



Release Notes for StorHouse Release 5.6

Publication Number
900029 Rev. P
July 22, 2004

The FileTek logo consists of the word "FileTek" in white, sans-serif font, centered within a teal square. The square is positioned on the right side of a horizontal line that spans the width of the page.

FileTek



No part of this publication may be reproduced, translated, stored in any electronic retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of FileTek, Inc.

This publication Copyright ©2004 FileTek, Inc. As an Unpublished Licensed Work. All Rights Reserved. Publication Number: 900029 Rev. P

NOTICE: U.S. GOVERNMENT USERS

This notice applies to all acquisitions of this work by or for the U.S. Government ("Government"), or by any prime contractor or subcontractor (at any tier) under any contract, cooperative agreement or other activity with the Government. By accepting delivery of this work, the Government agrees that this work and the Licensed Program(s) described herein qualify as "commercial" computer software within the meaning of the acquisition regulation(s) applicable to this procurement. The terms of conditions of the license for the Licensed Program(s) shall pertain to the Government's use and disclosure of this work and the Licensed Program(s), and shall supersede any conflicting contractual terms or conditions. If the license for this work and the Licensed Program(s) fails to meet the Government's need or is inconsistent in any respect with Federal law, the Government agrees to return this work and the Licensed Program(s), unused, to FileTek, Inc. The following additional statement applies only to acquisitions governed by DFARS Subpart 227.4 (October 1988) "Restricted Rights – Use, duplication and disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 (OCT. 1988)." Unpublished licensed work property of FileTek, Inc. Unauthorized use, duplication or distribution prohibited. All rights reserved. A copyright notice on this work and/or on the Licensed Program(s) by itself does not constitute publication or public disclosure of this work or the Licensed Program(s). The contractor/manufacturer is:

FileTek, Inc.
9400 Key West Avenue
Rockville, Maryland 20850

Information in this document is subject to change without notice and does not represent a commitment on the part of FileTek, Inc. Further, FileTek, Inc. reserves the right to supplement the document with information not available at the time of creation of the document. FILETEK, INC. PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, AND CANNOT WARRANT THE RESULTS YOU MAY OBTAIN USING THE DOCUMENT. IN NO EVENT SHALL FILETEK, INC. BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS, LOSS OF USE OR DATA, INTERRUPTION OF BUSINESS, OR FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND, EVEN IF FILETEK, INC. HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES ARISING FROM ANY DEFECT OR ERROR IN THIS PUBLICATION. Some states or jurisdictions do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

FileTek and StorHouse are registered U.S. trademarks of FileTek, Inc. VRAM is a U.S. trademark of FileTek, Inc. Centera is a trademark of EMC Corporation. All other brand or product names are trademarks or registered trademarks of their respective owners.

Documentation for FileTek's StorHouse product. Protected by the following U.S. Patents: 4,864,572; 5,247,660; 5,727,197; 6,049,804.

Contents

Welcome	v
Intended Audience	v
Document Organization	v
Related Documentation	vi
 Chapter 1: Release Enhancements	 1-1
Detailed Feature List	1-1
Enhancements to Level F Storage	1-3
Centera as Level F	1-3
Device Scan Recovery	1-3
Enhancements to Level L Storage	1-4
File Retention	1-5
About File Retention	1-5
Setting Retention Periods	1-6
Setting File-Level Retention	1-6
Setting File Set-Level Retention	1-7
Determining File Retention Values When DEFAULT is Specified	1-7
Uncataloging Volumes with Retained Files	1-8
File Replication	1-9
About Replication	1-9
Implementing Replication	1-10
System Parameter Changes	1-11
RETENTION_MODE System Parameter	1-11
Changes XFR_COUNT System Parameter	1-12
User Log Changes	1-12

Contents

security_file Record	1-12
security_cmd Record	1-14
command_exec Record	1-14
device_error	1-14
file_copy Record	1-15
New Messages	1-15

Chapter 2: Changes to the StorHouse Interface 2-1

StorHouse Command Language	2-1
CATALOG DEVICE	2-2
CREATE FILE	2-4
CREATE FSET	2-15
MIGRATE	2-21
PUT	2-22
REPLICATE	2-36
SET DEVICE	2-41
SET FILE	2-44
SET FSET	2-53
SHOW FILE	2-60
SHOW FSET	2-76
Generic Callable Interface	2-80
LSMCO - Create Open	2-80
LSMOS - Open Sequential	2-86
Mainframe Callable Interface	2-92
CREATE-OPEN	2-92
OPEN-SEQ	2-98

Chapter 3: Documentation Updates 3-1

MIGRATE Command	3-1
RELOCATE Command	3-1
SHOW ACCOUNT Command	3-2

Welcome

This document describes the content of StorHouse® Release 5.6. It also contains documentation updates to the MIGRATE, RELOCATE, and SHOW ACCOUNT commands in the StorHouse 5.4 *Command Language Reference Manual* since its last publication (March 28, 2002).

Release 5.6 supports all capabilities of previous releases except if noted otherwise in this document.

Intended Audience

These release notes are intended for StorHouse users who are familiar with the StorHouse software and for new users who want a summary of system changes.

Document Organization

This document contains three chapters:

- Chapter 1, “Release Enhancements,” lists the enhancements and features in StorHouse Release 5.6, explains new concepts and user log information, and describes system parameter changes and new StorHouse messages.
- Chapter 2, “Changes to the StorHouse Interface,” describes the StorHouse Release 5.6 modifications to Command Language commands, Generic Callable Interface functions, and mainframe Callable Interface functions.
- Chapter 3, “Documentation Updates,” provides changes to the MIGRATE, RELOCATE, and SHOW ACCOUNT commands in the StorHouse 5.4 *Command Language Reference Manual* since its last publication (March 28, 2002).

Related Documentation

The StorHouse User Document Set contains the following manuals:

- *Introduction to the StorHouse User Document Set*, publication number 900031, describes the general features of the StorHouse User Document Set and provides an overview of each document in the set.
- *Host Installation and Operations Guide*, publication number 900011 for IBM™ MVST™ hosts, 900051 for UNIX® hosts, and 900052 for DOS hosts, explains how to install the StorHouse host software. These guides are intended for system administrators and system programmers.
- *StorHouse Glossary*, publication number 900027, defines technical terminology used in all FileTek® StorHouse publications.
- *StorHouse Concepts and Facilities Manual*, publication number 900026, defines the basic concepts, structures, and functions of StorHouse. It is intended as a StorHouse reference.
- *Callable Interface Programmer's Guide*, publication number 900013 for mainframe hosts and the *Generic Callable Interface Programmer's Guide*, publication number 900046 for all other hosts, are references for programmers who write applications that invoke the StorHouse Callable Interface. These guides explain the functions of the Callable Interface and contain sample programs.
- *Command Language Reference Manual*, publication number 900005, is a general reference for Command Language, the standard command interface between StorHouse and all host computers. It contains descriptions of commands and related concepts.
- *System Operator's Guide*, publication number 900008, contains basic operating instructions for StorHouse hardware and software. It is intended for the system operator.
- *System Operator's Quick Reference*, publication number C00003, presents the procedures for logging in and out of the StorHouse operating system, manually starting the StorHouse software, and signing on and off StorHouse.
- *System Administrator's Guide*, publication number 900007, describes system recovery, account administration, and storage management procedures for StorHouse. It is intended for the system administrator.
- *System Administrator's Quick Reference*, publication number C00006, presents the syntax of StorHouse Command Language commands that system administrators frequently use.

- *Messages and Codes Manual*, publication number 900011, describes the messages and return codes generated by the StorHouse host software and the StorHouse software. It lists the messages by status code, gives the meaning of each message, and indicates any actions to take as a result of the messages.
- *User Log Format*, publication number 900028, describes the format of the StorHouse User Log. It is intended for programmers who write applications to generate reports from statistical information contained in the User Log.

Refer to these documents for more information about your StorHouse system.



Welcome

Related Documentation

Release Enhancements

This chapter lists the enhancements and features in StorHouse Release 5.6; describes the new retention, replication, and user log features; and presents system parameter changes and new StorHouse messages. Refer to Chapter 2, “Changes to the StorHouse Interface,” for a description of the new and/or modified StorHouse Command Language commands, Generic Callable Interface functions, and mainframe Callable Interface functions that support these features.

