SMF19RE	recall with error
		X’0000’		SMF19REN	no
		X’0001’		SMF19REY	yes
28	1C	character	6	SMF19AID	actual volser from VOL1 label
34	22	hexstring	4	SMF19MSZ	VTV media size
38	26	hexstring	4	SMF19BCM	number of bytes currently recalled
42	2A	hexstring	4	SMF19TIM	the last time the VTV was successfully mounted on a VTD (ttime format, seconds since 1/1/70)
46	2E	bitstring	2	SMF19MT	recall request type
		X’0001’		SMF19MTA	auto
		X’0002’		SMF19MTN	drain
		X’0003’		SMF19MTD	demand
		X’0004’		SMF19MTR	reclaim
		X’0005’		SMF19MTC	consolidate
		X’0006’		SMF19MTX	export
48	30	bitstring	2	SMF19CTP	cartridge type
		X’0000’		SMF19SCT	S-cart (max 400 Mb size)
		X’0001’		SMF19ECT	E-cart (max 800 Mb size)
50	32	hexstring	32	SMF19SNS	RTD sense
82	52	hexstring	8	SMF19RST	recall start timestamp (TOD)
90	5A	hexstring	8	SMF19RET	recall end timestamp (TOD)
98	62	character	8	SMF19MGT	VTV Management Class
106	6A	character	8	SMF19SCL	MVC Storage Class

SLSSMF20 - VTCS SMF Subtype 20 Record

Function

Records an RTD performance request.

Table 28. SLSSMF20 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF20	VTCS SMF record subtype 20
0	0	character	8	SMF20VTS	VTSS ID
8	8	hexstring	2	SMF20CNT	count of entries in this record the following fields repeat for each RTD in this record
10	A	area		SMF20ENT	start of entry
10	A	character	8	SMF20RNM	RTD name
18	12	bitstring	2	SMF20ST	RTD state
		X’0000’		SMF20STU	unconfigured
		X’0001’		SMF20STC	configured
20	14	hexstring	8	SMF20ATM	device available time (υ -seconds), which is the time the MVC is mounted on the RTD
28	1C	hexstring	8	SMF20ACT	device activity (initial selects)
36	24	hexstring	8	SMF20BTR	bytes transferred - read
44	2C	hexstring	8	SMF20BTW	bytes transferred - write
52	34	hexstring	8	SMF20DUT	device utilization time (υ -seconds), which is the accumulated time of each CCW chain to device end
60	3C	hexstring	8	SMF20DCT	device connect time (υ -seconds), which is the accumulated time of each CCW chain to device end

SLSSMF21 - VTCS SMF Subtype 21 Record

Function Records a vary RTD.

Table 29. SLSSMF21 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF21	VTCS SMF record subtype 21
0	0	character	8	SMF21VTS	VTSS ID
8	8	hexstring	2	SMF21RTD	RTD ID (0-F)
10	A	bitstring	2	SMF21STA	new device state
		X’0001’		SMF21OFF	offline
		X’0002’		SMF21ON	online
		X’0003’		SMF21MAI	maintenance

SLSSMF25 - VTCS SMF Subtype 25 Record

Function Records MVC status.

Table 30. SLSSMF25 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF25	VTCS SMF record subtype 25
0	0	character	6	SMF25MID	MVC volser ID
6	6	hexstring	4	SMF25TFS	total free space (includes any space for invalid VTVs)
10	A	hexstring	4	SMF25UFS	usable free space (after the last valid VTV on the MVC)
14	E	hexstring	4	SMF25NAV	number of active VTVS
18	12	character	8	SMF25SCL	MVC Storage Class
26	1A	hexstring	4	SMF25TUS	space in Kb used by current VTVs
30	1E	hexstring	4	SMF25NDV	number of "holes" (deleted VTVs)
34	22	hexstring	4	SMF25LUT	top 4 bytes of the TOD clock when the MVC was last used
38	26	hexstring	4	SMF25LWT	top 4 bytes of the TOD clock when the MVC was last updated

SLSSMF26 - VTCS SMF Subtype 26 Record

Function Records VTV movement.

Table 31. SLSSMF26 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF26	VTCS SMF record subtype 26
0	0	character	8	SMF26VTS	VTSS ID
8	8	character	6	SMF26VID	VTV volser ID
14	E	character	6	SMF26OMI	old MVC volser ID
20	14	character	6	SMF26NMI	new MVC volser ID
26	1A	hexstring	4	SMF26VPO	VTV position on new MVC (block ID)
30	1E	hexstring	8	SMF26MST	move start timestamp (TOD)
38	26	hexstring	8	SMF26MET	move end timestamp (TOD)
46	2E	character	8	SMF26MGT	VTV Management Class

SLSSMF27 - VTCS SMF Subtype 27 Record

Function Records VTV scratch status.

Table 32. SLSSMF27 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF27	VTCS SMF record subtype 27
0	0	character	6	SMF27VID	VTV volser ID
6	6	character	8	SMF27MCL	VTV Management Class
14	E	bitstring	2	SMF27STP	VTV scratch type
		X’0001’		SMF27STN	no delete on scratch
		X’0002’		SMF27STD	delete on scratch
16	10	hexstring	4	SMF27MSZ	VTV media size
20	14	hexstring	4	SMF27TIM	the last time the VTV was updated (ttime format, seconds since 1/1/70)
24	18	hexstring	4	SMF27LUS	the last time the VTV was used (TOD format)
28	1C	hexstring	6	SMF27MV1	volser of MVC 1 that contains the VTV
34	22	hexstring	6	SMF27MV2	volser of MVC 2 that contains the VTV
40	28	character	8	SMF27VTS	VTSS name
48	30	bitstring	1	SMF27RES	VTV last resident indicator
			X’80’	SMF27RVT	resident on VTSS
			X’40’	SMF27RM1	resident on MVC1
			X’20’	SMF27RM2	resident on MVC2
			X’10’	SMF27RM3	resident on MVC3
			X’08’	SMF27RM4	resident on MVC4
49	31		1	SMF27SPR	reserved
50	32	bitstring	2	SMF27CTP	cartridge type
		X’0000’		SMF27SCT	S-cart (max 400 Mb size)
		X’0001’		SMF27ECT	E-cart (max 800 Mb size)
52	34		4	SMF27SP2	reserved
56	38	character	6	SMF27MV3	volser of MVC3 that contains the VTV
62	3E	character	6	SMF27MV4	volser of MVC4 that contains the VTV

SLSSMF28 - VTCS SMF Subtype 28 Record

Function Records a VTV replication.

Table 33. SLSSMF28 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF28	VTCS SMF Record sub-type 28
0	0	character	8	SMF28VTS	Primary VTSSname
8	8	character	8	SMF28SVT	Secondary VTSSname
16	10	character	8	SMF28CLN	Cluster Name
24	18	character	6	SMF28VID	VTV Volser
30	1E	hexstring	2	SMF28AID	CLINK CHANIF
32	20	hexstring	1	SMF28DID	CLINK device-id
33	21	hexstring	1		reserved
34	22	bitstring	2	SMF28CTP	cartridge type
		X’0000’		SMF28SCT	S-cart (max 400 Mb size)
		X’0001’		SMF28ECT	E-cart (max 800 Mb size)
36	24	hexstring	4		reserved
40	28	hexstring	4	SMF28BCR	Bytes replicated for VTV
44	2C	hexstring	4	SMF28TIM	VTV last updated timestamp (seconds since 1/1/70)
48	30	hexstring	32	SMF28SNS	Sense data from CLINK
80	50	hexstring	8	SMF28RST	Replicate Start Time (TOD format)
88	58	hexstring	8	SMF28RET	Replicate End Time (TOD format)
96	60	hexstring	8	SMF28MGT	VTV Management Class

SLSSMF29 - VTCS SMF Subtype 29 Record

Function Records a VTV and MVC unlink event.

Table 34. SLSSMF29 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF29	VTCS SMF Record sub-type 29
0	0	character	6	SMF29VID	VTV volser
6	6	character	6	SMF29MVC	MVC volser
12	C	character	2	SMF29MV#	number of remaining MVCs
14	E	bitstring	2	SMF29RSN	reason for unlink
		X’0001’		SMF29NLC	VTV no longer current (dismount)
		X’0002’		SMF29DRN	MVC drain/reclaim
		X’0003’		SMF29DOS	delete on scratch
		X’0004’		SMF29IMP	VTV import
		X’0005’		SMF29VMN	VTVMAINT utility
		X’0006’		SMF29MVC	MVC inventory
		X’0007’		SMF29VTS	VTSS inventory
		X’0008’		SMF29VAD	VTV audit
16	10		6		reserved

SLSSMF30 - VTCS SMF Subtype 30 Record

Function Records a Vary Clink event.

Table 35. SLSSMF30 Record Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0	start of record		SLSSMF30	VTCS SMF Record sub-type 30
0	0	character	8	SMF30CLU	Cluster name
8	8	character	8	SMF30VTS	Primary or Sending VTSS name
16	10	character	8	SMF30PAR	Partner VTSS name
24	18	bitstring	2	SMF30CLI	Clink Id
26	1A	bitstring	2	SMF30STA	New Clink status
		X’0001’		SMF30OFF	Offline
		X’0002’		SMF30ON	Online

Appendix B. VTD Command Reference

Overview

This appendix describes *only* that portion of the Virtual Tape Storage Subsystem (VTSS) that responds to 3490 style tape commands. This appendix is intended to be a programmer's guide for defining how the VTD behaves.

This appendix is meant to be read in conjunction with the *IBM 3490 Magnetic Tape Subsystem Models A01, A02, A10, A20, B02, B04, B20, and B40 Hardware Reference* (referred to in this book as the *IBM 3490 Hardware Reference*) and as such is not intended to be a stand-alone document. This appendix contains only that functionality which is different from the *IBM 3490 Hardware Reference*. Please review that document before reading this appendix.

Bit Naming Conventions

Please be aware that the IBM convention of numbering bits has been used within this document. This means bit 0 is the most significant bit of a unit (byte, word, etc.)to be described and that the max bit N is the least significant bit.

Discussion of 3480/3490 Terms

This section expands upon IBM terms such as tape products, media, and media formats as they relate to VTDs. It is only important in the fact that the VTSS must respond accurately to the commands described within this document.

See "Product and Media Type Emulation" on page 283 for how the VTD responds in command data.

Table 36. Product Designators and Capabilities

Product or Model Designators	Commands Respond as...	18 track capable	IDRC capable	36 track capable	long tape capable
3480 B02 or B04	3480 02 or 04	yes	opt		
3490 B02 or B04	3480 02 or 04	yes	std		
3490 B20 or B40 (a.k.a. 3490E)	3490 20 or 40	read only	yes	yes	yes

Table 37. Media and Media capabilities

Media Name	with this Model	can support 18 track	can support IDRC	can support 36 track	can support long tape
3480	3480 B02 or B04	yes	opt.		
3480	3490E	read only	yes	yes	

Media Name	with this Model	can support 18 track	can support IDRC	can support 36 track	can support long tape
3490E “Enhanced” i.e. long tape	3490E	NO	yes	yes	yes

Table 38. ***Formatted Media Designators***

if IDRC blocks exist	if 36 track	then, media is known to be formatted as a
-	-	3480
yes	-	3480 XF
yes	yes	3480-2 XF
-	yes	3480 XF

Summary:

1. An “E” in a product designator always has IDRC, 36 track, and long tape capability (using 3490E media). The E in a product is the same as a 3490E or B20 or B40 product capability
2. An “E” in the media designator (a.k.a. 3490E media) always means long tape.
3. All 36 track machines always have IDRC capability
4. A 3490 B02 or B04 is a repackaged 3480 B02 OR B04 but responds in a command as a 3480. The B02 refers to 2 drives in a frame. The B04 refers to 4 drives in a frame.
5. The B20 refers to 2 drives in a frame. The B40 refers to 4 drives in a frame.

Command Overview

Characteristics of the VTD

Although the Virtual Tape Device (VTD) emulates a 3490 type of device, it does have certain characteristics that differentiate it from that of a physical tape I/O system.

Defects. Data written to and recalled from virtual tape volumes are free of defects and gaps. Once this data is migrated to and recalled from real tape, the possibility of defects arises. If the VTSS encounters a data error recalling a volume from real tape, that error will be preserved when the data is eventually read by the host.

IDRC always Enabled. Data is *always* compressed when it is written to the VTSS and then uncompressed when it is read back to the channel. It is not possible to disable IDRC within the VTSS.

Product and Media Type Emulation

Throughout the descriptions of command data, the VTD responds as described in Table 39. The Virtual Tape Volume (VTV) media always appears as a 400 MB (compressed) 36 track 3480 style cartridge. Command data refers to this as a 3480-2 XF format.

The amount of customer data that can be compressed into 400 MB varies with the data, but compression in the VTSS is better than that of standard IDRC rates.

Refer to “Discussion of 3480/3490 Terms” on page 281 for a discussion of what these terms mean.

Table 39. VTSS Type Emulation

	command response
Control Unit type	3490 A20
Tape Unit type	3490 B40
Media type	3480 (standard)
Media formatted as...	3480-2 XF

The commands that return this data include the Read Buffered Log, Read Subsystem Data, Read Device Characteristics, Sense ID and Read Configuration Data commands.

Command Summary

Table 40 represents a summary of commands described in the *IBM 3490 Command Reference* manual. They are grouped by their type along with their command codes. This table also indicates whether this command operates differently than that described in the *IBM 3490 Command Reference* manual. For those commands that are different, a unique command description is contained later on in this document.

***Table 40.* Command Difference Summary**

Command	Mnemonic	Hex	Cmd Differences	Typical Functionality
DATA TRANSFER COMMANDS				
Write	WRT	01	Yes	Only command that writes user data to tape.
Read Forward	RDF	02	Yes	Reads data in the forward direction
Read Backward	RDB	0C	Yes	Reads data in the backwards direction
Read Buffer	RBF	12	Yes	Returns any non-written buffered device data
TAPE CONTROL COMMANDS				
Rewind	REW	07	-	Positions the tape to the beginning of tape
Rewind Unload	RUN	0F	-	Rewinds and unloads the tape
Erase Gap	ERG	17	Yes	Used in error handling
Write Tape Mark	WTM	1F	-	Writes a tape mark on tape
Backward Space Block	BSB	27	-	Positions backward past the previous tape mark or block
Backward Space File	BSF	2F	-	Positions backward past the previous tape mark
Forward Space Block	FSB	37	-	Positions to the next tape mark or block
Forward Space File	FSF	3F	-	Positions to the next tape mark
Data Security Erase	DSE	97	Yes	Writes a random pattern from current position to the end of tape
DEVICE MANAGEMENT COMMANDS				
Test I/O	TIO	00	-	Not used in ESCON attached CUs
No-Operation	NOP	03	-	Indicates device readiness or returns pending status
Channel Path No-Operation	CPNOP	13	-	Indicates channel path readiness or returns pending status

Command	Mnemonic	Hex	Cmd Differences	Typical Functionality
Read Block ID	RBID	22	Yes	Returns current positional information
Synchronize	SYNC	43	Yes	Flushes buffered write data to device
Locate Block	LOC	4F	Yes	Positions the tape at the desired block ID
Load Display	LDD	9F	Yes	Controls the drive display panel; Also provides ECAM-T handshake.
Set Tape Write Immediate	TWI	C3	Yes	Controls the immediate mode of operation
3420 Compatibility Commands	-	CB D3	Yes	Not supported
Mode Set	MDS	DB	Yes	Controls various modes of operations
SUBSYSTEM COMMANDS				
Sense	SNS	04	-	refer to Sense format descriptions
Read Buffered Log	RBL	24	Yes	Returns statistics on the current device and volume
Read Subsystem Data	RSSD	3E	Yes	Returns data indicated by preceding PSF or Set Interface ID command
Read Message ID	RMID	4E	Yes	Returns status of asynchronous operations
Read Device Characteristics	RDC	64	Yes	Returns model and device characteristics
Set Interface Identifier	SII	73	Yes	With Read Subsys Data command following this command, NDs and NQs are returned (1)
Perform Subsystem Function	PSF	77	Yes	Performs various operations. One of which initiates asynchronous operations.
Sense ID	SNSID	E4	Yes	Returns mode, type and CIWs (2)
Read Configuration Data	RCD	FA	Yes	Returns various NED's and NEQs for the subsystem (1)
PATH MANAGEMENT COMMANDS				
Sense Path Group ID	SNID	34	(3)	Returns pathing information
Suspend Multipath Reconnection	SMR	5B	(3)	Suspends multipath reconnections

Command	Mnemonic	Hex	Cmd Differences	Typical Functionality
Set Path Group ID	SPID	AF	(3)	Controls grouping of paths and devices to a controlling computer
Assign	ASN	B7	(3)	Assigns the device to a channel path ID
Unassign	UNA	C7	(3)	Unassigns the device from a channel path ID
Control Access	CAC	E3	(3)	Provides password protected overrides of Assign facility

Notes:
(1) For more information, see *ESA/390 Common I/O-Device Commands and Self Description*. The contents of the VTSS's NEDs, NEQs, NDs, and NQs may vary slightly from that of an actual 3490 subsystem.
NED - Node Element Descriptor - contains data such as type of device, model, manufacturer, plant, serial-number.
NEQ - Node Element Qualifier -
ND - Node Descriptor - similar but different than NEDs
NQ - Node Qualifier - similar but different than NEQs
(2) CIW - Channel Information Word - Identifies to channel which commands to use for Read Configuration, Set Interface Identifier, Read Node Identifier
(3) Functions identically to an IBM 3490E tape drive, whose functionality differs from the commands' descriptions in the *IBM 3490E Hardware Reference*. See "Path Management Commands" on page 303.

Command Dependent Unit-Checks

There are very few changes to Command Dependent Unit-Checks described in the *IBM 3490E Hardware Reference*.

Any command used in an active ECAM-T chain is exempt from the “Assigned Elsewhere” unit Check.

Any command issued to an unconfigured virtual tape address can receive a “Device Offline” Unit-Check.

Virtual Not-Ready

As with a real 3490E, when there is no volume (VTV) mounted on a virtual tape, the unit can return a “Not Ready” Unit-Check to all eligible commands. While a real 3490E can also return a “Not Ready” Unit-Check for a mounted volume when the operator has toggled a Not-Ready state by depressing the “READY” key on the unit, no such manual facility exists for a VTSS virtual tape unit.

However, when an “Out-of-Capacity” condition exists in a VTSS, any Write command issued to a VTV mounted on that VTSS will create a “Virtual Not-Ready” condition at that virtual tape unit. The initial Write command will receive a “Drive Switched Not-Ready” Unit-Check, and all subsequent eligible commands will receive a “Not Ready” Unit-Check until the “Out-of-Capacity” condition is handled (by VTV migration or deletion).

When the “Out-of-Capacity” condition no longer exists, the VTSS will signal “Attention/Ready” to all paths which received a “Not Ready” Unit-Check.

In all aspects, the VTSS “Virtual Not-Ready” condition appears identical to an operator toggle of the “READY” key on a real 3490E.

Command Dependent Execution Status

Table 41 on page 288 shows the normal status that is generated for the commands as a result of command acceptance and execution. The VTSS status is different from that of a physical tape I/O subsystem

Note that the VTSS does not disconnect from the channel using CE only status. It always uses CCR (Disconnect Channel Command Retry) to disconnect. As a result, the possibility of a third status does not exist.

Note: Whenever the term CCR is used within this specification, it always refers to Disconnect Channel Command Retry (represented by a hex 4A). In ESCON I/O specifications this is also referred to as Deferred Command Retry.

When a CCRed command results in Unit Check or Unit Exception, the UC or UE indication can be presented combined with Device End in the final status. This will result in 0x06 or 0x05 final status instead of the 0x0E or 0x0D seen on the 3490e.

The column labeled CCR describes those conditions unique to this command for using CCR to disconnect. Those reasons are as follows.

b

The necessary buffer and data for performing the operation is not available.

p

The necessary data transfer path for performing the operation is not available.

Table 41. ***Command Dependent Execution Status***

Command	Mnem-onic	Hex value	Initial Status	Second Status	CCR b p	Other Status
DATA TRANSFER COMMANDS						
Write	WRT	01	CMR	CE+DE	y y	UC, UE
Read Forward	RDF	02	CMR	CE+DE	y y	UC,UE
Read Backward	RDB	0C	UC	-	- -	-
Read Buffer (no data is ever transferred)	RBF	12	CMR	CE+DE	- -	-
TAPE CONTROL COMMANDS						
Rewind	REW	07	CE+DE	-	- -	UC
Rewind Unload	RUN	0F	CE+DE	-	y -	UC
Erase GAP	ERG	17	CE+DE	-	- -	-
Write Tape Mark	WTM	1F	CE+DE	-	y -	UC, UE
Backward Space Block	BSB	27	CE+DE	-	y -	UC, UE
Backward Space File	BSF	2F	CE+DE	-	y -	UC
Forward Space Block	FSB	37	CE+DE	-	y -	UC, UE
Forward Space File	FSF	3F	CE+DE	-	y -	UC
Data Security Erase	DSE	97	CE+DE	-	y -	UC
DEVICE MANAGEMENT COMMANDS						
Test I/O	TIO	00	not issued to ESCON devices			
No-Operation	NOP	03	CE+DE	-	- -	UC
Channel Path No-Operation	CPNOP	13	CE+DE	-	- -	UC
Read Block ID	RBID	22	CMR	CE+DE	- -	UC

Command	Mnem-onic	Hex value	Initial Status	Second Status	CCR b p	Other Status
Synchronize	SYNC	43	CE+DE	--	- -	UC
Locate Block	LOC	4F	CMR	CE+DE	y -	UC
Load Display	LDD	9F	CMR	CE+DE	- -	UC
Set Tape Write Immediate	TWI	C3	CE+DE	-	- -	UC
Mode Set	MDS	DB	CMR	CE+DE	y -	UC
SUBSYSTEM COMMANDS						
Sense	SNS	04	CMR	CE+DE	- -	UC
Read Buffered Log	RBL	24	CMR	CE+DE	- -	UC, UE
Read Subsystem Data	RSSD	3E	CMR	CE+DE	- -	UC
Read Message ID	RMID	4E	CMR	CE+DE	- -	-
Read Device Characteristics	RDC	64	CMR	CE+DE	- -	UC
Set Interface Identifier	SII	73	CMR	CE+DE	- -	UC
Perform Sub-system Function	PSF	77	CMR	CE+DE	y -	UC
Sense ID	SNSID	E4	CMR	CE+DE	- -	UC
Read Configura-tion Data	RCD	FA	CMR	CE+DE	- -	UC
PATH MANAGEMENT COMMANDS						
Sense Path Group ID	SNID	34	CMR	CE+DE	- -	UC
Suspend Multi-path Reconnec-tion	SMR	5B	CE+DE	-	- -	UC
Set Path Group ID	SPID	AF	CMR	CE+DE	- -	UC
Assign	ASN	B7	CMR	CE+DE	- -	UC
Unassign	UNA	C7	CMR	CE+DE	- -	UC
Control Access	CAC	E3	CMR	CE+DE	- -	UC

<table>
<tr><th>Command</th><th>Mnemonic</th><th>Hex value</th><th>Initial Status</th><th>Second Status</th><th>CCR b p</th><th>Other Status</th></tr>
<tr><td colspan="7">Notes: For many of the immediate commands, IBM documents them as returning CE in initial status to be followed by DE. The VTSS has the capability to present CE+DE in initial status when it can immediately execute the command. This behavior is the same as the Timberline Tape Subsystem.</td></tr>
</table>

Retry Status

The VTSS does not issue command retry for various buffer synchronization scenarios that exist in typical tape I/O systems for buffer management, read to write switches, direction changes, and so forth.

The VTSS, however, does issue command retry when either:

- The necessary buffer and data for performing the operation is not available.
- The necessary data transfer path for performing the operation is not available.

Table 41 on page 288 details when these are issued for each command.

The VTSS also uses command retry to temporarily disconnect from the interface when the CU must perform certain recovery scenarios. These scenarios can occur at any time and will occur on *any* command. These scenarios arise when the VTSS must perform a restart of its internal microcode.

Differentiated Channel Command Descriptions

Data Security Erase Command

On physical tape I/O subsystems, the Data Security Erase command writes a random pattern from the position of the tape where the command is issued to the physical end of tape.

Due to the manner in which VTSS can store multiple versions of a given tape volume on various Multi-Volume-Cartridges (MVCs), it is not practical to perform an erase on these various MVCs. As such this command does not perform a physical erase of previous versions of the volume. It does, however, perform an erase of data from the current position on the tape to the end of the virtual tape. The erase is performed logically with no actual writing of data.

Erase Gap Command

The Erase Gap command writes a unique erase gap pattern on physical tape for the purposes of skipping over media.

Writes to VTSS have no such media problems, and as such this command performs no real function other than effecting a read-to-write change.

Load Display Command

For normal tape subsystems, this command controls the text of messages on the message display of the selected device. It also controls the operation of an automatic cartridge loader.

The VTSS has neither a message display nor an automatic cartridge loader. As such, this command is accepted but has no standard effect on the control unit. If the “Index Automatic Load” function-byte bit is set, a Command Reject results.

However, the Load Display command IS used by the VTSS for a special purpose: to signal that this channel program follows the ECAM-T protocol for special host-to-VTSS communications.

Note that the Assigned-Elsewhere Unit-Check does not apply to an LDD used for the “ECAM-T Handshake”; for the non-ECAM-T case it is a low-priority check.

Locate Block

The Locate Block command moves the tape into position on the addressed tape drive so the controlling computer can write or read a specific block on the tape. For normal tape I/O systems, the search first occurs on the segment or physical level and then proceeds to search for the Logical Block by reading data. For the VTSS, the search is performed logically using table lookup means and does not suffer from any typical media vagaries such as defects.

Mode Set Command

The Mode Set command controls specific aspects of command processing within a given command chain. The VTSS accepts this command but has the following characteristics.

- If the parameters specify that Tape-Write-Immediate Mode be turned off, it will be ignored because the VTSS always operates in non-buffered mode.
- If the parameters specify that IDRC be turned off, it too will be ignored because the VTSS always attempts to compress data - it cannot be disabled.
- The Supervisor Inhibit bit is implemented as described.

Perform Subsystem Function Command

The actual functionality of the Perform Subsystem Function command is determined by an order byte (byte 0 of the data). Only a few of these orders have any real functionality, and they are documented in the Table 42. If no support is provided for an order, the order and the following data is accepted and checked for parameter compliance. If the parameters are not legal, the command is rejected with ERA 27 (Command Reject). If the parameters are correct, the command is unit checked with ERA 29 (Function Not Compatible).

Table 42. Perform Subsystem Function orders

Order (hex)	Description	Support
18	Prepare for Read Subsystem Data	No
1B	Set Special Intercept Condition	No
1C	Message Not Supported	No
80	Activate Forced Error Logging	No
81	Deactivate Forced Error Logging	No
82	Activate Access Control	Yes
83	Deactivate Access Control	Yes
90	Reset Volume Fence	No
A1	Pin Device	No
A2	Unpin Device	No

Table 43. Perform Subsystem Function data

Byte	Description
0	Perform Subsystem Function order (refer to prior table)
1	Perform Subsystem Function flag
2-N	Order parameters (refer to following sections for parameters)

Activate Access Control Order

This order allows the control program to activate various access control mechanisms for the device. For any bit that is set to 1 that the VTSS supports, that feature is activated until another Activate or Deactivate Access Control Order is received OR the volume is unloaded.

Table 44. ***Activate Access Control Order parameters***

Byte	Bit	Description
0		Order, set to x80
1		Flag byte, set to 0
2	0	Logical Write Protect
	1-2	Reserved, set to 0
	3	Data Compaction Default - NOT SUPPORTED The VTSS *always* writes data using Compaction. It is not possible to disable this functionality. This bit is ignored.
	4-5	Reserved, set to 0
	6	Data Check Recovery - NOT SUPPORTED This bit is ignored.
	7	Extended Recovery - NOT SUPPORTED This bit is ignored.
3		Reserved, set to 0

Deactivate Access Control Order

This order allows the control program to *de*-activate various access control mechanisms for the device. For any bit that is set to 1 that the VTSS supports, that feature is *de*-activated until another Activate or Deactivate Access Control Order is received OR the volume is unloaded.

Table 45. ***De-activate Access Control Order parameters***

Byte	Bit	Description
0		Order, set to x81
1		Flag byte, set to 0
2	0	Logical Write Protect
	1-2	Reserved, set to 0
	3	Data Compaction Default - NOT SUPPORTED The VTSS *always* writes data using Compaction. It is not possible to disable this functionality. This bit is ignored.
	4-5	Reserved, set to 0
	6	Data Check Recovery - NOT SUPPORTED This bit is ignored.
	7	Extended Recovery - NOT SUPPORTED This bit is ignored.
3		Reserved, set to 0

Read Backwards Command

A Read Backward command causes data for one logical block to be transferred from the control unit to the channel. The logical block transferred is the next sequential logical block in the backward direction from the program's perspective. The data is transferred in reverse order (last byte first, first byte last).

The VTSS does not directly support this command. Instead, it always positions in front of the desired block, and if that block is not a tape mark, returns unit check status with an ERA code of 26 (Read Opposite) which forces the IOS driver to issue a Read Forward command followed by a Back Space Block command. If the desired block was a tape mark, the VTSS returns unit exception status indicating that end-of -file has been encountered.

Read Block ID Command

The Read Block ID command transfers 8 bytes of information from the control unit to the channel that identifies the next block to be read or written in the forward direction. For buffered physical tape I/O, the data returned identifies positional data for the channel side of the buffer and the physical side of the buffer.

Since VTSS tape I/O always operates in a non-buffered mode, the positional data for the channel side and physical side of the buffer are always the same. The positional data is fully emulated for a 400 MB cartridge.

Read Buffer Command

The Read Buffer command transfers data from the control unit to the channel if any buffered write data is in the control unit's buffer. This data is retrieved via this command typically when the tape CU is unable to write the data to the media.

But since all write data to the VTSS is held in non-volatile storage until it can be moved to either back-end RAID or physical type devices, write data is always immediately committed to permanent storage and as such no data is ever in the "buffer".

On receipt of this command, "no data available" or "buffer empty" is indicated by transferring zero bytes.

Read Buffered Log Command

For VTSS, the Read Buffered Log command always transfers 64 bytes (Format 30) of buffered log data from the control to the channel. Format 30 is presented since compression is always enabled. Refer to "Status and Sense Bytes" on page 304 for a description of the buffered log data. Note that as with a 3490, if all log counters are zero, the VTSS returns unit exception status.

Read Configuration Data Command

A Read Configuration Data command causes 160 bytes of data to be transferred from the control unit to the channel. The specific data returned by the VTSS is defined in Table 46.

***Table 46.* Read Configuration Data Command data**

Byte	Bit	Description
DEVICE NED		
0		Flags = 1100 1100
1		Type = x01 (I/O Device)

Byte	Bit	Description
2		Class = x02 (Magnetic Tape)
3	0-6	0
	7	Level = 0 (no hierarchic Relationship)
4-9		Type Number = xF0F0F3F4F9F0 (EBCDIC for 003490)
10-12		Model Number = xC2F4F0 (EBCDIC for B40)
13-15		Manufacturer = xE2E3D2 (EBCDIC for STK)
16-17		Plant of Manufacture
18-20		Family code which identifies this subsystem as a VTSS (EBCDIC)
21-29		Sequence Number in EBCDIC
30-31		Tag = x00XX where XX=VTape Address (0-F) (xFF & VDID)
CONTROL UNIT NED		
32		Flags = 1101 0100 (or 1100 1100)
33		Type = x02 (Control Unit)
34		Class = x00 (Unspecified class)
35	0-6	0
	7	Level = 0 (no hierarchic Relationship)
36-41		Type Number = xF0F0F3F4F9F0 (EBCDIC for 003490)
42-44		Model Number = xC1F2F0 (EBCDIC for A20)
45-47		Manufacturer = xE2E3D2 (EBCDIC for STK)
48-49		Plant of Manufacture
50-51		Family code which identifies this subsystem as a VTSS (EBCDIC)
52-61		Lower bytes of the Sequence Number in EBCDIC
62-63		Tag = x00XX where XX = Control-Unit-Image-# (VCU/Cluster) ... = ((xC0 & VDID) + (x01 & ClusterID)) [See “Control Unit Images” on page 6–43.]
LIBRARY NED		
64-95		0
TOKEN NED		
96		Flags = 1110 1100
97		Type = x00 (Unspecified)
98		Class = x00 (Unspecified class)
99	0-6	0
	7	Level = 0 (no hierarchic Relationship)

Byte	Bit	Description
100-105		Type Number = xF0F0F3F4F9F0 (EBCDIC for 003490)
106-108		Model Number = xC2F4F0 (EBCDIC for B40)
109-111		Manufacturer = xE2E3D2 (EBCDIC for STK)
112-113		Plant of Manufacture
114-116		Family code which identifies this subsystem as a VTSS (EBCDIC)
117-125		Sequence Number in EBCDIC
126-127		Tag = x0000
GENERAL NEQ		
128		Flags = 1000 0000
129		Record Selector = x80 for Control Unit 0 == Cluster 0 x81 for Control Unit 1 == Cluster 1
130-131		Interface ID (Refer to the “Tag or Interface ID” table under Read Subsys Data command)
132		Device Dependent Time-out = x00
133-135		Reserved = 0 (tbd: MIHPTO / MIHSTO)
136		Extended Information = x00 for logical addresses x0 - x7 x01 for logical addresses x8 - xF
137-159		0

Read Device Characteristics Command

The Read Device Characteristics command transfers 64 bytes of data from the control unit to the channel. The information fields are defined in Table 47 for the VTSS.