Detailed Feature List

StorHouse Release 5.6 contains the following additions and modifications:

- Enhancements to level F storage
 - Support for Centera as part of the StorHouse level F storage.
 - Device scan recovery for directory and disaster (extended) recovery of Centera devices.
 - New media specifications (MJA and MJB) for Centera.
- Enhancements to level L storage
 - Support for COPAN Systems MAID technology as part of StorHouse level L storage.
 - New media specifications (PAA) for COPAN Systems Revolution 200T MAID device.
- Support for file retention at the file and file set level.
- Support for application-independent file replication from one StorHouse system to another.

- New StorHouse Command Language commands
 - CATALOG DEVICE for implementing device scan recovery.
 - REPLICATE for duplicating files from one StorHouse system to another.
- Modifications to existing StorHouse Command Language commands
 - SET DEVICE: New /SIZE modifier to support configuring space on Centera clusters and filesystem level F drives (for example, network attached storage [NAS] devices).
 - SHOW FILE:
 - Command modifications consisting of a new /NAME command modifier; new /BKP_ATTR, /BUFFERED, /RESIDENT, and /SAFE_COPIES parameter modifiers; and expanded definition of the /DAMAGED parameter modifier.
 - SHOW FILE/FULL display modifications consisting of a new descriptor field (REPLICATED) and two new fields (RETENTION and RPL_CLASS).
 - SHOW FSET: New display that includes one new STATE (FORCE_RETENTION) and two new fields (RETENTION and RPL_CLASS).
 - SET FILE: New /FSET, /RETENTION, /NOREPLICATED, /RPL_CLASS, and /WAIT modifiers.
 - CREATE FILE: New /RETENTION modifier.
 - PUT: New /RETENTION modifier.
 - SET FSET: New /FORCE_RETENTION, /RPL_CLASS, and /RETENTION modifiers.
 - CREATE FSET: New /FORCE_RETENTION, /RETENTION and /RPL_CLASS modifiers.
 - MIGRATE /BY_VSET: Always creates noncontiguous destination file sets.
- New and modified system parameters
 - New RETENTION_MODE system parameter for setting a system-wide retention enforcement level.
 - Increased maximum for the XFR_COUNT system parameter.

- Changes to the Generic Callable Interface LMSOS and LSMCO functions to support file retention.
- Changes to the mainframe Callable Interface OPEN-SEQ and CREATE-OPEN functions to support file retention.
- Changes to five StorHouse user log records: security_file, security_cmd, command_exec, device_error, and file_copy.
- New StorHouse messages.

StorHouse Release 5.6 requires Sun™ Solaris™ 8 or higher.

Enhancements to Level F Storage

StorHouse Release 5.6 adds Centera (fixed content object-based disk) to the StorHouse level F storage layer. This release also supports device scan recovery for directory and disaster recovery of Centera devices.

Centera as Level F

A Centera cluster is defined as a logical level F volume (or device). Centera clusters may be shared by non-StorHouse applications and by other StorHouse systems.

When a Centera device is part of the StorHouse storage hierarchy, it must be configured as the only type of level F device. Subsequent release upgrades will support the coexistence of different kinds of level F devices.

StorHouse uses media type MJ to identify Centera devices. A recording type of A indicates a Centera device operating in non-compliance mode. A recording type of B indicates a Centera device operating in Basic Compliance or Compliance Plus mode. Media type MJ identifies the vendor application program interface (API) used to access the storage device. Vendor software/firmware controls all media characteristics.

Device Scan Recovery

StorHouse Release 5.4 does not support directory or disaster recovery of level F devices. Device scan recovery is a new recovery strategy that provides these features for Centera.

Device scan recovery scans the level F device to be recovered to obtain information about the StorHouse files, extents, and file sets on the device. Then it catalogs those files, extents, and file sets. System administrators implement device scan recovery with the new StorHouse Command Language CATALOG DEVICE command.

To prepare for disaster recovery of a level F device, the device at the primary site must replicate data to a device at the backup site. This requires a network connection of sufficient bandwidth. Because level F devices that support replication write files to the backup site almost continuously, device scan recovery restores files almost to the point of the disaster. (Device replication differs from StorHouse replication. Refer to page 1-9 for information about StorHouse file replication.)

Device scan recovery is independent of existing disaster recovery procedures for level L optical and tape volumes. StorHouse systems with level F and level L devices must perform disaster recovery for both. Furthermore, if checkpoint recovery is used to recover level L, it must be completed before level F device scan recovery begins.

As with checkpoint recovery, directory recovery from extraction files, and directory recovery from physical volumes, FileTek must assist customers with device scan recovery.

Enhancements to Level L Storage

StorHouse Release 5.6 adds COPAN Systems Revolution 200T massive arrays of idle disk (MAID) to the StorHouse level L storage layer. MAID is a new “power-managed” storage technology where disks spin and consume power only when required for data access. StorHouse enhances MAID reliability, performance, application support, scalability, and operation.

The media and recording type for COPAN Systems MAID is PAA, where the general media type P indicates power-managed disk arrays, the form factor A indicates a MAID device that emulates tape, and the recording type A specifically indicates the Revolution 200T.

PA has the following characteristics:

- MAID device emulating a tape library and tape drives.
- High-performance.
- Zero time extent positioning.
- Erasable (volume reusable).
- Drives with optional support for compression.

- Single-sided virtual tape volumes (VTC), which are permanent and cannot be removed.
- VTCs with a shelf life that is a function of the time-powered-on and the number of power cycles.
- Volumes that can be read many times without degrading the media.
- Support for one read or one write at a time.
- Recording type:
 - A indicates a COPAN Revolution 200T shelf with 27 VTCs (each with 750 GB) and a maximum of seven virtual tape drives (VTDs), which can be powered on concurrently.

File Retention

StorHouse Release 5.6 provides new features for setting and enforcing file retention. These features ensure StorHouse archiving adequately supports industry-based compliance rules and data remains accessible throughout its required life span.

About File Retention

Starting with SM 5.6, each file written to StorHouse has a *retention attribute*, which determines a file's retention period. A *retention period* indicates the time span that a file may not be deleted from StorHouse. A file is considered *retained* when it has a non-zero retention period and that retention period has not expired.

StorHouse calculates file retention periods by adding the number of days in the retention period to a file's last modified date (the `modified_file` date as displayed by the StorHouse Command Language `SHOW FILE` command). A file retention period ends when the current date is beyond the last modified date plus the retention period. For example, if a file with a 3-day retention period was last modified at 11 p.m. on December 12, it would expire at 11 p.m. on December 15.

StorHouse does not automatically delete files after their retention period expires. It simply allows applications or users to delete them.

A user or an application can specify one of four retention values at file create time:

- `DEFAULT` (the retention is not specified and assumes the default value).

- FOREVER (never allow the file to be deleted).
- ZERO (do not apply any retention period to the file).
- Number of days (retain the file for the specified number of days).

The maximum file retention period is 65,000 days, or approximately 178 years.

Copies of primary files (for example, backup, archive, replica, and relocated files) inherit the retention attribute of the primary. The exception is the performance buffer copy. It is treated as temporary cache and has no retention attribute.

StorHouse records a file's retention setting on the media where the file resides. If the retention period is modified, the system updates the media to preserve the change.

Note the following:

- To specify retention on a StorHouse 5.6 system, you can use any version of the StorHouse Host Application Programming Interface (API) with the Interactive Interface. You must use version 2.5 (for Windows/UNIX) or version 1.8 (for MVS) of the StorHouse Host API with the Callable Interface. If you use prior versions of the Host API, StorHouse assumes that the retention period is the default.
- The new Host API releases do not pass retention information from Callable Interface functions to a StorHouse system running any release prior to StorHouse 5.6. Therefore, the Callable Interface can be used with older StorHouse systems. However, interactive commands with retention parameters/modifiers will fail if executed on a StorHouse system running a release prior to 5.6.
- When an older StorHouse release is upgraded to Release 5.6, files and file sets created with previous StorHouse releases will be assigned a retention attribute of ZERO after the conversion.

Setting Retention Periods

Retention can be set at the file and file set level. File-level retention takes precedence over file set-level retention unless the system administrator or a user with the proper authority instructs the software otherwise. (See the `/FORCE_RETENTION` modifier described in the “File Set-Level Retention” section on page 1-7.)