Table 47. ***Read Device Characteristics Command data***

Byte	Bit	Description
0-1		Control unit type = x3490
2		Control unit model = x20
3-4		Tape unit type = x3490
5		Tape unit model = 0x40
6-7		0
8	0-3	0
	4	0 - Set special intercept condition
	5	1 - Channel Path no operation
	6	1 - Logical Write Protect
	7	1 - Extended Buffered Log
9	0	0
	1	1 - IDRC
	2	0
	3-7	0
10		Device class code = x80
11		Device type code = x80
12-31		0
32-39		0
40		Miscellaneous data record = x42 (Model A10 or A20)
41		Outboard recorder (OBR) ID = x81 (Model A10 or A20)
42-63		0

Read Forward Command

This command transfers data from the control unit to the channel. It operates as described in the *IBM 3490 Command Reference*.

Note: When data chaining a Read Forward command, the VTSS supports a minimum data chained update count as determined by the following formula:

T_{update} * (65152 / Update_Count) < 400ms

Where:

T_{update} = the time between data chained command updates (in milliseconds)

Update_Count = CCW count for the data chained update (in bytes, rounded up)

As an example, if T_{update} is 50 usec (0.050 msecs), the data chain update CCW count must be greater than 17 bytes.

Read Message ID Command

The Read Message ID command is used to read the message identifier that was assigned by the control unit to commands that indicated the message-required flag requesting notification when an asynchronous operation is complete.

This command must be chained to a sequence of PSF and PSR commands, and the VTD does not implement this sequence. The Read Message ID command always returns unit check status with ERA code 27.

Read Subsystem Data Command

This command is used to obtain information from the subsystem. The data returned by the subsystem depends on the command immediately preceding this command. If the Read Subsystem Data command is the first command in a command chain or it is not immediately preceded by a Set Interface Identifier command, a sequence error is detected and it is unit checked with ERA code 27. The data returned is dependent on the Node Selector value in the Set Interface Identifier data.

Table 48. Identification of Current Interface (node selector=0)

Byte	Bit	Description
Node Descriptor (ND)		
0		Flags = 0
1		Reserved = 0
2		Class = 0 (unspecified)
3		Reserved = 0
4-9		Type Number = xF0F0F3F4F9F0 (EBCDIC for 003490)
10-12		Model Number = xC2F4F0 (EBCDIC for B40)
13-15		Manufacturer = xE2E3D2 (EBCDIC for STK)
16-17		Plant of Manufacture
18-29		Sequence Number in EBCDIC

Byte	Bit	Description
30-31		Tag (or Interface ID) (refer to Tag or Interface ID table)
Node Qualifier (NQ)		
32		Flags = 0
33-39		Reserved = 0
40-43		Node Qualifier Information Port 0 = 0x42010080 (which is comprised of the following individual bits)
	0-1	Entry type = 01 (contains interface ID)
	2-3	Reserved = 0
	4-7	Interface Protocol Type = 0010 (ESCON I/O)
	8-11	Reserved = 0
	12-15	SAT (Subassembly Type) = 0001 (LED Fiber optic)
	16-31	Interface ID (refer to Tag or Interface ID table)

Table 49. ***Identification of Specified Interface 1D(node selector=1)***

Byte		Description
Node Descriptor (ND)		
0		Flags = 0
1		Reserved = 0
2		Class = 0 (unspecified)
3		Reserved = 0
4-9		Type Number = xF0F0F3F4F9F0 (EBCDIC for 003490)
10-12		Model Number = xC2F4F0 (EBCDIC for B40)
13-15		Manufacturer = xE2E3D2 (EBCDIC for STK)
16-17		Plant of Manufacture
18-29		Sequence Number in EBCDIC
30-31		Tag (or Interface ID) (refer to Tag or Interface ID table)

Table 50. ***Identification of Specified Interface 1D(node selector=2)***

Byte	Bit	Description
Node Descriptor (ND)		
0		Flags = 0
1-3		Reserved = 0

Byte	Bit	Description
4-9		Type Number
10-12		Model Number
13-15		Manufacturer
16-17		Plant of Manufacture
18-29		Sequence Number in EBCDIC
30-31	0-15 Tag (refer to Tag or Interface ID table)	

Table 51. ***Identification of Specified Interface 1D(node selector=2) but attached node is unknown***

Byte	Bit	Description
Node Descriptor (ND)		
0		Flags = 0
1-31		0

Table 52. ***Tag or Engraves ID***

Bit	Description
0-15 Tag (or Interface ID) 00000000 0ssc00pp where ss = Card slot in cluster (00-11) c = Cluster (0 or 1) pp = ESCON port id (00-01)	

Sense ID Command

The Sense ID command transfers 20 bytes of data from the control unit to the channel. Table 53 shows specific data returned for this product. The CIW format does not differ from that of the *IBM 3490 Hardware Reference* but is included here for convenience.

Table 53. Sense ID Command data

Byte	Description
0	xFF
1-2	Control unit type = x3490 (Model A10 or A20)
3	Control unit model = x20 (Model A20)
4-5	Tape unit type = x3490 (model B20 or B40)
6	Tape unit model = x40 (Model B20 or B40)
7	0
8-11	CIW Read Configuration Data = x40FA00A0
12-15	CIW Set Interface Identifier = x41730004
16-19	CIW Read Node Identifier = x423E0060 (specifies the Read Subsystem Data)

Table 54. Channel Information Word (CIW) Format

Bit	Description
0-1	Entry Type - 0,1 (for CIW)
2-3	0,0
4-7	Command type specified x0 (Read Configuration Data) x1 (Set Interface Identifier) x2 (Read Node Id)
8-15	Command Code
16-31	Maximum bytes by specified command

Set Interface Identifier

The valid Interface ID's for VTSS are: 0x0000, 0x0001, 0x0010, 0x0011, 0x0020, 0x0021, 0x0030, 0x0031, 0x0040, 0x0041, 0x0050, 0x0051, 0x0060, 0x0061,0x0070, and 0x0071. The actual Interface ID for an addressed device-path pair can be determined from the RCD command.

Set Tape Write Immediate Command

On physical tape I/O subsystems, the Set Tape-Write-Immediate command causes all subsequent Write commands in the channel program to perform as write-immediate commands.

For the VTSS, this command has no effect since tape writes *always* operate in non-buffered write mode.

Synchronize Command

On physical tape I/O subsystems, the Synchronize command causes synchronization between the tape drive and the controlling computer after buffered write operations.

Since the VTSS tape I/O does not “buffer” writes, this command performs no real operations and as such executes immediately without disconnection.

Write Command

The Write command causes data for one logical block to be transferred from the channel to the control unit. This command executes as described with the exception that it only operates in Non-buffered Write Mode.

Hint: When data chaining a Write command, the VTSS supports a minimum data chained update count as determined by the following formula:

T_{update} * (65152 / Update_Count) < 400ms

Where:

T_{update} = the time between data chained command updates (in milliseconds)

Update_Count = CCW count for the data chained update (in bytes, rounded up)

As an example, if T_{update} is 50 usec (0.050 msecs), the data chain update CCW count must be greater than 17 bytes.

Path Management Commands

The VTSS attempts to perform path management commands in a manner identical to the 3490E tape controller. For detailed information on how path management commands are designed to be handled by tape control units, refer back to one of the following:

- *IBM 3480 Magnetic Tape Subsystem Reference*
- *IBM 3480 Installation Guide and Reference*

While these describe their operation in the 3480 tape subsystem, the architecture of the commands for managing path groups, especially for “Selectable Devices”, does not appear to have been changed. In fact, to be compatible across multiple generations of equipment, they cannot be changed, only extended.

Assign (ASN)

No Change from the 3490E.

Control Access (CAC)

No Change from the 3490E.

Sense Path Group ID (SNID)

(See “Path Group Effects” on page 312.)

Set Path Group ID (SPID)

(See “Path Group Effects” on page 312.)

Suspend Multipath Reconnection (SMR)

As for an IBM 3490, this is effectively a non-operation as Multipath Reconnect mode for a path group is not supported.

Unassign (UNA)

No Change from the 3490E.

Status and Sense Bytes

Status Byte

The status byte provides response information regarding the current channel command word (CCW). This is the information that is contained in bits 32-39 of the System/370 channel status word (CSW) and also in the CSW formatted by the I/O Supervisor components of MSP Extended Architecture. The VTSS status byte is identical to the status byte described in the *IBM 3490 Hardware Reference*.

Sense and Log Data Formats

Sense and log data record and report subsystem error and error correction information. The Sense command transfers 32 bytes of data from the control unit to the channel. If the Sense command follows a Rewind Unload command, Format 22 data is transferred. Otherwise, it is Format 20 data. Both formats contain 32 bytes of information. The Read Buffered Log command transfers 64 bytes of Format 30 buffered log data to the channel. The other formats described in the *IBM 3490 Hardware Reference* are not applicable to VTSS.

The contents of bytes 0-7and 27-30 are common to all formats (for the 3490, it is only bytes 0-7,27-29 that are common). The contents of the remaining bytes are determined by the format, which is reported in sense byte 7. Differences between the VTSS sense and log data and those of the 3490 and fields whose values are constant or limited for VTSS are described in the following tables.

Formats 20, 22, and 30 Sense bytes 0-7

Table 55. Sense bytes 0-7

<table>
<tr><th>Sense Byte</th><th>Description</th></tr>
<tr><td>0</td><td>
<table>
<tr><td>Bits</td><td>Value</td><td>Description</td></tr>
<tr><td>0</td><td>x</td><td>Command Reject</td></tr>
<tr><td>1</td><td>x</td><td>Intervention Required</td></tr>
<tr><td>2</td><td>0</td><td>Bus Out Check (FIPS only)</td></tr>
<tr><td>3</td><td>x</td><td>Equipment Check - this bit is set for:
- Control Unit ERP failure (ERA 4A)
- Physical end of tape</td></tr>
<tr><td>4</td><td>x</td><td>Data Check - VTSS detected an invalid condition in the absence of a detected hardware malfunction</td></tr>
<tr><td>5</td><td>0</td><td>Overrun (synchronous mode only; not supported by VTSS)</td></tr>
<tr><td>6</td><td>x</td><td>Deferred Unit Check - Unit Check not associated with current cmd</td></tr>
<tr><td>7</td><td>x</td><td>Assigned elsewhere</td></tr>
</table>
</td></tr>
</table>

Sense Byte	Description
1	Bits Value Description 0 x Locate Failure 1 1 Drive Online to Control Unit 2 0 Reserved 3 0 Record Sequence Error - not applicable to VTSS 4 x Beginning of Tape 5 x Write Mode - most recent command was a write-type command 6 x Write Protect 7 0 Not Capable - not applicable to VTSS
2	Bits Value Description 0-3 xxxx Reporting Channel Path 4 x Reporting Control Unit 5 0 Automatic Cartridge Loader Active - not applicable to VTSS 6 0 Tape Synchronous Mode - not applicable to VTSS 7 x Tape Positioning - must be set to enable Read Opposite Recovery implementation of Read Backward command
3	Error Recovery Action (ERA)
4	Bits Value Description 0-1 01 3480-2 XF format 2-3 00 Reserved 4-7 xxxx High order bits of Channel Logical Block Number - the next data block or tape mark to be read or written in a forward direction
5-6	Last 16 bits of Channel Logical Block Number
7	Identifies format of remaining sense or log bytes: 0x20, 0x22, or 0x30

Format 20 Sense Bytes 8-31

Table 56. Format 20 - Sense bytes 8-31

Sense Byte	Description
8	0
9	VTSS Extended Path Info Bits Value Description 0-1 0 Reserved (HPID expansion) 2-3 00bb ESCON adapter Host-Path number (HPID) 4-7 0 Reserved (VDID expansion, Most Sig Bits >8)
10-11	Information for service representative - Fault Symptom Code (FSC)
12-13	Information for service representative - Reason Code
14-21	0
22-23	Reserved (Iceberg compatibility FSC)

Sense Byte	Description
24	Channel Adapter and Data-Transfer Mode Bits Value Description 0 -3 xxxx Channel adapter 4-7 0xE Data-transfer mode - ESCON 9.0 MB/s
25	Control Unit Features Installed Bits Value Description 0 1 Dual Control Unit Communication Coupler 1-3 0 Reserved 4 1 Improved Data Recording Capability enabled 5 0 Reserved 6 0 Upgraded Buffer 7 0 Automatic Cartridge Loader
26	Microcode EC Level - (not applicable) VTSS Sense-Bytes Revision Level (0x01)
27-29	Subsystem Configuration Bits Value Description 0-3 0010 Model Definition - Model A20/B40 4-23 x..x Sequence Number (lower order 12 bits plus virtual drive id in hex)
30	Virtual tape drive identifier 0-255 (VDID)
31	0 Buffered Write Data Bytes - not applicable

Format 22 Sense Bytes 8-31

Format 22 is presented when a Sense command follows a Rewind Unload command.

Table 57. ***Format 22 Sense Bytes 8-31***

Sense Byte	Description
8	0
9	VTSS Extended Path Info Bits Value Description 0-1 0 Reserved (HPID expansion) 2-3 00bb ESCON adapter Host-Path number (HPID) 4-7 0 Reserved (VDID expansion, Most Sig Bits >8)
10-11	Information for service representative - Fault Symptom Code (FSC)
12-21	0 Reserved
22-23	Reserved (Iceberg compatibility FSC)
24-26	0 Reserved
27-29	Subsystem Configuration Bits Value Description 0-3 0010 Model Definition - Model A20/B40 4-23 x..x Sequence Number (lower order 12 bits plus virtual drive id in hex)
30	Virtual tape drive identifier 0-255 (VDID)
31	0 Reserved

Format 30 Sense and Log Bytes 8-63

Format 30 is presented in response to a Read Buffered Log command.

Table 58. ***Format 30 - Sense Bytes 8-31***

Log Byte	Description
8-23	0
24	Count of Data-Request Errors (Excludes Data-request Time-outs) (used to represent internal data transfer errors, i.e. CRC errors)
25	0
26	Cleaning the Tape Drives and Volume Format. Bit 0-3 0000 (no cleaning support) 4-5 00 (no tape mounted) 10 (3480-2 XF format) 6 0 (Volume format changed) 7 0 (no forced error logging active)
27-29	Subsystem Configuration Bits Value Description 0-3 0010 Model Definition - Model A20/B40 4-23 x..x Sequence Number (lower order 12 bits plus virtual drive id in hex)

Log Byte	Description
30	Virtual tape drive identifier 0-255 (VDID)
31	Length of Tape Currently Mounted 0x00 Tape is unloaded 0x90 Enhanced Capacity Cartridge System Tape

Format 30 Log Bytes 32-63

Table 59. Format 30 Log Bytes 32 - 63

Log Byte	Description
For bytes 32-43, counts of bytes processed, each count is equal to 4K bytes (4096) and rounded up to the nearest 4K byte increment for blocks of less than 4K bytes. For example, a 3456-byte block of data is counted as ‘01’, and a 5678-byte block of data is counted as ‘02’.	
32-34	Count of Channel Read Bytes Processed (uncompressed)
35-37	Count of Channel Write Bytes Processed (uncompressed)
38-40	Count of Device Read Bytes Processed (compressed)
41-43	Count of Device Write Bytes Processed (compressed)
44-46	Count of Channel Read Blocks (including tape marks) Processed
47-49	Count of Channel Write Blocks (including tape marks) Processed
50-52	Count of Device Read Blocks (including tape marks) Processed
53-55	Count of Device Write Blocks (including tape marks) Processed
56-63	0

Error Recovery Action (ERA) Codes Support

Many of the Error Recovery Action (ERA) codes that are appropriate for a physical tape I/O device are not appropriate in a virtual tape device. Therefore the VTSS only supports a subset of the possible ERAs. They are identified by an X in Table 60.

Table 60. Error Recovery Action Codes

VTSS	Code	Description
X	00	Unsolicited Sense
	21	Data Streaming Not Operational
	22	Path Equipment Check
X	23	Read Data Check
	24	Load Display Check
	25	Write Data Check
X	26	Read Opposite
X	27	Command Reject
	28	Write ID Mark Check
X	29	Function Incompatible
	2A	Unsolicited Environmental Data
X	2B	Environmental Data Present
	2C	Permanent Equipment Check
	2D	Data Security Erase Failure
X	2E	Not Capable (BOT Error)
X	30	Write Protected
	31	Tape Void
	32	Tension Loss
	33	Load Failure
	34	Unload Failure
X	35	Drive Equipment Check
X	36	End of Data
	37	Tape Length Error
X	38	Physical End of Tape
X	39	Backward at BOT
X	3A	Drive Switched Not Ready
	3B	Manual Rewind/Rewind-Unload

VTSS	Code	Description
	40	Overrun
	41	Record Sequence Error
	42	Degraded Mode
X	43	Drive Not Ready
X	44	Locate Block Unsuccessful
X	45	Drive Assigned Elsewhere
X	46	Drive Not Online
	47	Volume Fenced
X	48	Unsolicited Informational Data
	49	Bus-Out Check
	4A	Control Unit ERP Failed
	4B	Control Unit and Drive Incompatible
	4C	Recovered Check-One Failure
X	4D	Resetting Event
X	4E	Maximum Block Size Exceeded
X	50	Read Buffered Log (Overflow)
X	51	Read Buffered Log (EOV)
X	52	End of Volume Complete
X	53	Global Command Intercept
	54	Channel Interface Recovery (Temporary)
	55	Channel Interface Recovery (Permanent)
X	56	Channel Protocol Error
X	57	Global Status Intercept
	5A	Tape Length Incompatible
	5B	3480 XF Incompatible
	5C	Format 3480-2 XF Incompatible
	5D	Tape Length Violation
	5E	Compaction Algorithm Incompatible

Control Unit Images

Introduction

Unlike recent prior generations of tape drives where the trend has been to “1-by-1” devices (a separate controller for each drive), the VTSS has an “N-by-N” architecture. Each Virtual Tape Unit (VTU) can be accessed from more than one Control Unit (CU), and each CU accesses multiple VTUs. This provides a high availability for every VTU due to the elimination of any single point of hardware failure or contention.

“Multiple Control Units” does not refer to multiple VTSS subsystems. Rather it refers to how an single VTSS behaves as if it were comprised of multiple CUs. It appears so for physical reasons (multiple hardware Clusters) and logical reasons (Virtual Controller images).

Dual Control Units - DCUs

A single VTSS subsystem is comprised of two fully redundant hardware “Clusters”. Each Cluster has its own set of ESCON physical paths and can access any VTU. Since it presents the same appearance as a 3490 “Dual Control Unit” (DCU), the VTSS reports itself a “Dual Control Unit” (DCU) type in all sense data. This includes the RCD, RDC, RSSD, and SNSID commands as well as the SNS command.

The selected ESCON path determines which half of the DCU is involved in an operation. Thus the DCU number (0,1) is based on a physical construct - the ESCON physical path.

Virtual Control Units - VCUs

Since a VTSS has so many more available data-paths and tape units than a 3490 subsystem, it cannot appear as a single “2-by-N” 3490E subsystem. An actual 3490 subsystem may have a maximum N of 16. But the VTSS supports 64 virtual tape units, which is 4 times the maximum supported by a single 3490 subsystem. The solution is have the VTSS present multiple Virtual Control Unit images (VCUs) to the host.

To support 64 VTUs the VTSS must appear as 4 separate VCUs, each controlling a range of 16 VTUs. Note that a range of VTUs is always comprised of sequential device addresses. So VTUs x40-x4F all appear to be attached to the same VCU, whereas VTUs x00, x40, x80, and xC0 appear attached to 4 distinct VCUs.

Thus the VCU number (0-3) is based on a virtual construct - the VTU device address.

DCU versus VCU

Note that VCUs are entirely distinct from the “Dual Control Unit” construct. VCUs are virtual and dependent upon VTU device address but independent of ESCON path, whereas DCUs are physical and dependent upon ESCON path but independent of VTU device address. This means that each VCU appears to be a DCU.

In other words, while the VTU device address determines the VCU image, the ESCON path address determines which Cluster is involved, and so which half of the DCU is involved. This means that a VTSS subsystem appears to be comprised of 8 individual CU images arranged as 4 DCUs, with each DCU controlling 16 VTUs.

Whenever an operation selects a particular ESCON path and device address, this determines which of the 8 CU images has been selected.

Note that not every Sense-type command directly reflects the VCU and DCU image. The basic SNS command data does, and so does the RCD data. The RSSD command data reflects DCU, but the RDC and SNSID commands' data does not. The VCU/DCU address information is reported directly only in the RCD data, in the Control Unit NED Tag field.

System Reset Effects

Under ESCON, a System Reset must be addressed not just to a path, but to a particular Control Unit. As the DCU is tied to a particular path, and System Reset is issued to a specific path, DCU construct has no particular effect upon System Reset processing that deviates from a 3490 subsystem.

But VCU cuts across path-address. Each VCU must be defined to ESCON as a distinct and separate CU (because of the 16 device per 3490 CU limitation). Hence, ***to reset an ESCON path to a VTSS one must issue 4 System Resets - one to each VCU*** "attached" to that path.

Path Group Effects

The DCU construct has no effect upon path grouping. This is no different than in a 3490 subsystem. Any combination of pathing commands issues to a single VTU over multiple paths is not affected by whether or how the paths used are split across DCUs.

But the VCU construct does have an effect, and it does not map precisely to any 3490 subsystem effects.

All path management commands are addressed to a particular ESCON path and VTU. All may affect the path grouping and partitioning state of the device and/or path (or path-group). However, the effect of VCUs is noticed in the SPID and SNID commands.

SPID Command -- Path versus Device -- Deep Background

The SPID command "Establishes" or "disestablishes" a particular device-path (specific ESCON path to a specific VTU device). The "disestablish" operations have two flavors: "Resign" and "Disband". But for purposes of this discussion, the difference is irrelevant., and no distinction will be made.

The SPID command also "establishes" the *path* over which it is issued, regardless of the selected VTU device and r***egardless of whether the specified operation establishes or disestablishes the selected device***. In other words, if a SPID command is sent down a "not-previously-established" ESCON path, *the path itself is always "Established"*, even if the specified operation was "Resign" or "Disband" and no device-path is established.

This behavior is explained by analyzing the SNID command. The SNID command reports both pathing status and partitioning status.

Partitioning Status refers to the implicit or explicit assignment state (Enabled or Disabled) for the selected device-path. It is affected directly by the ASN, UNA, and CAC commands, and indirectly by the SPID command and System Reset. Note that it has no path-specific meaning independent of a specific device. It is *unaffected by the VCU or DCU* construct.

Pathing Status refers to path and path-group status for the selected path and device-path. It is affected solely by the SPID command and System Reset. It depends upon both the pathing state of the path and of the device-path, which are partially independent. The device-path can either be “Grouped” or “Ungrouped”, and in addition the path itself can also be “Reset”.

The device-path state is set to “Grouped” in response to a SPID-Establish command, and is set to “Ungrouped” in response to a SPID Resign or Disband command. The path state is set to “Reset” in response to a System Reset signal. The “Reset” status is cleared by the first subsequent SPID command (of any type) issued down this path.

When a System Reset occurs, all device-path pathing and partitioning states associated with this path are reset (cleared, forgotten). Hence the SNID issued to any device on this path will return “Reset” as its Pathing Status and zeroes as a Path-Group-ID. (Note that this is distinct from the “Resetting-Event” Unit-Check, which is returned for the first command issued to a device-path after a System Reset. The “Reset” pathing status will continue to be presented by a SNID to any device-path associated with the path until a SPID command is accepted by any device on that path. After that, a SNID to any device on that path will present either “Grouped” or “Ungrouped”.

While in “Reset” Pathing Status, a device-path can accept other commands, including ASN and UNA, which may change its Partitioning Status, but will not affect its Pathing Status.

So to summarize, a path is “established” independent of the device-paths associated with it, and this consists of clearing the path’s “Reset” Pathing Status and also in setting the path’s Path-Group-ID (both reported by SNID). Note that once a path’s Path-Group-ID has been set, any subsequent SPID command must specify the same Path-Group-ID, or it will be rejected.

VCU Effects on SPID and SNID

The effects of VCU upon SPID and SNID are counter-intuitive from the way VCU affects System Reset processing. Basically, when a SPID is issued to a path which is in the “Reset” Pathing Status, it clears the “Reset” status and sets the Path-Group-ID for the path ***for all VCUs, regardless of the VCU to which it was addressed.*** In other words, a SPID (of any type) to *any* VCU establishes that path for *all* VCUs.

Note that this is opposite of the effect for System Reset, which only resets the path for a particular VCU.

Appendix C. NCS/VTCS Alphabetic Volsers

NCS/VTCS supports alphabetic volser ranges for all commands and utilities. The rules for alphabetic volser ranges are as follows:

1. An alphabetic volser range consists of a pair of volsers (start volser and end volser) containing an incrementing alphabetic portion of 1 to 6 characters. For example:, 00000<u>A</u>-00000<u>Z</u>, ABC<u>AAA</u>-ABC<u>ZZZ</u>, 9<u>AA</u>000-9<u>CC</u>000, A00<u>A</u>00-A00<u>M</u>00.

   a. A volser is composed of sequence of one to six numerics, (upper case) alphabetic or national characters (#, @ and the primary national currency symbol).

   b. A volser of less than six characters is left justified and blank padded. Each volser element in a range must have the same number of characters specified. For example, if the first volser element is 4 characters, the second must be exactly 4 characters.

2. The start and end volsers forming a volser range consists of the following sub-elements: an optional prefix, an incremental portion, and an optional suffix. Table 64. on page 318 shows examples of alphabetic volser ranges.

   a. The optional prefix consists of identical leading characters (if any) in the start and end volsers.

   b. The incremental portion starts at the first non-identical leading character in the start and end volsers forming a range. The incremental portion is either:

      - All numeric (contains characters 0 through 9 only).
      - All alphabetic (contains character A through Z only).

      The incremental portion of a volser range, therefore, terminates where a change of character type (numeric -> alphabetic or alphabetic -> numeric) is detected.

The incremental type is derived from the character type of the first character in the incremental part (numeric/alphabetic). Table 61 shows example incremental ranges.

Table 61. ***Example Incremental Ranges***

volser Range	Incremental Portion	Data Type
00000A-00000Z	A-Z	Character
ABCAAA-ABCZZZ	AAA-ZZZ	Character
9AAZ00-9CCZ00	AAZ-CCZ	Character
A00B00-A99B00	00-99	Numeric
A00A00-A00M00	A-M	Character
A00B00-A00B99	00-99	Numeric

Note the following rules for incremental ranges:

- The expansion of an alphabetic incremental part is derived from a collating sequence of A-Z (it will not include the national character set).
- The data types of the incremental portions in the start and end volsers must be identical.
- The position of the incremental portion of the start volser must match that of the end volser.
- The length of the incremental portion of the start and end volsers must be identical.
- The incremental portion of the end volser must be greater than or equal to the start volser.

c. The optional suffix consists of the trailing characters from the end of the incremental portion onwards. Table 62 shows an example range suffix.

Table 62. ***Example Range Suffix***

Volser Range	Incremental Portion	Suffix
A00B00-A00B99	00-99	none
A00B@0-A00D@0	B-D	@0
9AAZ00-9CCZ00	AAZ-CCZ	00 (not Z00)
900A@A-950A@A	900-950	A@A
ABCAAA-ABCZZZ	AAA-ZZZ	none

For a range to be valid the suffix of the start and end volsers forming the range must be identical.

3. The number of volumes generated from an alphabetic volser range is dependent on the number of elements in the incremental portion of the volser elements. For an A to Z range in each character position, the number of volumes can be calculated by 26 to the power of the number of positions that are being incremented as shown in Table 63.

Table 63. ***Size of Alphabetic Volser Ranges***

Range	Calculation	Number of Volumes
A-Z	26^1	26
AA-ZZ	26^2	676
AAA-ZZZ	26^3	17,576
AAAA-ZZZZ	26^4	456,976

Warning: Per Table 63, it is possible to define 26^4 VTVs in a single range. **Note, however, that** the more VTVs you define, the bigger your CDS has to be.

Alphabetic Volser Examples

Table 64 and Table 65. on page 319 describe valid and invalid alphabetic ranges.

Table 64. ***Valid Alphabetic Ranges***

Range	Subcomponents			Number of VTVs
	Prefix	Incremental Portion	Suffix	
AAA000-AAZ000		AAA-AAZ	000	26
A00A00-A00A99	A00A	00-99		100
0AAAA0-0ZZZZ0	0	AAAA-ZZZZ	0	456,976
A00A00-A99A00	A	00-99	A00	100
99AA##-99ZZ##	99	AA-ZZ	##	676
A9A000-A9Z000	A9	A-Z	000	26
######-######	######			1
AA00##-ZZ00##		AA-ZZ	00##	676
AA00##-AA99##	AA	00-99	##	100
PROD00-PROD99	PROD	00-99		100
PROD00-PROZ00	PRO	D-Z	00	23
A4Z#@0-A9Z#@0	A	4-9	Z#@0	6
A4Z#@0-Z4Z#@0		A-Z	4Z#@0	26
A4Z#@0-A4Z#@6	A4Z#@	0-6		7
AAAAAA-AAACCC		AAAAAA-AAACCC		1407
A3BZZ9-A3CDE9	A3	BZZ-CDE	9	84
999AM8-999CM8	999	AM-CM	8	53
111AAA-111ZZZ	111	AAA-ZZZ		17576

Table 65. Invalid Alphabetic Ranges

Range	Subcomponents			Number of VTVs	Comments
	Prefix	Incremental Portion	Suffix		
0AAAAA-0BAAAA	0	AAAAA-BAAAA		456,977	Greater than 456,976 VTVs
A9A000-A9Z999					Cannot mix incremental portions
######-#####@					National characters cannot increment
AA00##-ZZ99##					Invalid range
CCNNZZ-CDNZAA		CCNNZZ-CDNZAA		464,414	Greater than 456,976 VTVs
A4Z#@0-A9Z#@9					Invalid range

Appendix D. Using the HSC Significant Event Notification Facility

HSC provides a Significant Event Notification Facility (SEN). The SEN has a macro interface that allows an application to request notification of specific HSC and VTCS events. The application can then process the data passed by the SEN. SEN notification requests are maintained across HSC warm starts and are purged on HSC cold starts. When HSC is cold started, therefore, all previously established notification requests must be reestablished.

The SEN macro interface is supported at HSC base service level. There is no operator interface for the SEN itself, but the HSC 5.0.0 and above `Display` command lists SEN notification requests.

For information on the XML format output of the supported SEN events, see “VTCS and HSC Events XML Tags” on page 345.

Caution: When HSC initializes, the SEN facility is disabled, so you must enable via the HSC `OPTION SEN` operator command. For example, to enable the SEN, enter the following:

```
.OPTION SEN=ON
```

Overview of the SEN Macro Interface

You invoke the `SLSXSEN` macro to make SEN requests. The `SLSXSEN` macro has two forms:

- A **list form** that generates a parameter list. The parameter list is mapped by macro `SLSXSENM` and must be included in routines invoking `SLSXSEN`.
- The **execute form** that populates the parameter list and calls the SEN request module `SLSXSENR`. Modules using the `SLSXSEN` macro must have an authorization code of 1 and reside in an APF authorized library.