Setting File-Level Retention

You can set file-level retention when writing a file to StorHouse by specifying:

- The /RETENTION modifier on the StorHouse Command Language CREATE FILE or PUT command.
- The retention_interval list member on the Generic Callable Interface LSMCO or LSMOS (MODE=WRITE) function.
- The FATTR-RETENTION-INTERVAL on the mainframe Callable Interface OPEN-SEQ or CREATE-OPEN function.

You can change the file-level retention attribute for a particular file version by specifying a new retention value on the SET FILE command.

Setting File Set-Level Retention

System administrators or users with the proper authority can configure file set-level retention by specifying the /RETENTION modifier on the CREATE FSET or SET FSET command. CREATE FSET /RETENTION applies to all files in the file set. SET FSET/RETENTION applies only to new files in the file set.

CREATE FSET and SET FSET also support a /FORCE_RETENTION modifier. When a file is created, the /FORCE_RETENTION modifier tells StorHouse to override the file retention attribute with the file set retention attribute. Administrators can use /FORCE_RETENTION to supersede the retention period specified by an application through the Callable or Interactive Interface.

For example, assume the following:

- The system administrator specified /FORCE RETENTION and a file set-level retention of FOREVER on the CREATE FSET command for FSET_A.
- An application creates File A in FSET_A and explicitly specifies a 2-day retention period.

In this case, at file create time, the file set-level retention attribute overrides the file-level retention attribute of two days, causing StorHouse to retain File A forever.

Determining File Retention Values When DEFAULT is Specified

When StorHouse creates a primary file, it checks several places to determine the file's retention value. As soon as StorHouse finds a retention value other than DEFAULT, it assigns that value to the file.

- First, StorHouse checks the Interactive Interface command or Callable Interface function that creates the file. If it finds a value other than DEFAULT, StorHouse assigns that value to the file unless /FORCE_RETENTION is in effect on the file set. In this case, StorHouse overrides the file-level specification with the file set-level specification.

- If the file retention value is DEFAULT, StorHouse checks the file set retention attribute for a value other than DEFAULT. If found, it assigns that retention value to the file.
- If the file set retention value is DEFAULT, StorHouse uses the value of the RETENTION_MODE system parameter to determine file retention. If RETENTION_MODE is set to BASIC, StorHouse sets the file-level retention to ZERO, which indicates no retention. If set to STRICT, StorHouse sets the file - level retention to FOREVER, which indicates infinite retention. (Refer to the section, “RETENTION_MODE System Parameter,” on page 1-11 for more information.)

Table 1-1 provides some examples of how StorHouse determines file-level retention.

Table 1-1: File Retention Examples

Category	Value			
	Example 1	Example 2	Example 3	Example 4
File-level retention	DEFAULT	DEFAULT	30 days	30 days
File set-level retention	DEFAULT	DEFAULT	60 days	60 days
/FORCE_RETENTION	N/A	N/A	No	Yes
RETENTION_MODE	BASIC	STRICT	N/A	N/A
Final File Retention	ZERO	FOREVER	30 days	60 days

Uncataloging Volumes with Retained Files

The new retention feature affects the operation of the UNCATALOG command. Therefore, customers who currently use UNCATALOG and EXPORT to remove archive volumes from StorHouse may want to rethink this strategy.

StorHouse cannot uncatalog a volume that contains retained files because those files cannot be deleted until their retention period expires. When StorHouse attempts to uncatalog such a volume, it ignores the retained files, deletes and removes all non-retained files on the volume, and leaves the volume in the UNCATALOGING state. A volume in the UNCATALOGING state cannot be exported. To circumvent this restriction, FileTek recommends that instead of using UNCATALOG and EXPORT to remove archive volumes from StorHouse, you simply write lock the volumes to be exported and use the MOVE VOLUME command to transfer them from level L to shelf storage.

File Replication

StorHouse Release 5.6 supports application-independent replication of one or more eligible primary files from a source StorHouse system to a destination, or target, StorHouse system. Replication is typically used to provide redundancy for disaster recovery or off-site storage. A file is considered *replicated* when at least one copy resides on the destination system, including on the performance buffer. Any StorHouse file type can be replicated (VRAM, sequential, or STORHOUSE).

About Replication

A primary file is eligible for replication when it has an assigned replication class. A *replication class* is a named set of information about the destination StorHouse system. Table 1-2 describes the components that compose a replication class.

Table 1-2: Replication Class Components

Component	Specifies the
RPL_CLASS_NAME	Name of the replication class. A replication class name can consist of from 1-8 of the following ASCII characters: A_Z, 0-9, _ and \$. StorHouse always forces replication class names to uppercase, even when enclosed in quotes.
DISABLED_FLAG	Indicator specifying whether the replication class is disabled. A non-blank character indicates disabled.
NETWORK_DEVICE	Network device used to connect to the destination StorHouse system. This value must be set to N00 to indicate TCP/IP.
SYSNAME	Network system name used to connect to the destination StorHouse system. This value must be the same as the SM_HOSTID in the default SMCONFIG file on the destination system.
LINKNAME	Network link name used to connect to the destination StorHouse system. This value must be the same as the SM_LINKNAME in the default SMCONFIG file on the destination system.

Table 1-2: Replication Class Components (continued)

Component	Specifies the
VSET	<p>Volume set that will contain the replicated files on the destination StorHouse. Valid values are:</p> <ul style="list-style-type: none"> • VSET=volume_set_name • VSET=* (indicates to use the same volume set name as the source file) <p>If VSET is omitted, StorHouse uses the account's default volume set on the destination StorHouse.</p>
FSET	<p>File set name that will contain the replicated files on the destination StorHouse system. Valid values are:</p> <ul style="list-style-type: none"> • FSET=file_set_name • FSET=* (indicates to use the same file set name as the source file) <p>If FSET is omitted, StorHouse uses the account's default file set on the destination StorHouse system.</p>

StorHouse associates a file with a replication class by:

- Assigning a replication class to the file's resident file set (SET FSET or CREATE FSET command). In this case, at create time, the file inherits its replication class from its file set.
- Explicitly specifying a default replication class for the file version (SET FILE command).

Currently, FileTek is responsible for creating and maintaining replication classes. For more information about these tasks, contact your FileTek customer support representative.

Implementing Replication

System administrators implement replication with the StorHouse Command Language REPLICATE command. For convenience, this command can be scheduled to run periodically.

A replication operation works as follows:

- When an application creates or modifies a replication-eligible file, StorHouse queues the same action to occur on the destination StorHouse system the next time the REPLICATE command executes on the source system and selects that file.

- When a file is deleted (and removed) from the source system, StorHouse queues the same action to occur on the destination StorHouse the next time the REPLICATE command executes on the destination system and selects any file.

StorHouse does not copy file and file set replication class attributes from one system to another. A replica inherits the replication class attribute of its target file set. In addition, the system does not write queued files to the destination system in any particular order or within a set time interval.

REPLICATE uses the same account, password, and file access group on the source and target systems. The system administrator must manually create the respective accounts and volume sets on the target location. If the required groups and file sets do not already exist on the destination system, StorHouse creates them automatically. System-created file sets are noncontiguous and use the other CREATE FSET command defaults.

Refer to the REPLICATE command description on page 2-36 for more information.

System Parameter Changes

StorHouse Release 5.6 contains one new system parameter and changes to the XFR_COUNT parameter.

RETENTION_MODE System Parameter

The RETENTION_MODE system parameter supports two system-wide *retention enforcement levels*: BASIC and STRICT. These levels determine how strictly StorHouse manages file retention. FileTek Customer Support sets the value of RETENTION_MODE for your StorHouse system at installation and can subsequently reset the parameter at your request.

The RETENTION_MODE system parameter is defined as follows.

RETENTION_ MODE

Specifies the retention enforcement level for all files.

- Expanded Name: Retention mode
- Type: Static parameter
- Range: BASIC, STRICT
- Default: BASIC
- User Access: SHOW

Table 1-3: RETENTION_MODE Value Descriptions

Value	Description
BASIC	<p>Prevents deletion of a retained file, irrespective of user privilege, before the file's retention period expires.</p> <p>Sets the default retention attribute to ZERO when the retention period is not specified at the file and file set level.</p> <p>Enables the retention setting specified on SET FSET to be more or less restrictive than the current setting. For example, a retention setting of 30 days can be changed to 60 days or 20 days.</p>
STRICT	<p>Prevents deletion of a retained file, irrespective of user privilege, before the file's retention period expires.</p> <p>Sets the default retention attribute to FOREVER when the retention period is not specified at the file and file set level.</p> <p>Forces StorHouse to write retained files to WORM or compliant media.</p> <p>Forces the retention setting specified on SET FSET to be more rather than less restrictive than the current setting. For example, a retention period of 30 days can be changed to 60 days but not to 20 days.</p>

The default RETENTION_MODE value at StorHouse installation is BASIC.

Changes XFR_COUNT System Parameter

The maximum value of the XFR_COUNT system parameter has been increased from 64 to 128.

User Log Changes

StorHouse Release 5.6 supports changes to four user log records:

- security_file
- security_cmd
- command_exec
- file_copy.

security_file Record

StorHouse logs a security_file record (record type 7) when:

- Access to a file is denied due to either a missing or incorrect file password or an appropriate privilege.
- There is an attempt to shorten the retention period of a retained file.

Two new fields, status and status_message, have been added to this record. Table 1-4 shows the new record layout.

Table 1-4: security_file record layout

Field	Type	Max. Len.	Value
record_code	num	2	7
system_id	char	6	System identifier
account_id	char	12	Account id of user who submitted the access request.
user_id	num	10	User id of user's session.
log_time	time	20	Time record was logged.
start_time	time	20	Time request was received.
end_time	time	20	Time request processing was completed. (This value is always the same as start_time.)
command	char	255	Description of the activity that caused the violation. This may be a StorHouse command, Callable Interface function, or another action (for example, an internal StorHouse process).
file_sysid	num	5	System identifier specified in command on which the violation attempt was made. If no file identifier was specified or the file identifier had not been located before the violation attempt was recognized, this field contains a 0 (zero).
file_fno	num	10	File number specified in the command on which the violation attempt was made. If no file identifier was specified, or the file identifier had not been located before the violation attempt was recognized, this field contains a 0 (zero).
filename	char	56	Name of file for which required password was not specified.
group	char	8	Name of group in which the specified file is located.

Table 1-4: security_file record layout (continued)

Field	Type	Max. Len.	Value
status	num	5	StorHouse status code that identifies error conditions.
status_message	char	132	Text message corresponding to the status code in the status field.

security_cmd Record

In addition to its other functions, the security_cmd record (record type 8) reports retention violations (for example, an attempt to delete a retained file).

command_exec Record

The following values have been added to the list of command identifiers in the StorHouse user log command_exec record (record type 9):

- 1711 for the CATALOG DEVICE command
- 1411 for the REPLICATE command
- 1412 to indicate [REPLICATION_SERVER]. This identifier specifies that a file copy action occurred on a secondary StorHouse system due to a REPLICATE command on a primary StorHouse system.

device_error

The following identifiers have been added to the list of values for the dtype field in the user log device_error (record type 14) record.

- FILESYST
- CENTERA.

file_copy Record

The following identifiers have been added to the list of values for the command field in the user log file_copy (record type 22) record.

- 7 for the REPLICATE command
- 8 to indicate [REPLICATION_SERVER]. This identifier specifies that a file copy action occurred on a secondary StorHouse system due to a REPLICATE command on the primary StorHouse system.

This record applies to the ARCHIVE, CREATE BACKUP, CREATE PRIMARY, RELOCATE, RECOVER VOLUME, RETIRE VOLUME, and REPLICATE commands.

New Messages

5673 XWRETEN Request rejected, operation prohibited by retention rules.

Explanation: File retention rules prohibit the requested operation. Examples include attempting to delete a file whose retention period has not expired, and attempting to shorten a file's retention period.

4959 XTRPLCLNF Replication class not found.

Explanation: The replication class does not exist. Verify that the correct replication class name was specified.

4960 XTRPLSYS Replication system name invalid.

Explanation: The replication system name does not exist. Verify that the correct system name was specified.

4961 XTRPLNCONN Replication connect failed.

Explanation: Replication failed to connect to the destination StorHouse. The condition may be permanent (e.g., a configuration error) or temporary (e.g., destination StorHouse is down).

4962 XTRPLLOGIN Replication login failed.

Explanation: Replication failed to log in to the destination StorHouse. Verify that the account and password are defined on the destination StorHouse.

4963 XTRPLNDISC Replication network disconnected.

Explanation: Replication was disconnected from the destination StorHouse. This is usually caused by an error with the network or the destination StorHouse.

4964 XTRPLCNFLCT Replication file or extent conflict.

Explanation: Replication detected a conflict between a source file or its extents and an existing file on the destination StorHouse. Files on the source and destination StorHouse systems can get out of sync if they are updated independently.

Changes to the StorHouse Interface

This chapter describes the StorHouse Release 5.6 changes to Command Language commands, Generic Callable Interface functions, and mainframe Callable Interface functions.

StorHouse Command Language

StorHouse Release 5.6 supports the following changes to StorHouse Command Language:

- New CATALOG DEVICE and REPLICATE commands
- Updated CREATE FILE, CREATE FSET, CREATE VSET, PUT, SET DEVICE, SET FILE, SET FSET, SHOW FILE, and SHOW FSET commands.

Except where noted, this chapter presents complete descriptions of enhanced and new StorHouse commands.

CATALOG DEVICE

CATALOG DEVICE initiates device scan recovery for a level F device.

Format

CATALOG DEVICE did

COMMAND FORMAT SUMMARY					
COMMAND, PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
CATALOG DEVICE	ALLOCATION + SYSTEM	-	-	-	-
/CONFIRM	-	-	-	-	/CONFIRM
/REPORT	-	-	-	-	-
/WAIT	-	-	-	-	-
did	-	-	-	-	(Required)

Description

CATALOG DEVICE initiates device scan recovery for a level F device. It scans the level F device to be recovered to obtain information about the StorHouse files, extents, and file sets on the device. Then it catalogs those files, extents, and file sets.

One CATALOG DEVICE command does device scan recovery on one device.

Note the following:

- The device to be cataloged must be online and uncataloged.
- During the device scan, if the command encounters any files that are already in the StorHouse catalog, it skips them.
- Multiple CATALOG DEVICE commands can be executed concurrently as long as each specifies a different device.
- CATALOG DEVICE only supports EMC Centera devices.

Parameters

did Specifies the device identification code (did) of the level F device to be scanned. This must be a Centera cluster.

- **FORMAT:** F{unit_number}
- **DEFAULT:** None; you must specify this parameter.

Command Modifiers

/CONFIRM Controls whether StorHouse asks you to confirm the command.

- **FORMAT:** /CONFIRM or /NOCONFIRM
- **DEFAULT:** /CONFIRM

When the system requests a confirmation, enter YES (also Y or YE) or NO (also N). If you press **Enter** or enter any characters other than those described as a YES response, StorHouse interprets them as NO.

/REPORT Controls the generation of special text responses for the completion of significant actions. /REPORT instructs StorHouse to generate a text response. /NOREPORT instructs StorHouse not to generate a text response. The report generated for this command only includes summary information (for example, the number of files added, modified, and skipped). It does not contain the names of all files that are processed.

- **FORMAT:** /REPORT or /NOREPORT
- **DEFAULT:** /NOREPORT

/WAIT Instructs StorHouse to wait for a locked file to be unlocked before attempting to use it in the command execution. Without /WAIT, StorHouse returns an error status if it encounters a locked file.

- **FORMAT:** /WAIT
- **DEFAULT:** No /WAIT modifier. StorHouse aborts the command or processing of an item if the specified file version is not available.

Examples

To initiate device scan recovery for a level F device with a device identification code of F01 and generate text responses, enter:

? CATALOG DEVICE F01 /REPORT

CREATE FILE

The CREATE FILE command creates a new VRAM file or file version.