The `SLSXSEN` macro supports the following four requests:

`LISTEN`
: Request notification for SEN defined events, which requires providing a routine to be invoked when the event occurs.

`DELETE`
: Remove a specific event notification request.

`DISABLE`
: Disable the `LISTEN` routine for event notification request.

`ENABLE`
: Enable the `LISTEN` routine for event notification request.

Programs invoking `SLSXSEN LISTEN` requests must supply the entry points of two routines:

- The SEN request module `SLSXSENR`.
- The listener routine to get control when an event occurs.

`SLSXSENR` must be loaded from an APF authorized library. The listener routine must reside in common storage. The invoking program is responsible for managing this storage.

The listener routine is validated during `LISTEN` request processing by invoking the routine with general purpose register 1 set to zeros. Therefore, the listener routine must be sensitive to this condition and process it accordingly. The listener routine can simply return back to the caller on this condition or perform any type of initialization required. If `LISTEN` request processing detects an abend during validation, the request is rejected. A successful `LISTEN` request returns a token. Programs invoking `SLSXSEN` with `DELETE`, `DISABLE`, or `ENABLE` requests must provide this token to target a specific `LISTEN` request. The HSC SAMPLIB contains a sample program showing the use of `SLSXSEN`.

When events occur for which there is a listener routine, it is invoked and receives data that describes the event. The data passed is in XML format, which is provided on the StorageTek Customer Resource Center (CRC). For more information on the supported events, see “Supported HSC and VTCS SEN Events” on page 341.

When SEN request processing completes and control is returned to the requesting program, the parameter list and Register 15 contain the return code. Return codes are described in “Execute Form - Syntax and Parameters” on page 325.

If HSC detects an abend while the listener routine is in control, the associated request is disabled and must be programmatically reenabled or deleted. When a notification request is disabled, the listener routine is not invoked.

Note: The HSC 5.0 `OPTION` command and control statement lets you set the SEN facility ON or OFF.

`OPTion SEN(ON) | OPTion SEN(OFF)`

Note that even if SEN is disabled, `SLSXSEN` requests can still be processed.

SEN Macro Interface Authorization Requirements and Module Attributes

Modules invoking the SEN macro interface must:

- Have an authorization code of 1 and reside in an APF authorized library, **or**
- Be running in key 0 - 7 or supervisor state and be running in AMODE 31.

The listener routine must be running in TCB mode (not in SRB mode) and must be re-entrant.

The SLSXSEN Macro

The `SLSXSEN` macro has two forms:

- A **list form** that generates a parameter list. The parameter list is mapped by macro `SLSXSENM` (see “SLSXSENM Macro Format” on page 324) and must be included in routines invoking `SLSXSEN`. On completion of SEN request processing, the SEN parameter list contains the return code and token response areas.
- The **execute form** that populates the parameter list and calls the SEN request module `SLSXSENR`.

List Form - Syntax and Parameters

Syntax

```
►►─label─SLSXSEN──MF=L─────────────────────────────────────────►◄
```

Figure 147. ***SLSXSEN Macro Syntax - List Form***

Parameters

`MF=L`
: Specifies that a remote parameter list is generated. All other options are ignored.

SLSXSENM Macro Format

Table 66 shows the format of macro SLSXSENM, which generates a DSECT that maps the SEN parameter list generated by the list form of macro SLSXSEN.

Table 66. SLSXSENM Macro Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
0	0			SLSXSENM	Start of DSECT
0	0	hexstring	2	SLSXSLEN	Length of parameter list
2	3	bitstring	1	SLSXSFC	SEN request code
		X’01’		SLSXLIS	LISTEN
		X’02’		SLSXSDEL	DELETE
		X’03’		SLSXSDIS	DISABLE
		X’04’		SLSXSENA	ENABLE
3	3	bitstring	1	SLSXSFLG	Processing flag
		X’80’		SLSXSEOT	EOT = YES
		X’40’		SLSXSEOM	EOM = YES
4	4	hexstring	8	SLSXSEM	Event mask
12	C	hexstring	4	SLSXSLA	Listener routine address
16	10	hexstring	4	SLSXSRT	Current TCB of requestor
20	14	hexstring	4	SLSXSRA	ASCB of requestor
22	16	hexstring	2	SLSXSTOK	Listen request token
30	1E	character	8	SLSXSREQ	Requestor name
38	26	character	8	SLSXSLNR	Listener routine name
40	28	hexstring	2	SLSXSQA	SEN queue entry address
42	2A	hexstring	2	SLSXSRC	SEN return code
			36	SLXRDI	Duplicate request encountered, request ignored
			32	SLXRNHSC	HSC not active or at proper level
			28	SLXRBADA	Authorization requirements not met
			24	SLXRBILR	Listener failed validation
			20	SLSXRSAB	Abend in HSC SEN processing
			16	SLSXRIR	Invalid request
			12	SLSXRNSQ	No LISTEN request found for DELETE, DISABLE, or ENABLE

Table 66. SLSXSENM Macro Format

Decimal Offset	Hexadecimal Offset	Type	Length	Label	Description
			08	SLSXRDR	Duplicate listener encountered
			04	SLSXRDOK	SEN request successful, however the SEN facility is disabled
44	2C			SLSXSMLN	Length

Execute Form - Syntax and Parameters

Syntax

```
►—label—SLSXSEN—function—parameter——————————————————————►◄
```

Figure 148.SLSXSEN Macro Syntax - Execute Form

Parameters

function is one of the following:

- LISTEN; see “LISTEN” on page 326.
- DELETE; see “DELETE” on page 330.
- DISABLE; see “DISABLE” on page 331.
- ENABLE; see “ENABLE” on page 333.

LISTEN

LISTEN requests notification for the events described in “Supported HSC and VTCS SEN Events” on page 341. Also see “The SEN Listener Exit Routine” on page 335.

Note: A sample program of a LISTEN request (SENRQST) is provided in file SLSSAMP during SMP/E installation.

Syntax.

```
label   SLSXSEN LISTEN,
               EVENT=(event_1,event_2,..),
               RTOKEN=token,
               LNRADR=listener_address,
               SENRADR=SLSXSENR_address,
               EOT=YES/NO,
               EOM=YES/NO,
               REQNAME=requestor_name,
               LNRNAME=listener_name,
               MF=(E,parm_list)
```

Parameters.

EVENT=

Specifies the event(s) requiring notification. For more information, see “Supported HSC and VTCS SEN Events” on page 341. You can specify the event name or the numeric equate. Event names are case sensitive. You can specify a single event or EVENT=ALL. This parameter is required and there is no default.

Note: To specify multiple events on a single LISTEN request, StorageTek recommends that you use the numeric equates to ensure that the parameter value specified will not exceed IBM’s 256 byte size limitation for macro variables.

RTOKEN=

Specifies the address of a fullword token associated with a successful LISTEN request. Valid values are any RX-type address or registers 2 through 12. This parameter is required and there is no default.

LNRADR=

Specifies the entry point address of the listener routine that is invoked when the events specified occur. This program must reside in persistent common storage (that is, subpool 241) if EOT = YES or EOM=YES. It is your responsibility to delete the listener routine and free the storage associated with it. Valid values are any RX-type address or registers 2 through 12. This parameter is required and there is no default.

SENRADR=

Specifies the entry point address of the SEN request module (SLSXSENR) that is provided with HSC. SLSXSENR must be loaded before invoking macro SLSXSEN. Valid values are any RX-type address or registers 2 through 12. This parameter is required and there is no default.

EOT=

Specifies whether the listener routine associated with this request is invoked if the task that issued the LISTEN request ends.

YES

Invoke the listener routine if the associated task ends.

NO

Do not invoke the listener routine if the task associated with this request ends.

This parameter is optional and NO is the default.

EOM=

Specifies whether the listener routine associated with this request is invoked if the address space that issued the LISTEN request ends.

YES

Invoke the listener routine if the associated address space ends.

NO

Do not invoke the listener routine if the address space associated with this request ends.

This parameter is optional and NO is the default.

STOP **Caution:** EOT and EOM control execution of the listener routine, so make sure you set them to specify how you want the listener routine to operate.

REQNAME=

Specifies the name of the requestor and is used to uniquely identify a specify request for display purposes.The name must be an eight byte field containing a combination of uppercase alphanumeric values (A-Z,0-9), national characters ($,@,#) with no embedded blanks.Valid values are any RX-type address or registers 2 through 12. This parameter is optional if LNRNAME is not specified and has no default.

`LNRNAME=`

Specifies the name of the listener routine The name must be an eight byte field containing a combination of uppercase alphanumeric values (A-Z,0-9), national characters ($,@,#) with no embedded blanks. Valid values are any RX-type address or registers 2 through 12. If `LNRNAME` is specified, `REQNAME` must also be specified. Otherwise, this parameter is optional and has no default.

Hint: Although not required, `REQNAME` and `LNRNAME` are very useful when displaying active SEN requests. For more information, see "The listener routine must be running in TCB mode (not in SRB mode) and must be re-entrant." on page 323.

`MF=(E,parm_list)`

Specifies the execute form of the macro using the specified parameter list generated by the list form. The parameter list, *parm_list*, can be specified as any RX-type address or registers 2 through 12.

Responses. The return code is set in register 15 and the SEN parameter list. If the return code is zero, the address specified by `RTOKEN` contains the token representing this `LISTEN` request. This token is used when deleting, disabling or enabling a request. Table 67 describes the `SLSXSEN LISTEN` return codes.

Table 67. SLSXSEN LISTEN Return Codes

Return Code	Meaning
00	SEN LISTEN request successful
04	SEN LISTEN request successful however the SEN facility is disabled.
08	Listen request with conflicting event lists. For more information, see “Detecting Duplicate Listen Requests” on page 339.
12	N/A
16	Invalid parameter list received.
20	Indicates that an abend occurred in HSC SEN processing, request rejected.
24	Indicates that listener routine validation failed, request rejected
28	The HSC determined that the SLSXSEN macro is being invoked by a program that does not meet the authorization requirements specified in “SEN Macro Interface Authorization Requirements and Module Attributes” on page 323.
32	The HSC is not active or at the proper initialization level for SEN services.
36	An identical LISTEN request already exists, request ignored. For more information, see “Detecting Duplicate Listen Requests” on page 339.
40	Indicates that the `REQNAM` or `LNRNAME` syntax is invalid. Valid syntax is uppercase A-Z, 0-9, valid national characters ($,#,@), with no embedded blanks.

DELETE

Syntax.

```
label   SLSXSEN DELETE,
               RTOKEN=token,
               MF=(E,parm_list)
```

Parameters.

RTOKEN=
: Specifies the address of a fullword token representing the request to be removed. This token was returned from a successful LISTEN request. This parameter is required and there is no default.

MF=(E,parm_list)
: Specifies the execute form of the macro using the specified parameter list generated by the list form. The parameter list, *parm_list*, can be specified as any RX-type address or registers 2 through 12.

Note: All other parameters are ignored if entered on the DELETE request.

Responses. The return code is set in register 15 and the SEN parameter list. Table 68 describes the SLSXSEN DELETE return codes.

Table 68. SLSXSEN DELETE Return Codes

Return Code	Meaning
00	SEN DELETE request successful.
04	N/A
08	N/A
12	No existing LISTEN request was found using the TOKEN specified on the DELETE request.
16	Invalid request received (not LISTEN, DELETE, DISABLE, or ENABLE).
20	An abend occurred in HSC SEN processing, request rejected.
24	N/A
28	The HSC determined that the SLSXSEN macro is being invoked by a program that does not meet the authorization requirements specified in “SEN Macro Interface Authorization Requirements and Module Attributes” on page 323.
32	HSC is not active or at the proper initialization level for SEN services.
36	N/A

DISABLE

Disable notification for a specific LISTEN request. The targeted request remains but the associated listener routine is not invoked when the specified events occur.

Syntax.

```
label   SLSXSEN DISABLE,
             RTOKEN=token,
             MF=(E,parm_list)
```

Parameters.

`RTOKEN=`
: Specifies the address of a fullword token representing the request to be disabled. This token was returned from a successful `LISTEN` request. This parameter is required and there is no default.

`MF=(E,parm_list)`
: Specifies the execute form of the macro using the specified parameter list generated by the list form. The parameter list, *parm_list*, can be specified as any RX-type address or registers 2 through 12.

Note: All other parameters are ignored if entered on the `DISABLE` request.

Responses. The return code is set in register 15 and the SEN parameter list. Table 69 describes the SLSXSEN DISABLE return codes.

Table 69. SLSXSEN DISABLE Return Codes

Return Code	Meaning
00	SEN DISABLE request successful.
04	SEN DISABLE request successful however the SEN facility is already disabled.
08	N/A
12	No existing LISTEN request was found using the TOKEN specified on the DISABLE request.
16	Invalid request received (not LISTEN, DELETE, DISABLE, or ENABLE).
20	An abend occurred in HSC SEN processing, request rejected.
24	N/A
28	The HSC determined that the SLSXSEN macro is being invoked by a program that does not meet the authorization requirements specified in “SEN Macro Interface Authorization Requirements and Module Attributes” on page 323.
32	HSC is not active or at the proper initialization level for SEN services.
36	N/A

ENABLE

Enable notification for a specific LISTEN request. The targeted request's listener routine is now invoked when the specified events occur.

Syntax.

```
label   SLSXSEN ENABLE,
               RTOKEN=token,
               MF=(E,parm_list)
```

Parameters.

`RTOKEN=`
: Specifies the address of a fullword token representing the request to be enabled. This token was returned from a successful `LISTEN` request. This parameter is required and there is no default.

`MF=(E,parm_list)`
: Specifies the execute form of the macro using the specified parameter list generated by the list form. The parameter list, *parm_list*, can be specified as any RX-type address or registers 2 through 12.

Note: All other parameters are ignored if entered on the `ENABLE` request.

Responses. The return code is set in register 15 and the SEN parameter list. Table 70 describes the SLSXSEN ENABLE return codes.

Table 70. SLSXSEN ENABLE Return Codes

Return Code	Meaning
00	SEN ENABLE request successful.
04	SEN ENABLE request successful however the SEN facility is disabled.
08	N/A
12	No existing LISTEN request was found using the TOKEN specified on the ENABLE request.
16	Invalid request received (not LISTEN, DELETE, DISABLE, or ENABLE).
20	An abend occurred in HSC SEN processing, request rejected.
24	N/A
28	The HSC determined that the SLSXSEN macro is being invoked by a program that does not meet the authorization requirements specified in “SEN Macro Interface Authorization Requirements and Module Attributes” on page 323.
32	HSC is not active or at the proper initialization level for SEN services.
36	N/A

The SEN Listener Exit Routine

To use the SEN, you must write a listener exit routine that is invoked when the requested event(s) occur. The following sections describe the requirements and implementation of this exit.

Input Registers

Registers on entry to the SEN listener routine are as follows:

- R1 points to the address of input parameter list (mapped by macro SWSPGMIA) or zero.
- R13 points to the address of register save area.
- R15 points to the listener routine entry point address.
- All other registers are undefined.

On entry, Register 1 always contains the address of the SWSPGMIA data area except during SEN event registration. When requesting an event, you must provide the address of a listener exit routine that is invoked whenever the specified event(s) occur (for more information, see “LNRADR=” on page 327). As part of this process, HSC invokes the listener exit routine to ensure the address provided is valid. When this validation call is made, Register 1 contains zeros, indicating validation is being performed. During validation, the listener routine can simply return control or perform any routine specific initialization. If the validation call is not successful, the SEN request fails.

Output Registers

Registers on exit from the SEN listener exit routine must be as follows:

- R13 points to the address of original register save area.
- All other registers are undefined (however R15 can be set to 4 upon return to instruct SEN to stop passing any further XML data. For more information, see “How to Tell SEN to Stop Calling Your Listener Routine During Event Processing” on page 338).

Entry Environment

On entry, the SEN listener exit receives control as follows:

- STATE - supervisor
- KEY - key 0
- AMODE: 24 or 31 bit
- LOCKS - none

Listener Exit Routine Programming Considerations

During the SEN registration process each successful request is queued in FIFO order. When a SEN defined event occurs all registered listener exits are invoked serially, in the order in which registration requests were received. A potential performance degradation can occur if a listener exit performs long running units of work. This is particularly critical because each SEN event causes listener exits to be invoked multiple times.