Format

CREATE FILE filename

COMMAND FORMAT SUMMARY					
COMMAND, PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
CREATE FILE	RECORD	-	-	-	-
/REPORT	-	-	-	-	-
filename	-	W	-	-	(Required)
/ASCII	-	-	-	-	Binary format
/ATF=...	ATF	D	-	-	See text
/CACHE=...	-	-	-	-	/CACHE=0
/DIRECT	-	-	-	-	-
/EDC	-	-	-	-	See text
/EXTERNAL	-	-	-	-	No external keys
/FSET=...	-	-	-	-	Current default
/GROUP=...	SETGROUP	W	-	-	Current default
/LIMIT=...	DELETE	D	-	-	See text
/NEWPASSWORDS=...	PASSWORD	D	D	-	See text
/PASSWORDS=...	-	-	-	-	-
/REPLACE=...	DELETE	D	D	-	-
/RETENTION=...	-	-	-	-	DEFAULT
/SIZE=...	-	-	-	-	(Required)
/TYPE=...	-	-	-	-	/TYPE=RECORD
/VSET=...	-	-	-	-	Current default
/VTF=...	VTF	D	-	-	See text

Description

CREATE FILE creates a new StorHouse file or a new file version.

The /SIZE modifier is required. It specifies the maximum number of bytes needed to store the largest extent set created in an append operation. An *extent set* consists of a data extent, a DF extent, and for KEYED files, a K extent. If a file is not checkpointed, it has one extent set created from OPEN to CLOSE. If a file is checkpointed, a new extent set is created each time a checkpoint is issued.

If the file is not checkpointed, the amount of space specified by /SIZE is allocated to store the extent set that is written from OPEN to CLOSE. If the file is checkpointed, the amount of space specified by /SIZE is allocated when the file is opened and *each time* a CHECKPOINT is taken.

When you write records into the file, the system places the records in the performance buffer and allows the backup function to copy the records to their primary file set (the file set specified by /VSET and /FSET) unless the file is created in a level F volume set and file set. If it is created in a level F volume set, the performance buffer is not used. If you specify /DIRECT, the system transfers the records directly to the primary file set, bypassing the performance buffer. The system writes file updates to the file set's update space, if available.

Parameters

filename Specifies the StorHouse name of the VRAM file version to be created.

- **FORMAT:** filename

File names must be unique within an access group. The StorHouse file name must contain 1 to 56 printable ASCII characters. At least one character must be non-blank. StorHouse translates lowercase letters to uppercase letters and compresses multiple consecutive spaces to a single space unless they are enclosed in quotes. File names containing special characters must be enclosed in quotes, unless the characters are any of the following:

[] : \$. ; _

You can use the quote symbol (") in the file name as long as you enclose the name in quotes and you place two quotes (") wherever a single quote (") is to appear.

- **DEFAULT:** None; you must specify this parameter.
- **ACCESS REQUIREMENTS:** Write access to the file's group.

Command Modifier

/REPORT Controls the generation of a special text response for the completion of the command. The text includes the file identifier (fid) of the new file or file version created.

/REPORT instructs StorHouse to generate a text response. /NOREPORT instructs StorHouse not to generate a text response.

- **FORMAT:** /REPORT or /NOREPORT
- **DEFAULT:** /NOREPORT

Parameter Modifiers

/ASCII Causes the host interface to translate a host file's data from the host's native character set into ASCII characters. The host interface formats the translated data into a transportable ASCII character-stream file format while transferring the file to StorHouse.

- **FORMAT:** /ASCII
- **DEFAULT:** If you omit /ASCII, StorHouse formats the file data into transportable binary bit-stream format.
- **RESTRICTIONS:** Do not specify /ASCII with a different format indicator, such as /BINARY, or with files that cannot be translated into ASCII characters.
- **HOST DEPENDENCIES:** The native UNIX character set is ASCII, so no character translation is necessary.

The mainframe Callable Interface translates data between EBCDIC characters and ASCII characters.

/ATF Specifies a value for the ATF (Access Time Factor) attribute for a file version. The ATF attribute indicates the importance of access time for the file. Setting an ATF value does not initiate a file transfer directly, but it may cause the file to be migrated in a subsequent migration.

- **FORMAT:** /ATF={1,2,3}

A value of 1 indicates that a short access time for the file is very important; 2 indicates that access time is moderately important; 3 indicates that it is minimally important. The MIGRATE function migrates files off of the performance buffer, beginning with files with the largest ATF values.

- **DEFAULT:** If you omit /ATF when creating a new file version, the default is the current value of the ATF system parameter.

/CACHE Specifies the number of sequential records that VRAM will cache for a READ-SEQUENTIAL, READ-RECORD, or READ-KEYED function for this file when it is opened with an access mode of READ or UPDATE and an access method of RECORD and/or KEYED. The system can use this cache to optimize subsequent reads. VRAM caches n-1 records preceding the current record through n records following the current record; in other words, VRAM caches twice the specified number of records including the current record. (This assumes that there is enough cache memory available to accommodate the total number of records.)

- **FORMAT:** /CACHE=number_of_records

The number of records can range from 0 to 65,535; however, StorHouse limits the number of records it caches to a number less than or equal to the number of bytes specified by the VRAM_CACHE_MAX system parameter.

Note that VRAM caches records preceding the current record up to and including the current record only if the records already reside in memory and begin in the currently loaded frame.

- **DEFAULT:** If you omit /CACHE, the default value is /CACHE=0 (no cache).

/DIRECT Indicates that when you write data records to the file, the system is to write the records directly to the primary file set (specified by /VSET and /FSET) specified in this command.

- **FORMAT:** /DIRECT
- **DEFAULT:** If you omit /DIRECT, the system writes the records to the performance buffer and allows the backup function to copy the data to the specified primary file set.
- **PRIVILEGE:** None.
- **RESTRICTIONS:** This modifier has the same function as /VTF=DIRECT. If you specify /DIRECT, the system ignores /VTF.

/EDC Controls the generation of error detection codes by the host interface during data transfer to StorHouse.

- **FORMAT:** /EDC or /NOEDC

If you specify /EDC when executing CREATE FILE, the host interface will generate or check error detection codes when the file is transferred to or from StorHouse. If you specify /NOEDC, the host interface will not generate or check error detection codes.

- **DEFAULT:** If you omit /EDC, the default is given by the EDC system parameter. If the value of the parameter is TRUE, the default is /EDC. If the value of the parameter is FALSE, the default is /NOEDC.

/EXTERNAL Indicates that you will define external keys—key data provided in special records that are not a part of the file's data records.

Note: External key values cannot be changed if file records are updated. Also, records with duplicate external key values cannot be distinguished unless the record data contains information that you can use for this purpose.

- **FORMAT:** /EXTERNAL
- **DEFAULT:** If you do not specify /EXTERNAL for a KEYED file, the system assumes that the keys will be internal to the user data records.
- **RESTRICTIONS:** /EXTERNAL is valid only if you also specify /TYPE=KEYED.

/FSET Specifies the primary file set for the file. The performance buffer file set is not allowed. The specified file set must exist.

If /FSET and /VSET specify or default to a level F file set and volume set, StorHouse does not use the performance buffer when you write data into the file. Data is written directly to the primary file set and volume set on level F, regardless of the value of the /VTF or /DIRECT modifier, if specified.

- **FORMAT:** /FSET=fset_name
- **RESTRICTIONS:**
 - You cannot use a wild card.
 - Do not specify the performance buffer file set name.

/GROUP Specifies a file access group name and, optionally, group passwords. The specified group must exist.

- **FORMAT:**
 - /GROUP=groupname<::writepw>
 - /GROUP=groupname
- **FORMAT RESTRICTIONS:** Wild cards are not allowed in the group name.
- **ACCESS REQUIREMENTS:** You must have write access to the group. Also, you must specify the group's write password unless:
 - The group is not protected by a write password.
 - Your privilege bypasses write access password checks.
 - Your default access to the group includes write access.

If you enter parameter modifiers that require delete access to the group, you must also have delete access to the group.

- **DEFAULT:**
 - If you omit /GROUP, the default is your current default group and default access rights.
 - If you specify the current default group name and omit the write password, the defaults for your group access rights apply.
 - If you specify a group name that is not the current default group and omit the write password, the write password defaults to null.
- **PRIVILEGE:** SETGROUP is required to specify any group except your default group.

/LIMIT Specifies a value for the LIMIT attribute for a file.

- **FORMAT:** /LIMIT=maximum_versions

The value of maximum_versions can range from 1 through 32768.

- **DEFAULT:** If you omit /LIMIT and this is a new file, the default limit is specified by the LIMIT system parameter. If this is a new version of an existing file, the default is the limit for the existing file.
- **PRIVILEGE:** DELETE privilege.

/NEWPASSWORDS Assigns file access passwords to the file.

Note: You cannot obtain access to the file by specifying /NEWPASSWORDS (see the next modifier /PASSWORDS).

- **FORMAT:**
 - /NEWPASSWORDS=<readpw>:<writepw>:<deletepw>
 - /NEWPASSWORDS=<readpw>:<writepw>
 - /NEWPASSWORDS=readpw
 - /NONEWPASSWORDS

Specifying /NONEWPASSWORDS is equivalent to specifying /NEWPASSWORDS with null read, write, and delete passwords.

Passwords must be null or contain 1 to 8 characters, consisting of the following ASCII characters: A-Z (uppercase), 0-9, _ (underscore), and \$ (dollar sign). StorHouse always translates passwords to uppercase characters, even if they are enclosed in quotes.

- **DEFAULT:** If you do not specify /NEWPASSWORDS and the file already exists, the system retains the existing passwords, if any. If you do not specify /NEWPASSWORDS and the file does not exist, the system assigns null passwords to it.

If you specify /NEWPASSWORDS but do not specify one or more passwords, the system assigns a null password for each unspecified password.

- **ACCESS REQUIREMENTS:** You must have delete access to the file and group and PASSWORD privileges.

/PASSWORDS Specifies passwords to gain access to an existing file protected by passwords. Specify a delete password to change file attributes. Specify a write password to create a new version of an existing file. If your privilege bypasses the access password checks, you do not have to specify a password.

If the file does not already exist, StorHouse ignores /PASSWORDS.

/PASSWORDS must be used in conjunction with /REPLACE.

- **FORMAT:**
 - /PASSWORDS=<readpw>:<writepw>:<deletepw>
 - /PASSWORDS=<readpw>:writepw
 - /PASSWORDS=readpw

A file password can be null or contain 1 to 8 characters, and can consist of the following characters: A-Z (uppercase), 0-9, _ (underscore), and \$ (dollar sign). StorHouse always translates passwords to uppercase characters, even if they are enclosed in quotes.

- **DEFAULT:** If you omit /PASSWORDS, the passwords default to nulls.

/REPLACE Indicates that after creating a new version of the file, the system will delete all older versions. If no file of the same name exists, StorHouse ignores this modifier. If a file of the same name exists, StorHouse verifies that you have the required access to the file and that the file is not retained before deleting it. If the existing file is retained, the CREATE FILE/REPLACE operation fails.

After StorHouse deletes the file, it does not retain the old passwords and file attributes. You must specify new passwords or attributes on the CREATE FILE statement.

- **FORMAT:** /REPLACE
- **DEFAULT:** If you omit /REPLACE and a file of the same name already exists in the directory, the system does not delete the existing version.
- **PRIVILEGE:** DELETE privilege
- **ACCESS REQUIREMENTS:** If a file with the same group and file names already exists, you must obtain delete access to the group and file.

/RETENTION Specifies the retention attribute (retention period) for the file being created.

- **FORMAT:**

Option	Description
/RETENTION=DEFAULT	Sets the retention period to the default value.
/RETENTION=number_of_days	<p>Sets the retention period to the specified number of days. The retention period ends when the current date is beyond the file's last modification date plus the specified retention value.</p> <p>A value of 0 indicates no retention period (same as specifying ZERO).</p> <p>Example: /RETENTION=3</p> <p>In this example, the retention period is 3 days. The retention period for a file that was last modified at 11 p.m. on December 12 expires at 11 p.m. on December 15.</p>
/RETENTION=ZERO	Sets no retention period, which indicates the file may be deleted.
/RETENTION=FOREVER	Sets an infinite retention period, which indicates the file may never be deleted.

- **DEFAULT:** If you omit /RETENTION or specify /RETENTION=DEFAULT, StorHouse determines the file's default retention attribute as follows:
 - If the file's resident file set has a retention attribute equal to FOREVER, ZERO, or a specified number of days, the file set retention attribute determines the default file retention attribute.
 - If the file's resident file set has a retention attribute of DEFAULT, the RETENTION_MODE system parameter determines the default file retention attribute. If RETENTION_MODE is set to BASIC, the default file retention is ZERO. If RETENTION_MODE is set to STRICT, the default file retention is FOREVER.
- **RESTRICTIONS:** None.

/SIZE Specifies the number of bytes of storage space to allocate for the file whenever the file is opened for an append operation and whenever a checkpoint is issued. The value must contain enough space for the largest extent set that is written. This extent set includes a data extent, a DF extent, and for KEYED files, a K extent.

- If the file is not checkpointed, /SIZE specifies the space required to store the extent set that is written from OPEN to CLOSE.

- If the file is checkpointed, the space specified by /SIZE is allocated when the file is opened and *each time* a CHECKPOINT is taken.

Note: The largest value that you can specify for /SIZE is 2G. Do not assign a value for /SIZE greater than the size of the volume (or performance buffer) that will contain the extent set. Otherwise, CREATE FILE returns an error.

If the system cannot allocate enough space when the file is opened, it rejects the open. If the system cannot allocate enough space after a checkpoint, it rejects the next write. After a file has been closed or checkpointed, StorHouse returns any unused file storage space to the file set as free space.

For updates, StorHouse does not use the /SIZE value. It automatically allocates space for each extent separately. It allocates as much space as required, up to the file set LIMIT.

Caution: After you have used CREATE FILE to create the file, you cannot change the value of /SIZE.

- FORMAT: /SIZE=number_of_bytes

The number_of_bytes value can be specified as n, nK, nM, or nG.

The letter n represents a numeric field. The value of n can range from 0 up to 2000000000. (Do not include commas when specifying a number with more than three digits.) K indicates that the number is in 1,000-byte units; M indicates 1,000,000-byte units; and G indicates 1,000,000,000-byte units. If K, M, or G is not present, the number defaults to 1-byte units.

The maximum value of /SIZE is limited by the capacity of the volume to which the file is to be written. If you do not use /DIRECT or /VTF=DIRECT, the value must be smaller than the capacity of the largest partition of the performance buffer.

- DEFAULT: None; you must specify this modifier.

/TYPE Specifies the type of file organization desired. A /TYPE value of RECORD indicates that the file can only be accessed sequentially or by record number. KEYED or KEYSEQUENTIAL indicates that the file can be accessed sequentially, by record number, or by key.

KEYSEQUENTIAL files are like KEYED files, but with the following restrictions:

- You can define only one key.
- Duplicate key values are not allowed.
- When writing records into the file, you must write the records in ascending key value order.

- When updating a record, you cannot change the key value.

If /TYPE is KEYED or KEYSEQUENTIAL, StorHouse requests you to enter key definitions. For further information about key definitions, refer to the *Command Language Reference Manual*.

- FORMAT:
 - /TYPE=KEYSEQUENTIAL
 - /TYPE=KEYED
 - /TYPE=RECORD
- DEFAULT: /TYPE=RECORD
- RESTRICTIONS: The VRAM_KEYED system parameter must be set to TRUE for VRAM_KEYED files to be created.

/VSET Specifies the file's primary volume set.

If /VSET and /FSET specify or default to a level F volume set and file set, StorHouse does not use the performance buffer when you write data into the file. Data is written directly to the primary file set and volume set on level F, regardless of the value of the /VTF or /DIRECT modifier, if specified.

- FORMAT: /VSET=vset_name
- DEFAULT: If you omit /VSET, the default is your current default volume set.
- RESTRICTIONS:
 - Wild cards are not allowed.
 - The volume set must be a primary volume set.

/VTF Specifies a file version's Vulnerability Time Factor (VTF) attribute, which determines how long StorHouse can leave new extents of the file version in the performance buffer before copying them to their primary file set.

- FORMAT: /VTF=NEXT, /VTF=NOW, or /VTF=DIRECT
 - If you specify /VTF=NEXT, the file is written to the performance buffer. The next time a backup occurs, the file is copied to its primary file set.
 - If you specify /VTF=NOW, StorHouse copies the new version from the performance buffer to its primary file set as part of the command.
 - If you specify /VTF=DIRECT, the file bypasses the performance buffer. Extents are written directly to their primary file set.
- DEFAULT: If you omit /VTF, the default is the value of the VTF system parameter.