Warning: StorageTek **strongly recommend**s that you design “short running” listener exit routines. In addition, the use of facilities that can result in “wait conditions”, such as WAIT/POST, STIMER/STIMERM and file I/O, should be used with caution or avoided all together.

The SWSPGMIA Data Area

On entry to the SEN listener exit during event processing, Register 1 contains the address of a data area mapped by macro `SWSPGMIA`. This storage area contains pointers to the XML defined elements describing each SEN defined event. For more information on the XML structures and tags, see “VTCS and HSC Events XML Tags” on page 345.

The listener exit XML interface basically consists of three types of tag definitions:

- Structure start tags
- Structure end tags
- Data tags and associated data

Structure start tags are passed to the listener exit, without any other data, and are used to indicate a set of logically grouped tags will follow.

Data tags (both start and end) along with the associated data are self defining XML elements.

Structure end tags are passed to the listener exit, without any other data, and are used to indicate the set of logically grouped tags has ended.

The following fields of `SWSPGMIA` support these XML tag structures:

`PGMIRSVA`
: Address of complete XML element being passed.

`PGMIRSVL`
: Length of the start tag plus, optionally, data length plus end tag length.

`PGMIRSTA`
: Address of XML start or end tag.

`PGMIRSTL`
: Length of the XML tag.

`PGMIRSDA`
: Address of associated data, or zero.

`PGMIRSDL`
: Length of associated data, or zero.

Fields PGMIRSVA, PGMIRSVL, PGMIRSTA, and PGMIRSTL always contain values. PGMIRSVA contains the address of either a structure start tag, a structure end tag or a data start tag. If PGMIRSVA points to a structure start or end tag, PGMIRSVL contains the length of the tag (including both the leading and trailing ‘<‘,’>’). If PGMIRSVA points to a data tag, PGMIRSVL contains the length of the tag (including both the leading and trailing ‘<‘,’>’), plus the length of the data, plus the length of the data end tag.

Fields PGMIRSTA and PGMIRSTL always contain values. PGMIRSTA contains the address of either a structure start tag or a structure end tag. PGMIRSTL contains the length of the tag (including both the leading and trailing ‘<‘,’>’),

Fields PGMIRSDA and PGMIRSDL optionally contain values. If PGMIRSVA points to either a structure start or structure end tag, PGMIRSDA and PGMIRSDL will contain zero since there is no data associated with these tags. But if PGMIRSVA points to a data start tag, then PGMIRSDA will contain the address of the associated data, and PGMIRSDL will contain the data’s length. The following sections show examples of how the listener exit routine processes XML structures and tags.

Example 1. XML tag `<libvol_insert_event>` is passed to the listener exit, which produces the following:

- PGMIRSVA points to: `<libvol_insert_event>`.
- PGMIRSVL contains: 00000015.
- PGMIRSTA points to: `<libvol_insert_event>`.
- PGMIRSTL contains: 00000015.

Because `<libvol_insert_event>` is a structure tag and has no related data:

- PGMIRSDA contains: 0.
- PGMIRSDL contains: 0.

Example 2. XML tag `<hsc_version>5.0.0</hsc_version>` is passed to the listener exit, which produces the following:

- PGMIRSVA points to: `<hsc_version>`.
- PGMIRSVL contains: 00000020.
- PGMIRSTA points to: `<hsc_version>`.
- PGMIRSTL contains: 0000000D.

Since <hsc_version> is a data start tag, it does have related data. As a result:

- PGMIRSDA points to: 5.0.0.
- PGMIRSDL contains: 00000005.

As described in “Listener Exit Routine Programming Considerations” on page 336, each SEN defined event causes multiple invocations of registered listener exit routines. For example, when an HSC insert volume event occurs the following XML structure is built. The listener routine will be invoked for each of the tag sets:

`<libvol_insert_event>` (passed on the 1st call to listener exit)

`<header>` (passed on 2nd call to the listener exit)

`<hsc_version>5.0.0</hsc_version>` (passed on 3rd call to the listener exit)

`<date>2002Mar19</date>` (passed on 4th call to the listener exit)

`<time>17:53:17</time>` (passed on 5th call to the listener exit)

`<host_name>HOSTA</host_name>` (passed on 6th call to the listener exit)

`</header>` (passed on 7th call to listener exit)

`<libvol_data>` (passed on 8th call to listener exit)

`<vol_status>xx</vol_status>` (passed on 9th call to listener exit, `xx` is a 1 byte flag field documented in member `SLSUX06P` of the `SLSMAC` installation file, field `UX06FLGS`)

`<volser>VOL001</volser>` (passed on 10th call to listener exit)

`<volume_location>` (passed on 11th call to listener exit)

`<acs>00</acs>` (passed on 12th call to listener exit)

`<lsm>000</lsm>` (passed on 13th call to listener exit)

`<panel>07</panel>` (passed on 14th call to listener exit)

`<row>02</row>` (passed on 15th call to listener exit)

`<column>10</column>` (passed on 16th call to listener exit)

`</volume_location>` (passed on 17th call to listener exit)

`<select_count>0</select_count>` (passed on 18th call to listener exit)

`</libvol_data>` (passed on 19th call to listener exit)

`</libvol_insert_event>` (passed on 20th and final call to listener exit)

How to Tell SEN to Stop Calling Your Listener Routine During Event Processing

Because listener routines are invoked to process each XML tag for the event, the data required by the listener may be satisfied before the last tag for the event is passed. For example, your listener routine only requires ACS and LSM number when a cartridge is entered into an LSM. Using “Example 2” on page 337, the listener routines requirements are satisfied on the 13th call. In this situation, a listener routine can set general purpose register 15 to 4 prior to returning control. This instructs SEN not to call it any more for this event. The remaining XML tags are bypassed.

Detecting Duplicate Listen Requests

When an application attempts to register a Listen request with the SEN facility, HSC first validates the request. As part of this validation, HSC insures duplicate registration requests are not accepted to prevent the same user exit routine from being called multiple times for a single event. The following list differentiates between a unique and a duplicate registration request:

- If a new Listen request specifies an exit routine entry point address that is different than all other previously registered entry point addresses, the Listen request is accepted.
- If a new Listen request specifies an exit routine entry point address that matches a previously registered entry point address but their respective event lists contain no common events (that is, each Listen request specified a unique event list), the Listen request is accepted.
- If a new Listen request specifies an exit routine entry point address that matches a previously registered entry point address and their respective event lists contain all common events (that is, each Listen request specified identical event lists), the Listen request is rejected as an identical duplicate and terminates with a return code of 36.
- If a new Listen request specifies an exit routine entry point address that matches a previously registered entry point address and their respective event lists contain some common events but not all (that is, each Listen request specified partially matching event lists), the Listen request is rejected as a duplicate with conflicting event lists and terminates with a return code of 8.

Display SEN Command

You can use the `Display SEN` command to display the SEN `LISTEN` requests and their statuses. The display can either show a specific request, using the requestor and listener names, a list of related requests using only the requestor name, or all SEN requests. The information displayed includes:

- Requestor name and listener name if any.
- EOT/EOM settings.
- Disable/enable status.
- The token associated with the request.
- The event list of the request.

Syntax

```
►►─Display SEN──────────────┬──────────────────────────┬──────────────►◄
                            └─REQNAME=────LNRNAME=─────┘
```

Figure 149.Display SEN Syntax

Parameters

`REQNAME=`
: Specifies the name of the requestor, which is a unique name eight byte alphanumeric field, and is used to uniquely identify a specify request for display purposes.Valid values are any RX-type address or registers 2 through 12. This parameter is optional if `LNRNAME` is not specified and has no default.

`LNRNAME=`
: Specifies the name of the listener routine The name must be an eight byte alphanumeric field. Valid values are any RX-type address or registers 2 through 12. If `LNRNAME` is specified, `REQNAME` must also be specified. Otherwise, this parameter is optional and has no default.

Note: `REQNAME=` and `LNRNAME=` are optional. If you specify one, however, you must specify the other.

Supported HSC and VTCS SEN Events

HSC Events

Table 71 describes the supported HSC SEN events.

Table 71. ***HSC SEN Events***

Event Name	Numeric Equate
libvol_insert_event	18
libvol_delete_event	19
hsc_termination_event	20
ACS_added_event	23
ACS_removed_event	24
lsmrail_added_event	25
lsmrail_removed_event	26
libdrive_added_event	27
libdrive_removed_event	28

VTCS SEN Events

Table 72 describes the supported VTCS SEN events.

Table 72. ***VTCS SEN Events***

Event Name	Numeric Equate
vtss_performance_event	01
vtss_chanif_performace_event	02
vtv_mount_event	03
vtv_dismount_event	04
vtv_delete_event	05
mvc_mount_event	06
mvc_dismount_event	07
vtv_migrate_event	08
vtv_recall_event	09
rtd_performance_event	10
rtd_vary_event	11
mvc_usage_event	12
vtv_movement_event	13
vtv_scratch_event	14

Table 72. ***VTCS SEN Events***

Event Name	Numeric Equate
vtss_performance_event	01
vtv_replicate_event	15
vtv_unlink_from_mvc_event	16
clink_vary_event	17

SEN Messages

SLS4970I `SEN QUEUE DISPLAY STATUS CURRENT REQUESTORS` *`99999999`* `REQNAME` *`RRRRRRRR`* `LNRNAME` *`LLLLLLLL`* `TOKEN` *`TTTTTTTT`* `FLAGS` *`FFFFFFFF`* `EVENT` *`EEEEEEEE`*

Explanation: This is the output from the Display SEN Queue command. The output produced varies depending on the parameters specified. If no parameters are specified all entries in the SEN QUEUE are displayed. If REQname is specified then only Requestors matching REQname will be displayed. If LNRname is specified REQname must be specified. If LNRname is specified then only Requestors and Listeners matching REQname and LNRname will be displayed.

99999999 Number of Requestors on the SEN QUEUE *RRRRRRRR* Requestor name

LLLLLLLL Listener name

TTTTTTTT Hex display of the Requestor Token

FFFFFFFF Description of the processing flags for the SEN QUEUE ENTRY

EEEEEEEE Description of the Event to be listened for.

System Action: None.

User Response: None.

SLS4971I `Significant Event Notification facility not active`

Explanation: An attempt was made to display SEN Queue entries. However, Significant Event Notification facilitate is not active.

System Action: The DISPLAY SEN command is rejected.

User Response: Make certain you are running the correct version of HSC. If you are certain you are running the correct version contact StorageTek HSC support.

SLS4972I `SEN QUEUE IS EMPTY`

Explanation: An attempt was made to display SEN Queue entries. However, the SEN QUEUE contained no entries.

System Action: Processing continues.

User Response: Make certain you have run the task to add entries to the SEN QUEUE. If the task has run corselette contact StorageTek HSC support.

`SLS4973I` `REQNAME` *nnnnnnnn* `NOT FOUND`

Explanation: An attempt was made to display a specific Requestor on the SEN QUEUE. However, the specified Requestor was not found.

System Action: Processing continues.

User Response: Make certain you have spelled the Requestor name correctly.

`SLS4974I` `LNRNAME` *nnnnnnnn* `NOT FOUND`

Explanation: An attempt was made to display a specific Listener on the SEN QUEUE. However, the specified Listenor was not found.

System Action: Processing continues.

User Response: Make certain you have spelled the Listener name correctly.

VTCS and HSC Events XML Tags

This section describes the XML output of the VTCS and HSC events processed by the HSC Significant Event Notification (SEN) facility as follows:

- Listing of the data tags.
- Tag definitions.
- Cross reference of data tags to structure or event tags.

For more information, see the following sections:

- “VTCS Events XML Tags” on page 346
- “HSC Events XML Tags” on page 350

VTCS Events XML Tags

Table 73. VTCS Events XML Tags

XML Tag	Definition	Where Used
<addresses_trapped>	From SMF11NAT in turn from NOADRTRP parm returned from Channel Interface Performance ECAM-T request. Contains the number of unit addresses trapped by a channel interface.	<chanif_data>
<async_end_tod>	Structure that contains date and time that the asynchronous event completed. Date is supplied as: <date>yyyymmmdd</date> Time is supplied as: <time>hh:mm:ss.thm</time>	<vtv_mount_event>
		<vtv_dismount_event>
		<mvc_mount_event>
		<vtv_migrate_event>
		<vtv_recall_event>
		<vtv_replicate_event>
<async_start_tod>	Structure that contains date and time that the asynchronous event started. Contains <date> and <time> as in <async_end_tod>.	<vtv_mount_event>
		<vtv_dismount_event>
		<mvc_mount_event>
		<vtv_migrate_event>
		<vtv_recall_event>
		<vtv_replicate_event>
<base_cache_size>	Cache size in MB of VTSS	<vtss_performance_event>
<bytes_read_freespace_collection>	Bytes read for free space collection within a partition.	<part_data>
<bytes_xfered_read>	Bytes transferred for read	<rtd_preformance_event>

Table 73. VTCS Events XML Tags

XML Tag	Definition	Where Used
<bytes_xfered_write>	Bytes transferred for write	<rtd_preformance_event>
<chanif_data>	Structure that contains all the information for a single channel IF. Contains:	<vtss_chanif_performance_event>
	<name>	
	<installed>	
	<enabled>	
	<link_type>	
	<addresses_trapped>	
	<speed>	
	<io_count>	
	<cu_busy_count>	
<collected_free_backend_capacity>	Bytes read for free space collection with the VTSS partition.	<part_data>
<cu_busy_count>	Count of control unit busy events on a chan IF	<chanif_data>
<customer_cache_size>	Customer cache size in MB	<vtss_performance_event>
<dev_activity>	RTD activity (initial selects)	<rtd_preformance_event>
<dev_available_time>	RTD mounted time	<rtd_preformance_event>
<dev_connect_time>	RTD connected time	<rtd_preformance_event>
<dev_util_time>	RTD utilized time	<rtd_preformance_event>
<dsname>	Data set name of the VTV being mounted/dismounted	<vtv_mount_event>
		<vtv_dismount_event>
<ecam_bypassed_bufferspace_count>	Count of ECAM-T requests bypassed – no buffer space	<vtss_performance_event>
<ecam_bypassed_configbusy_count>	Count of ECAM-T requests bypassed – configuration busy	<vtss_performance_event>
<ecam_processed_count>	Count of ECAM-T requests processed	<vtss_performance_event>
<enabled>	Indicates if a channel if is enabled - contains YES/NO	<chanif_data>
<installed>	Indicates if a channel if installed - contains YES/NO	<chanif_data>

Table 73. VTCS Events XML Tags

XML Tag	Definition	Where Used
<io_count>	Number of I/Os on this channel interface since last report.	<chanif_data>
<jobname>	MSP jobname requesting the mount/dismount	<vtv_mount_event>
		<vtv_dismount_event>
<link_type>	Identifies the link type contains HOST/RTD	<chanif_data>
<mount_type>	Indicates the type of mount performed. Contains:	<vtv_mount_event>
	EXISTING-SPECIFIC	
	CREATE-SL	
	CREATE-AL	
	EXISTING-SCRATCH	
<name>	The configured name of the channel IF.	<chanif_data>
<new_mvc>	Structure that describes the target MVC on a VTV move. Contains <mvc_data>	<vtv_movement_event>
<nvs_size>	NVS Size (MB)	<vtss_performance_event>
<offline_cache_size>	Offline cache size	<vtss_performance_event>
<old_mvc>	Structure that describes the source MVC on a VTV move. Contains <mvc_data>	<vtv_movement_event>
<part_data>	Structure containing all the information from a VTSS partition. Contains:	<vtss_performance_event>
	<total_backend_capacity>	
	<total_free_backend_capacity >	
	<collected_free_backend_capa city	
	<bytes_read_freespace_collect ion>	
	<standard_capacity_defined>	