- **RESTRICTIONS:** If you specify /DIRECT on the command, /VTF is ignored.
- **ACCESS REQUIREMENTS:** Delete access to the group and VTF privilege.

Refer to the CREATE FILE description in the *Command Language Reference Manual* for the estimating file sizes and defining keys sections of this command.

CREATE FSET

The CREATE FSET command creates a file set.

Format

CREATE FSET fset_name

COMMAND FORMAT SUMMARY					
COMMAND, PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
CREATE FSET	ALLOCATION	-	-	-	-
/REPORT	-	-	-	-	-
fset_name	-	-	-	-	(Required)
/AUTO_STAGE	-	-	-	-	/NOAUTO_STAGE
/CONTIGUOUS	-	-	-	-	/CONTIGUOUS
/FORCE_RETENTION	-	-	-	-	/NOFORCE_RETENTION
/LIMIT=...	-	-	-	-	/LIMIT=0
/NONCONTIGUOUS	-	-	-	-	-
/RETENTION=...	-	-	-	-	-
/RPL_CLASS=...	-	-	-	-	DEFAULT
/SIZE=...	-	-	-	-	/SIZE=0
/UPDATE=...	-	-	-	-	/UPDATE=0
/VSET=...	-	-	-	-	See text

Description

CREATE FSET creates a file set, allocates space to the set, and assigns the initial file set attributes.

File sets have either a *contiguous* or *noncontiguous* attribute. A contiguous file set is contained entirely on one volume side or on entire volume sides with any remainder on one side of a volume. When allocating space for a file set with the contiguous attribute, StorHouse allocates entire volume sides unless the LIMIT attribute for the file set restricts an allocation to less than a full side.

When allocating space for a noncontiguous file set, the system may allocate space on more than the minimum number of sides required for the file set size. It allocates all free space that it locates in the volume set until the size specified for the file set is satisfied.

The auto_stage attribute controls whether files in a file set are subject to automatic file staging when they are accessed. Two system parameters, MIG_REPOP_LOAD and

MIG_REPOP_MAX, control whether the auto-staging feature is enabled.
MIG_REPOP_LOAD determines the number of stage requests that can be queued for transfer. MIG_REPOP_MAX specifies the maximum file extent size (in bytes) that can be considered for staging.

When auto-staging is enabled, StorHouse stages file extents with the auto_stage attribute to the performance buffer when they are accessed. If a file extent already resides in the performance buffer, it is not recopied. If the performance buffer contains insufficient space, StorHouse does not stage the file nor does it initiate a regular migration to free up additional space.

The /RETENTION parameter defines the file set retention attribute. This attribute applies to all files (in the file set) that are created without a specified retention attribute or whenever /FORCE_RETENTION is in effect on the file set.

The replication_class_name attribute specifies the name of the replication class for files in the file set. The replication class defines information (replication parameters) about the destination StorHouse system that will contain the file copies.

Parameters

fset_name Specifies a unique file set name in the volume set specified by /VSET.

- **FORMAT:** fset_name

File set names must contain 1 to 8 characters, consisting of the following ASCII characters: A-Z (uppercase), 0-9, _ (underscore), and \$ (dollar sign). StorHouse always translates file set names to uppercase characters, even if they are enclosed in quotes.

- **DEFAULT:** None; you must specify this parameter.

Command Modifiers

/REPORT Controls the generation of special text responses for the completion of significant actions. /REPORT instructs StorHouse to generate a text response. /NOREPORT instructs StorHouse not to generate a text response.

- **FORMAT:** /REPORT or /NOREPORT
- **DEFAULT:** /NOREPORT

Parameter Modifiers

/AUTO_STAGE Assigns or clears the AUTO_STAGE attribute to the file set. You set this attribute to enable StorHouse to automatically stage (copy) the files in the specified file set from Level L storage to the performance buffer for faster access when read from the host.

- FORMAT: /AUTO_STAGE or /NOAUTO_STAGE
 - DEFAULT: /NOAUTO_STAGE
 - RESTRICTIONS: The file set must be a primary file set on Level L storage.
 - PRIVILEGES: None.
- /CONTIGUOUS** Assigns the CONTIGUOUS storage allocation attribute to the file set.
- FORMAT: /CONTIGUOUS
 - DEFAULT: /CONTIGUOUS
 - RESTRICTIONS: /CONTIGUOUS and /NONCONTIGUOUS are mutually exclusive.
- /FORCE_RETENTION** Specifies whether the file set retention value overrides the file retention value (for files in this file set) explicitly supplied by an application at file create time. /NOFORCE_RETENTION indicates that the file set retention value does not override the file retention value.
- FORMAT:
 - /FORCE_RETENTION
 - /NOFORCE_RETENTION
 - DEFAULT: /NOFORCE_RETENTION
 - RESTRICTIONS: /FORCE_RETENTION is valid for primary file sets only.
- /LIMIT** Specifies the maximum number of bytes this file set can contain. A value of zero indicates that the size of the file set is unlimited.
- FORMAT: /LIMIT=number_of_bytes or /LIMIT=0
- You must specify the number_of_bytes value as n, nK, nM, or nG.
- The letter n represents a numeric field. The value of n can range from 0 up to 4294967295. (Do not include commas when specifying a number with more than three digits.) K indicates that the number is in 1,000-byte units; M indicates 1,000,000-byte units; and G indicates 1,000,000,000-byte units. If K, M, or G is not present, the number defaults to 1-byte units.
- DEFAULT: If you omit /LIMIT, the default is /LIMIT=0.
 - RESTRICTIONS: If the value of /LIMIT is not zero, it must be greater than or equal to the value of /SIZE.

/NONCONTIGUOUS Assigns the NONCONTIGUOUS storage allocation attribute to the file set.

- **FORMAT:** /NONCONTIGUOUS
- **DEFAULT:** If you omit /NONCONTIGUOUS, the default is /CONTIGUOUS.
- **RESTRICTIONS:** /NONCONTIGUOUS and /CONTIGUOUS are mutually exclusive.

/RETENTION Specifies the file set retention attribute (retention period). The file set retention attribute applies only to files (in the file set) that are created without a specified retention value, or whenever /FORCE_RETENTION is in effect on the file set.

- **FORMAT:**

Option	Description
/RETENTION=DEFAULT	Sets the retention period to the default value.
/RETENTION=number_of_days	<p>Sets the retention period to the specified number of days. The retention period ends when the current date is beyond the file's last_modified date plus the retention value.</p> <p>A value of 0 indicates no retention period (same as specifying ZERO).</p> <p>Example: /RETENTION=3</p> <p>In this example, the retention period is 3 days. The retention period for a file that was last modified at 11 p.m. on December 12 expires at 11 p.m. on December 15.</p>
/RETENTION=ZERO	Sets no retention period.
/RETENTION=FOREVER	Sets an infinite retention period.

- **DEFAULT:** If you omit /RETENTION, the default retention attribute is DEFAULT.
- **RESTRICTIONS:** /RETENTION is valid for primary file sets only.

/RPL_CLASS Specifies the name of the replication class for files written to this file set. A replication class is a collection of replication-related information (system, file set, volume set, and link names and network device identification) about the target StorHouse system. If the specified replication class does not exist, StorHouse generates a warning message.

- **FORMAT:** /RPL_CLASS=replication_class_name

A replication_class_name can consist of from 1 to 8 of the following ASCII characters: A-Z, 0-9, _, and \$. StorHouse always forces replication class names to uppercase, even when enclosed in quotes.

- **DEFAULT:** If you omit `/RPL_CLASS`, the default is no assigned replication class.
- **RESTRICTIONS:** `/RPL_CLASS` is valid for primary file sets only.

/SIZE Specifies the initial number of bytes to be allocated to the file set. If the specified size is zero, the system allocates no space to the file set until the space is actually needed. A zero size file set is useful only if the file set is extendible.

- **FORMAT:** `/SIZE=number_of_bytes`

You must specify the `number_of_bytes` value as `n`, `nK`, `nM`, or `nG`.

The letter `n` represents a numeric field. The value of `n` can range from 0 up to 4294967295. (Do not include commas when specifying a number with more than three digits.) `K` indicates that the number is in 1,000-byte units; `M` indicates 1,000,000-byte units; and `G` indicates 1,000,000,000-byte units. If `K`, `M`, or `G` is not present, the number defaults to 1-byte units.