Table 73. VTCS Events XML Tags

XML Tag	Definition	Where Used
< name>	Name of a VTSS partition	<part_data>
<pinned_cache_size>	Pinned Cache Size	<vtss_performance_event>
<read_buffered_log>	64-byte rbl data represented as 128 characters	<mvc_dismount_event>
<read_only>	Indicates whether or not the VTV was mounted READONLY – contains YES/NO	<vtv_mount_event>
<reason>	The reason for the VTV delete, MVC mount, VTV migrate, VTV recall or the VTV/MVC unlink.	<vtv_delete_event>
		<mvc_mount_event>
		<vtv_migrate_event>
		<vtv_recall_event>
		<vtv_unlink_from_mvc_event>
<recall_required>	Indicates whether or not a recall was required in order to mount the VTV – contains YES/NO	<vtv_mount_event>
<recall_with_error>	Indicates if the recall was performed with error – contains YES/NO	<vtv_recall_event>
<sense_data>	32-byte sense data represented as 64 characters	<mvc_mount_event>
		<mvc_dismount_event>
		<vtv_recall_event>
		<vtv_replicate_event>
<speed>	The speed of a channel IF.	<chanif_data>
<standard_capacity_ defined>	Standard capacity defined for the partition	<part_data>
<stepname>	MSP stepname of the job requesting the mount/dismount of the VTV	<vtv_mount_event>
		<vtv_dismount_event>
<total_backend_capacity>	Total backend capacity for a partition	<part_data>

Table 73. VTCS Events XML Tags

XML Tag	Definition	Where Used
<total_free_backend_ capacity>	Total free backend capacity for a partition.	<part_data>

HSC Events XML Tags

Table 74. HSC Events XML Tags

Head Tag	Structure/Data Tags			Definition
<header>				header structure, occurs in: <libvol_ insert_ event> <libvol_ delete_ event> <hsc_ termination_ event> **Note:** Any x22 abends resulting from the cancellation of HSC will not generate the HSC termination event. <libdrive_ added_event> <libdrive_ removed_ event> <lsmrail_ added_event> <lsmrail_ removed_ event>
	<hsc_version>			HSC Version
	<date>			date

Table 74. HSC Events XML Tags

Head Tag	Structure/Data Tags			Definition
	<time>			time
	<host_name>			host name
<libdrive_added_event>				dynamically added drive event
	<libdrive_data>			drive data
		<location_data>		drive location
			<acs>	ACS
			<lsm>	LSM
			<panel>	panel
			<libdrive_number>	drive number
		<libdrive_model>		drive model (for example, 9840C)
		<libdrive_unit_address>		MSP unit address

Table 74. HSC Events XML Tags

Head Tag	Structure/Data Tags			Definition
<libdrive_removed_event>				dynamically removed drive event
	<libdrive_data>			drive data
		<location_data>		drive location
			<acs>	ACS
			<lsm>	LSM
			<panel>	panel
			<libdrive_number>	drive number
		<libdrive_model>		drive model (for example, 9840C)
		<libdrive_unit_address>		MSP unit address

Table 74. HSC Events XML Tags

Head Tag	Structure/Data Tags			Definition
<lsmrail_added_event>				dynamically added LSM rail event (SL8500 only)
	<lsmrail_data>			LSM rail data
		<location_data>		rail location
			<acs>	ACS
			<lsm>	LSM
<lsmrail_removed_event>				dynamically removed LSM rail event (SL8500 only)
	<lsmrail_data>			LSM rail data
		<location_data>		rail location
			<acs>	ACS
			<lsm>	LSM

Glossary

A

access method A technique for moving data between processor storage and input/output devices.

ACS *See* Automated Cartridge System.

ACSid A method used to identify an ACS. An ACSid is the result of defining the SLIALIST macro during the library generation (LIBGEN) process. The first ACS listed in this macro acquires a hexadecimal identifier of 00, the second ACS listed acquires a hexadecimal identifier of 01, and so forth, until all ACSs are identified.

ACS routine An SMS term, referring to automatic class selection routine. Not to be confused with the HSC term, ACS, referring to automatic cartridge system.

AMT automatic migration threshold.

APF Authorized Program Facility.

APPL VTAM APPLID definition for the HSC.

archiving The storage of backup files and associated journals, usually for a given period of time.

audit A VSM audit (which is not the same as an HSC audit) reconstructs VTV and MVC information.

Automated Cartridge System (ACS) The library subsystem consisting of one or two LMUs, and from 1 to 16 attached LSMs.

automated library *See* library.

automatic mode A relationship between an LSM and all attached hosts. LSMs operating in automatic mode handle cartridges without operator intervention. This is the normal operating mode of an LSM that has been modified online.

automatic migration Migrating VTVs to MVCs that is automatically initiated and controlled by VSM.

automatic migration threshold (AMT) AMT values are percentage values that determine when virtual tape volume migration begins and ends. VTV migration begins when the VTSS buffer reaches the high AMT and ends when the buffer reaches or falls below the low AMT. These thresholds apply to all VTSSs.

automatic recall Recalling VTVs to the VTSS that is automatically initiated and controlled by VSM.

automatic reclaim Reclaiming MVC space that is automatically initiated and controlled by VSM.

B

block A collection of contiguous records recorded as a unit. Blocks are separated by interblock gaps, and each block may contain one or more records.

buffer A routine or storage used to compensate for a difference in rate of data flow, or time of occurrence of events, when transferring data from one device to another.

C

CA–1 (TMS) Computer Associates Tape Management System. Third–party software by Computer Associates International, Inc.

CAP *See* Cartridge Access Port.

capacity *See* media capacity.

CAPid A CAPid uniquely defines the location of a CAP by the LSM on which it resides. A CAPid is of the form *AAL:CC* where *AA* is the ACSid, *L* is the LSM number, and *CC* is the CAP number. Some commands and utilities permit an abbreviated CAPid format of *AAL*.

cartridge The plastic housing around the tape. It is approximately 4 inches (100 mm) by 5 inches (125 mm) by 1 inch (25 mm). The tape is threaded automatically when loaded in a transport. A plastic leader block is attached to the tape for automatic threading. The spine of the cartridge contains a Tri–Optic label listing the VOLSER (tape volume identifier).

Cartridge Access Port (CAP) An assembly which allows an operator to enter/eject cartridges during automated operations. The CAP is located on the access door of an LSM. (*see also,* standard CAP, enhanced CAP, WolfCreek CAP, WolfCreek optional CAP.)

Cartridge Scratch Loader An optional feature for the Cartridge Drive. It allows the automatic loading of premounted tape cartridges or the manual loading of single tape cartridges.

cartridge system tape The basic tape cartridge media that is used with 4480, 4490, or 9490 Cartridge Subsystems. They are visually identified by a one–color cartridge case.

CAW *See* Channel Address Word.

CDRM Cross Domain Resource Manager definition (if not using existing CDRMs).

CDRSC Cross Domain Resource definition.

CDS *See* control data set.

CE Channel End.

cell A storage slot in the LSM that is used to store a tape cartridge.

Central Support Remote Center (CSRC) *See* Remote Diagnostics Center.

CFT Customer field test.

channel A device that connects the host and main storage with the input and output control units.

Channel Address Word (CAW) An area in storage that specifies the location in main storage at which a channel program begins.

channel command A command received by a CU from a channel.

Channel Status Word (CSW) An area in storage that provides information about the termination of input/output operations.

check Detection of an error condition.

CI Converter/Interpreter (JES).

Clink (cluster link). The path between a primary VTSS and secondary VTSS in a cluster. The Clink path is used to copy replicate VTVs from the primary to the secondary.

Cluster. Two VTSSs which are physically cabled together by Clink paths and are defined in CONFIG as a cluster. A cluster consists of a primary and a secondary VTSS. VTVs with the replicate attribute attached will be copied from the primary to the secondary as soon as possible after dismount time.

connected mode A relationship between a host and an ACS. In this mode, the host and an ACS are capable of communicating (at least one station to this ACS is online).

control data set (CDS) The HSC database. In addition to the current information in the CDS, VSM keeps all its persistent data in the CDS as well.

control data set allocation map A CDS subfile that marks individual blocks as used or free.

control data set data blocks CDS blocks that contain information about the library and its configuration or environment.

control data set directory A part of the CDS that maps its subdivision into subfiles.

control data set pointer blocks CDS blocks that contain pointers to map data blocks belonging to a subfile.

control data set recovery area A portion of the CDS reserved for maintaining integrity for updates that affect multiple CDS blocks.

control data set subfile A portion of the CDS consisting of Data Blocks and Pointer Blocks containing related information.

Control Unit (CU) A microprocessor–based unit situated logically between a host channel (or channels) and from two to sixteen tape transports. It functions to translate channel commands into tape transport commands, send transport status to the channel(s), and pass data between the channel(s) and transport(s).

conventional Nearline transport An HSC–controlled transport that is not defined to VSM as an RTD.

cross–host recovery The ability for one host to perform recovery for another host that has failed.

CSE Customer Service Engineer.

CSI Consolidated System Inventory.

CSL Cartridge Scratch Loader.

CSRC Central Support Remote Center (*See* Remote Diagnostics Center)

CSW Channel Status Word.

CU *See* Control Unit.

D

DAE Dump Analysis Elimination.

DASD Direct access storage device.

data Any representations such as characters or analog quantities to which meaning is, or might be, assigned.

data class A collection of allocation and space attributes, defined by the storage administrator, that are used to create a data set.

data compaction An algorithmic data–reduction technique that encodes data from the host and stores it in less space than unencoded data. The original data is recovered by an inverse process call decompaction.

data–compaction ratio The number of host data bytes divided by the number of encoded bytes. It is variable depending on the characteristics of the data being processed. The more random the data stream, the lower the opportunity to achieve compaction.

Data Control Block (DCB) A control block used by access routines in storing and retrieving data.

data set The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

data streaming A continuous stream of data being transmitted in character or binary–digit form, using a specified format.

DBU disk buffer utilization.

DCB Data Control Block.

demand allocation An MSP term meaning that a user has requested a specific unit.

demand migration Migrating VTVs to MVCs that an administrator does with the MIGRATE command or utility.

demand recall Recalling VTVs to the VTSS that an administrator does with the RECALL command or utility.

demand reclaim Reclaiming MVC space that an administrator does with the RECLAIM command or utility.

device number A four–digit hexadecimal number that uniquely identifies a device attached to a processor.

device separation The HSC function which *forces* the MSP device selection process to choose either a nonlibrary transport or a transport in a particular ACS, based on the location of the volume (specific requests) or the given subpool rules in effect (nonspecific request).

DFP Data Facility Product. A program that isolates applications from storage devices, storage management, and storage device hierarchy management.

DFSMS Refers to an environment running MSP/ESA SP and DFSMS/MSP, DFSORT, and RACF. This environment helps automate and centralize the management of storage through a combination of hardware, software, and policies.

DFSMS ACS routine A sequence of instructions for having the system assign data class, storage class, management class, and storage group for a data set.

directed allocation The HSC function of *influencing* MSP's selection of library transports. For a specific request, the HSC influences MSP to choose a transport requiring the fewest number of pass–thrus; for a nonspecific (scratch) request, HSC's influencing is based on the given subpool rules in effect.

disconnected mode A relationship between a host and an ACS. In this mode, the host and an ACS are not capable of communicating (there are no online stations to this ACS).

disk buffer utilization (DBU). The ratio of used to total VTSS buffer capacity.

DOMed Pertaining to a console message that was previously highlighted during execution, but is now at normal intensity.

drain The deletion of data from an MVC. May be accompanied by a “virtual” eject to prevent the MVC from being reused.

drive loaded A condition of a tape drive in which a tape cartridge has been inserted in the drive, and the tape has been threaded to the beginning–of–tape position.

DSI Dynamic System Interchange (JES).

dual LMU A hardware/u–software feature that provides a redundant LMU capability.

dual LMU HSC release 1.1.0 or later that automates a switchover to the standby LMU in a dual LMU configuration.

dump To write the contents of storage, or of a part of storage, usually from an internal storage to an external medium, for a specific purpose such as to allow other use of storage, as a safeguard against faults or errors, or in connection with debugging.

Dynamic Device Reconfiguration (DDR) A facility that allows a demountable volume to be moved, and repositioned if necessary, without abnormally terminating the job or repeating the initial program load procedure.

E

Ecart Cartridge system tape with a length of 1100 feet that can be used with 4490 cartridge drives. These tapes are visually identified by a two–tone colored case.

EDL *See* eligible device list.

eligible device list A group of tape drives that are available to satisfy an allocation request.

enhanced CAP An enhanced CAP contains two forty–cell magazine–style CAPs and a one–cell priority CAP (PCAP). Each forty–cell CAP holds four removable magazines of ten cells each. An LSM access door with an enhanced CAP contains no cell locations for storing cartridges. An enhanced CAP is ordered as Feature Number CC80. (*see also,* Cartridge Access Port (CAP), standard CAP, WolfCreek CAP, WolfCreek optional CAP.)

Effective Recording Density The number of user bytes per unit of length of the recording medium.

eject The LSM robot places a cartridge in a Cartridge Access Port (CAP) so the operator can remove it from the LSM.

ExPR Expert Performance Reporter.

Expert Performance Reporter Expert Performance Reporter collects performance data and generates reports about StorageTek Nearline ACSs and VTSS status and performance. It has an MSP component and a PC component.

Enhanced Capacity Cartridge System Tape Cartridge system tape with increased capacity that can be used with 4490 and 9490 Cartridge Drives. These tapes are visually identified by a two–tone colored case.

EOT End–of–Tape marker.

EPO Emergency Power Off.

ERDS Error Recording Data Set.

EREP Environmental Recording, Editing, Printing.

ERP Error recovery procedures.

error recovery procedures (ERP) Procedures designed to help isolate and, where possible, to recover from errors in equipment.

ExtendedStore Library One or more LSMs with no cartridge drives (CDs) that are attached by pass–thru ports to other LSMs (with CDs) in an ACS. These LSMs provide archive storage for cartridges containing less active data sets. Cartridges can be entered and ejected directly into and out of this LSM though either a standard CAP or an enhanced CAP.

F

file protected Pertaining to a tape volume from which data can be read only. Data cannot be written on or erased from the tape.

format The arrangement or layout of data on a data medium.

G

GB 1,073,741,824 bytes of storage.

GDG Generation Data Group. An MSP data set naming convention. Sequence numbers are appended to the basic data set name to track the generations created for that data set.

GTF Generalized Trace Facility. An MSP facility used to trace software functions and events.

H

HDA Head/disk assembly.

Host Software Component (HSC) That portion of the Automated Cartridge System which executes on host systems attached to an automated library. This component acts as the interface between the operating system and the rest of the automated library.

host system A data processing system that is used to prepare programs and the operating environments for use on another computer or controller.

HSC Host Software Component.

HSM Hierarchical Storage Manager.

HWS High Watermark Setup. Relates to chains set up for tape transport allocation in JES3.

I

ICRC See Improved Cartridge Recording Capability.

Improved Cartridge Recording Capability (ICRC) An improved data recording mode that, when enabled, can increase the effective cartridge data capacity and the effective data rate when invoked.

ID Identifier or identification.

IDAX Interpreter Dynamic Allocation Exit. This is a subfunction of the DFSMS/MSP subsystem request (SSREQ 55) that the MSP JCL Interpreter and dynamic allocation functions issue for calling DFSMS ACS routines for management of the data set requested.

IML *See* Initial Microprogram Load.

index a function performed by the cartridge loader that moves cartridges down the input or output stack one cartridge position. A loader can perform multiple consecutive indexes.

Initial Microprogram Load (IML) A process that activates a machine reset and loads system programs to prepare a computer system for operation. Processors having diagnostic programs activate these programs at IML execution. Devices running u–software reload the functional u–software usually from a floppy diskette at IML execution.

Initial Program Load (IPL) A process that activates a machine reset and loads system programs to prepare a computer system for operation. Processors having diagnostic programs activate these programs at IPL execution. Devices running u–software reload the functional u–software usually from a floppy diskette at IPL execution.

initial value A value assumed until explicitly changed. It must then be explicitly specified in another command to restore the initial value. An initial value for the HSC is the value in effect when the product is installed.

inline diagnostics Diagnostic routines that test subsystem components while operating on a time–sharing basis with the functional u–software in the subsystem component.

input stack The part of the cartridge loader where cartridges are premounted.

intervention required Manual action is needed.

ips Inches per second.

IVP Installation Verification Programs. A package of programs that is run by a user after the library is installed in order to verify that the library is functioning properly.

J

JCL See Job Control Language.

Job Control Language Problem–oriented language designed to express statements in a job that are used to identify the job or describe its requirements to an operating system.

journal The log associated with journaling. The log (stored in a data set) contains a record of completed work and changes to the control data set since the last backup was created.

journaling A technique for recovery that involves creating a backup control data set and maintaining a log of all changes (transactions) to that data set.

K

KB Kilobyte, thousand bytes, or 1024 bytes.

kb kilobit, or thousand bits (10^3 bits).

keyword parameter In command and utility syntax, operands that include keywords and their related values (*see* "positional parameter"). Values are concatenated to the keyword either by an equal sign, "KEYWORD=value," or by parentheses, "KEYWORD(value)." Keyword parameters can be specified in any order. The HSC accepts (tolerates) multiple occurrences of a keyword. The value assigned to a keyword reflects the last occurrence of a keyword within a command.

L

LAN Local Area Network.

LCU *See* Library Control Unit.

LED *See* Light Emitting Diode.

LIBGEN The process of defining the configuration of the automated library to the host software.

library An installation of one or more ACSs, attached cartridge drives, volumes placed into the ACSs, host software that controls and manages the ACSs and associated volumes, and the library control data set that describes the state of the ACSs.

library control data set *See* control data set.

Library Control Unit (LCU) The portion of the LSM that controls the picking, mounting, dismounting, and replacing of cartridges.

Light Emitting Diode (LED) An electronic device used mainly as an indicator on status panels to show equipment on/off conditions.

LMU Library Management Unit. The portion of the ACS that manages from one to sixteen LSMs and communicates with the host CPU.

loader *See* Cartridge Scratch Loader.

load point The beginning of the recording area on magnetic tape.

Local Area Network (LAN) A computer network in which devices within the network can access each other for data transmission purposes. The LMU and attached LCUs are connected with a local area network.

logical ejection The process of removing a volume from the control data set without physically ejecting it from its LSM location.

LSM Library Storage Module. Provides the storage area for cartridges plus the robot necessary to move the cartridges. The term LSM often means the LCU and LSM combined.

LSMid An LSMid is composed of the ACSid concatenated with the LSM number.

LSM number A method used to identify an LSM. An LSM number is the result of defining the SLIACS macro LSM parameter during a LIBGEN. The first LSM listed in this parameter acquires the LSM number of 0 (hexadecimal), the second LSM listed acquires a hexadecimal number of 1, and so forth, until all LSMs are identified (maximum of sixteen or hexadecimal F).

M

machine initiated maintenance See ServiceTek.

magnetic recording A technique of storing data by selectively magnetizing portions of a magnetizable material.

magnetic tape A tape with a magnetizable surface layer on which data can be stored by magnetic recording.

magnetic tape drive A mechanism for moving magnetic tape and controlling it's movement.

maintenance facility Hardware contained in the CU and LMU that allows a CSE and the RDC to run diagnostics, retrieve status, and communicate with respective units through their control panels.

management class A collection of management attributes, assigned by the storage administrator, that are used to control the allocation and use of space by a data set. Note that SMS Management Classes are different from VSM Management Classes.

manual mode A relationship between an LSM and all attached hosts. LSMs operating in manual mode have been modified offline and require human assistance to perform cartridge operations.

master LMU The LMU currently controlling the functional work of the ACS in a dual LMU configuration.

MDS Main Device Scheduler (JES3).

media capacity The amount of data that can be contained on storage media and expressed in bytes of data.

micro–software *See* υ –software under Symbols.

migration The movement of VTVs from the VTSS to the RTD where the VTVs are stacked onto MVCs. See *automatic migration* and *demand migration*.

MIM Multi–Image Manager. Third–party software by CA Corporation.

mixed configurations Installations containing cartridge drives under ACS control and cartridge drives outside of library control. These configurations cause the Host Software Component to alter allocation to one or the other.

modem Modulator/demodulator. An electronic device that converts computer digital data to analog data for transmission over a telecommunications line (telephone line). At the receiving end, the modem performs the inverse function.

monitor A device that observes, records, and verifies selected system activities to determine significant departure from expected operation.

Multi-Volume Cartridge (MVC) A physical tape cartridge residing in an LSM that either contains migrated virtual tape volumes (VTVs) or is identified as a volume that can be selected for VTV stacking.

MVCPool Statement An HSC control statement that is contained in the definition data set specified by the `VT MVCDEF` command. An `MVCPool` statement specifies the MVCs that VTCS uses.

MVCDEF An HSC command that is used to load the definition data set that contains `MVCPool` statements.

N

O

output stack The part of the cartridge loader that receives and holds processed cartridges.

P

paired–CAP mode The two forty–cell CAPs in an enhanced CAP function in paired–CAP mode as a single eighty–cell CAP.

PARMLIB control statements Parameter library (PARMLIB) control statements allow you statically specify various operation parameters which take effect at HSC initialization. Identifying your system requirements and then specifying the appropriate control statements permits you to customize the HSC to your data center.

Pass–Thru Port (PTP) A mechanism that allows a cartridge to be passed from one LSM to another in a multiple LSM ACS.

physical end of tape A point on the tape beyond which the tape is not permitted to move.

positional parameter In command and utility syntax, operands that are identified by their position in the command string rather than by keywords (*see* "keyword parameter"). Positional parameters must be entered in the order shown in the syntax diagram.

POST *See* Program for Online System Testing.

PowderHorn A high–performance LSM (model number 9310) featuring a high–speed robot. The PowderHorn has a capacity of up to approximately 6000 cartridges.

Primary. One of two VTSSs in a cluster which is designated in CONFIG as the primary. During normal operations the primary services the host workload and copies replicate VTVs to the secondary.

Program for Online System Testing (POST) A program in a host computer that allows it to test an attached subsystem while the subsystem is online.

Program Temporary Fix A unit of corrective maintenance delivered to a customer to repair a defect in a product, or a means of packaging a Small Programming Enhancement (SPE).

Program Update Tape A tape containing a collection of PTFs. PUTs are shipped to customers on a regular basis under the conditions of the customer's maintenance license.

PTF *See* Program Temporary Fix.

PTP *See* pass–thru port.

PUT *See* Program Update Tape.

R

RACF See Resource Access Control Facility.

Real Tape Drive (RTD) The physical transport attached to the LSM. The transport has a data path to a VTSS and may optionally have a data path to MSP or to another VTSS.

RDC *See* Remote Diagnostic Center.

recall The movement of VTVs from the MVC back to the VTSS. May be automatic or on demand.

reclaim Refers to MVC space reclamation. For automatic and demand reclamation, VTCS uses the amount of fragmented free space on the MVC and the amount of VTV data that would have to be moved to determine if space reclamation is justified.

Reconciliation. An automatic process initiated when a cluster is reestablished after the primary or secondary has been offline. Reconciliation ensures that the contents of the primary and secondary are identical with respect to replicate VTVs.

Recording Density The number of bits in a single linear track measured per unit of length of the recording medium.

Remote Diagnostic Center (RDC) The Remote Diagnostic Center at StorageTek. RDC operators can access and test StorageTek systems and software, through telecommunications lines, from remote customer installations. Also referred to as the Central Support Remote Center (CSRC).

Replication. Copying a replicate VTV from the primary VTSS to the secondary VTSS in a cluster. When replication completes, there are two copies of the VTV, one in the primary and one in the secondary.

Replicate VTV. A VTV which has had the replicate attribute attached to it by a management class statement.

Resource Access Control Facility (RACF) Security software controlling access to data sets.

RTD *See* real tape drive.

S

SCP See System Control Program.

scratch tape subpool A defined subset of all scratch tapes. Subpools are composed of one or more ranges of VOLSERs with similar physical characteristics (type of volume {reel or cartridge}, reel size, length, physical location, etc.). Some installations may also subdivide their scratch pools by other characteristics, such as label type (AL, SL, NSL, NL). The purpose of subpooling is to ensure that certain data sets are built only within particular ranges of volumes (for whatever reason the user desires). If a volume which does not belong to the required subpool is mounted for a particular data set, it is dismounted and the mount reissued.

Secondary. One of two VTSSs in a cluster which is designated in CONFIG as the secondary. During normal operations the secondary receives copies of replicate VTVs, stores them, and makes a migration copy on an MVC as soon as possible.

secondary recording A technique for recovery involving maintaining both a control data set and a copy (secondary) of the control data set.

SER Software Enhancement Request.

ServiceTek (machine initiated maintenance) A unique feature of the ACS in which an expert system monitors conditions and performance of subsystems and requests operator attention before a potential problem impacts operations. Customers can set maintenance threshold levels.

servo A device that uses feedback from a sensing element to control mechanical motion.

Small Programming Enhancement (SPE) A supplement to a released program that can affect several products or components.

SMF System Management Facility. An MSP facility used to record system actions which affect system functionality.

SMP System Modification Program.

SMP/E System Modification Program Extended.

SMS System Managed Storage.

SPE Small Programming Enhancement.

standard CAP A standard CAP has a capacity of twenty–one cartridges (three rows of seven cells each). An LSM access door with a standard CAP contains cell locations for storing cartridges. (*see also*, Cartridge Access Port (CAP), enhanced CAP.)

standard LSM A model 4410 LSM which a has a storage capacity of up to approximately 6000 cartridges.

standby The status of a station that has been varied online but is connected to the standby LMU of a dual LMU ACS.

standby LMU The redundant LMU in a dual LMU configuration that is ready to take over in case of a master LMU failure or when the operator issues the SWitch command.

station A hardware path between the host computer and an LMU over which the HSC and LMU send control information.

storage class A named list of storage attributes that identify performance goals and availability requirements for a data set. Note that SMS Storage Classes are different from VSM Storage Classes.

storage group A collection of storage volumes and attributes defined by the storage administrator. Note that this is an SMS concept, not a VSM concept.

switchover The assumption of master LMU functionality by the standby LMU.

System Control Program The general term to describe a program which controls access to system resources, and allocates those resources among executing tasks.

system–managed storage Storage that is managed by the Storage Management Subsystem, which attempts to deliver required services for availability, performance, space, and security applications.

System Modification Program Extended An IBM–licensed program used to install software and software maintenance.

T

tape cartridge A container holding magnetic tape that can be processed without separating it from the container.

tape drive A device that is used for moving magnetic tape and includes the mechanisms for writing and reading data to and from the tape.

TAPEREQ An HSC control statement that is contained in the definition data set specified by the TREQDEF command. A TAPEREQ statement defines a specific tape request. It is divided into two parts, the input: job name, step name, program name, data set name, expiration date or retention period, and an indication for specific requests or nonspecific (scratch) requests; and the output: media type and recording technique capabilities. You can use TAPEREQ statements to direct data sets to VSM.

tape unit A device that contains tape drives and their associated power supplies and electronics.

Timberwolf (9740) LSM A high performance LSM that provides a storage capacity of up to 494 cartridges. Up to 10 drives (STD, 4490, 9490, 9490EE, 9840, and SD-3) can be configured. Timberwolf LSMs can only attach to other Timberwolfs.

TMS Tape Management System.

TP Tape–to–Print.

transaction A short series of actions with the control data set. These actions are usually related to a specific function (e.g., Mount, ENter).

transport An electromechanical device capable of threading tape from a cartridge, moving the tape across a read/write head, and writing data onto or reading data from the tape.

TREQDEF An HSC command that is used to load the definition data set that contains TAPEREQ control statements.

Tri–Optic label An external label attached to the spine of a cartridge that is both human and machine readable.

TT Tape–to–Tape.

U

UNITATTR An HSC control statement that is contained in the definition data set specified by the UNITDEF command. A UNITATTR statement defines to the HSC the transport's media type and recording technique capabilities. For VSM, the UNITATTR statements define the VTD addresses to VSM as virtual and associate them with a VTSS.

UNITDEF An HSC command that is used to load the definition data set that contains UNITATTR control statements.

utilities Utility programs. The programs that allow an operator to manage the resources of the library and to monitor overall library performance.

V

Virtual Storage Manager (VSM) A storage solution that virtualizes volumes and transports in a VTSS buffer in order to improve media and transport use. The hardware includes VTSS, which is the DASD buffer, and RTDs. The software includes VTCS, an HSC–based host software, and VTSS microcode.

Virtual Tape Control System (VTCS) The primary host code that controls activity and information about VTSSs, VTVs, RTDs, and MVCs.

Virtual Tape Drive (VTD) An emulation of a physical transport in the VTSS that looks like a physical tape transport to MSP. The data written to a VTD is really being written to DASD. The VTSS has 64 VTDs that do virtual mounts of VTVs.

Virtual Tape Storage Subsystem (VTSS) The DASD buffer containing virtual volumes (VTVs) and virtual drives (VTDs). The VTSS is a STK RAID 6 hardware device with microcode that enables transport emulation. The RAID device can read and write "tape" data from/to disk, and can read and write the data from/to an RTD.

Virtual Tape Volume (VTV) A portion of the DASD buffer that appears to the operating system as a real tape volume. Data is written to and read from the VTV, and the VTV can be migrated to and recalled from real tape.

virtual thumbwheel An HSC feature that allows read–only access to a volume that is not physically write–protected.

VOLATTR An HSC control statement that is contained in the definition data set specified by the VOLDEF command. A VOLATTR statement defines to the HSC the media type and recording technique of the specified volumes. For VSM, the VOLATTR statements define the volsers for volumes that will be used as MVCs.

VOLDEF An HSC command that is used to load the definition data set that contains VOLATTR control statements.

VOLSER A six–character alphanumeric label used to identify a tape volume.

volume A data carrier that is mounted or demounted as a unit. (*See* cartridge).

VSM *See* Virtual Storage Manager.

VTCS *See* Virtual Tape Control System.

VTD *See* virtual tape drive.

W

WolfCreek A smaller capacity high–performance LSM. WolfCreek LSMs are available in 500, 750, and 1000 cartridge capacities (model numbers 9360–050, 9360–075, and 9360–100 respectively). WolfCreek LSMs can be connected by pass–thru ports to 4410, 9310, or other WolfCreek LSMs.

WolfCreek CAP The standard WolfCreek CAP contains a 20–cell magazine–style CAP and a priority CAP (PCAP). (*see also,* Cartridge Access Port (CAP), Enhanced CAP, standard CAP, WolfCreek optional CAP.)

WolfCreek optional CAP The WolfCreek optional CAP contains a 30–cell magazine–style CAP which is added to the standard WolfCreek CAP. (*see also,* Cartridge Access Port (CAP), Enhanced CAP, standard CAP, WolfCreek CAP.)

Write Tape Mark (WTM) The operation performed to record a special magnetic mark on tape. The mark identifies a specific location on the tape.

WTM *See* Write Tape Mark.

WTO Write–to–Operator.

WTOR Write–to–Operator with reply.

Symbols

υ **-software.** Microprogram. A sequence of microinstructions used to perform preplanned functions and implement machine instructions.

Numerics

4410 LSM *See* standard LSM.

9310 LSM *See* Powderhorn LSM.

9360 LSM See Wolfcreek *LSM.*

9490 Cartridge Subsystem Cartridge tape transports that provide read/write capability for 36–track recording format and extended capacity tape and provide improved performance over the 4490 Cartridge Subsystem. 9490 transports can also read data recorded in 18–track format. The StorageTek 9490 Cartridge Subsystem offers better performance (faster data transfer rate, faster load/unload) than a 3490E device.

9490EE Cartridge Subsystem A high performance tape transport that provides read/write capability for Extended Enhanced (EEtape) cartridges. It is functionally equivalent to the IBM 3490E device.

9**740 LSM** See Timberwolf *LSM.*

9840 Cartridge Subsystem A high performance tape transport system for Enterprise and Open Systems environments that reads and writes 9840 cartridges. 9840s can be defined in 10-drive and 20-drive panel configurations. The 9840 can perform as a stand-alone subsystem with a cartridge scratch loader installed, or it can be attached to a StorageTek ACS.

Index

M

O

P

Q

R

S

T

U

V

W

NEED MORE INFORMATION?

www.storagetek.com

ABOUT STORAGETEK

Storage Technology Corporation (NYSE: STK) is a $2 billion global company that enables businesses, through its information lifecycle management strategy, to align the cost of storage with the value of information. The company's innovative storage solutions manage the complexity and growth of information, lower costs, improve efficiency and protect investments. For more information, visit www.storagetek.com, or call 1.800.275.4785 or 01.303.673.2800.

WORLD HEADQUARTERS

Storage Technology Corporation
One StorageTek Drive
Louisville, Colorado 80028 USA
1.800.678.4430 or 01.303.673.4430