- **DEFAULT:** If you omit `/SIZE`, the default is `/SIZE=0`.

/UPDATE Specifies a percentage of the file set size to be set aside for VRAM file updates. The system reserves a percentage of each surface allocation that makes up the set. If the value of this attribute is zero percent, the system reserves no space for updates and uses general free space to store updates.

- **FORMAT:** `/UPDATE=percentage`

The `percentage` can range from 0 to 99 percent.

- **DEFAULT:** `/UPDATE=0`

/VSET Specifies the volume set in which the file set is to be created.

- **FORMAT:** `vset_name`
- **FORMAT RESTRICTIONS:** Wild cards are not allowed.
- **DEFAULT:** If you omit `/VSET`, the default is your current default volume set.

Examples

- To create the contiguous file set `USERFSET` in the volume set `USERVSET` with a `/SIZE` of 1000 MB, a `/LIMIT` of 4000 MB, and a `/RETENTION` of 60 days, enter:

```
? CREATE FSET USERFSET /VSET=USERVSET /SIZE=1000M
/LIMIT=4000M /RETENTION=60
```

Because you did not specify `/UPDATE` (the default is `/UPDATE=0`), no update space is reserved in `USERFSET`.

Assume the following:

- USERFSET is a level L file set.
- Volumes have 3500 MB per side.
- All volumes allocated to the volume set are empty.

Because the file set is contiguous, StorHouse allocates one full volume side (3500 MB) to USERFSET even though only 1000 MB are required to satisfy the specified size request.

If more than 3500 MB are subsequently allocated to files in USERFSET, StorHouse extends USERFSET to another volume side. It allocates all free space on the second side up to USERFSET's 4000 MB limit. Because 3500 MB are allocated on the first side, the file set would have 500 MB allocated on the second side.

- To create the noncontiguous file set USERFSET in the volume set USERVSET with a /SIZE of 0, a /LIMIT of 0, and the AUTO_STAGE attribute, enter:

```
? CREATE FSET USERFSET /NONCONTIGUOUS /VSET=USERVSET  
/SIZE=0 /LIMIT=0 /AUTO_STAGE
```

Because the initial size is zero, StorHouse does not allocate space to the file set until it is required to store a file. USERFSET's size is unlimited as long as its volume set contains space for extension. Because you did not specify /UPDATE, no space is reserved for updates because the default is /UPDATE=0.

When the system allocates space for USERFSET, it allocates only the amount of space required to store the file being written. It allocates the space on the first volume it finds that has at least the required amount of free space; however, the first volumes it searches are those that already have space allocated to the file set.

Because /RETENTION was omitted on the command, the file set retention attribute is DEFAULT.

MIGRATE

StorHouse Release 5.6 changes the MIGRATE command so that MIGRATE /BY_VSET always creates noncontiguous destination file sets. For the complete MIGRATE command description, refer to the *Command Language Reference Manual*.

PUT

The PUT command transfers a copy of a file on host storage to StorHouse, automatically creating a new StorHouse file or file version.

Format

PUT filename

COMMAND FORMAT SUMMARY					
COMMAND PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
PUT	PUT	-	-	-	-
/CONFIRM	-	-	-	-	-
/REPORT	-	-	-	-	/REPORT
filename	- +	W	W ¹	-	(Required)
/ASCII	-	-	-	-	
/ATF=...	ATF	D	D	-	-
/BINARY	-	-	-	-	
/DATA=...	-	-	-	-	-
/DELETE	-	-	2	-	-
/EDC=...	-	-	-	-	See text
/FSET=...	-	-	-	-	Current default
/GROUP=...	SETGROUP	W	-	-	Current default
/HOSTNAME=...	-	-	1	-	-
/LIMIT=...	DELETE	D	D	-	-
/LOCK	LOCK	-	-	-	-
/NEW	-	-	-	-	-
/NEWPASSWORDS=...	PASSWORD	D	D		/NEWP=::
/PASSWORDS=...	-	-	-	-	-
/RETENTION=...	-	-	-	-	DEFAULT
/VSET=...	-	-	-	-	Current default
/VTF=...	VTF	D	D	-	-

¹ You also need read access to the host file.

² You also need delete access to the host file.

Description

PUT creates the specified file in StorHouse and transfers a copy of it from the host to StorHouse. If the file already exists in StorHouse, the system creates a new version of

the file. The file being PUT to StorHouse always becomes the latest version of the StorHouse file.

If you specify `/VTF=DIRECT`, the command writes the file directly to the primary file set, the file set specified by `/VSET` and `/FSET`. If you do not specify `/VTF=DIRECT`, the command reserves storage for the file in the primary file set. Then, the command writes the file to the performance buffer and either allows the backup function to copy it to the primary file set (if `/VTF=NEXT`) or copies it to the primary file set immediately (if `/VTF=NOW`).

If the `/FSET` and `/VSET` modifiers specify or default to a level F file set and volume set, StorHouse does not PUT the file to the performance buffer. The file is PUT directly to the primary file set and volume set on level F, regardless of the value of the `/VTF` modifier, if specified.

StorHouse converts the file to a StorHouse format during the transfer based upon the file's host and file system types, unless you select `/ASCII` or `/BINARY`. In this case, StorHouse converts the file to a transportable format. The original copy of the file on the host is left unchanged unless you specified `/DELETE`.

Parameters

filename Specifies the host file to be transferred to StorHouse and the name of the destination file in StorHouse. You cannot specify VRAM files in this command.

- **FORMAT:** filename
- **FORMAT RESTRICTIONS:** Wild cards are not allowed.
- **DEFAULT:** None; you must specify this parameter.
- **RESTRICTIONS:** If specified, the `/HOSTNAME` modifier helps to determine how the system uses the parameter to derive the StorHouse and host file names.
- **ACCESS REQUIREMENTS:** Write access to the StorHouse file and read access to the destination file on the host. If you specify a modifier that changes passwords or attributes, you must also have delete access to StorHouse file.
- **HOST DEPENDENCIES:** An IBM MVS host user can PUT a dataset with a host-dependent record format if the Data Facility Data Set Services (DFDSS) utility has been installed on the host. When the file is retrieved using GET, the format of the host dataset is the same as it was when the dataset was PUT to StorHouse. For hosts with DFDSS, the format can be ASCII, BINARY, or host-dependent; for all other IBM hosts, it is either ASCII or BINARY.

The UNIX Interactive Interface supports transportable ASCII and host-dependent formats, but it does not support transportable BINARY format.

Command Modifiers

/CONFIRM Controls whether StorHouse asks you to confirm the command.

- **FORMAT:** /CONFIRM or /NOCONFIRM
- **DEFAULT:** /NOCONFIRM

When the system requests a confirmation, enter YES (also Y or YE) or NO (also N). If you press **[Enter↵]** or enter any characters other than those described as a YES response, StorHouse interprets them as NO.

/REPORT Controls the generation of special text responses for the completion of significant actions. /REPORT instructs StorHouse to generate a text response. /NOREPORT instructs StorHouse not to generate a text response.

- **FORMAT:** /REPORT or /NOREPORT
- **DEFAULT:** /REPORT

Parameter Modifiers

/ASCII Causes the host interface to translate a file's data into ASCII characters in a transportable ASCII character-stream file format (FILE TYPE=1, FILE SYSTEM TYPE=65) while transferring the file to StorHouse.

- **FORMAT:** /ASCII
- **DEFAULT:** If you do not select /ASCII, the host interface translates the file into a StorHouse host-dependent format based upon the host and file system types.
- **RESTRICTIONS:** Do not specify /ASCII with a different format indicator, such as /BINARY, or with files that cannot be translated into ASCII characters.
- **HOST DEPENDENCIES:**
 - The IBM MVS Disk File Transfer and Callable Interfaces translate standard EBCDIC-character files to ASCII files.
 - You must specify /ASCII or /BINARY for IBM MVS files if IBM's DFDSS utility is not installed on the host. See the "Parameters" section on page 3-23 for more information about DFDSS.

/ATF Specifies a value for the ATF (Access Time Factor) attribute for a file version. The ATF attribute indicates the importance of access time for the file. Setting an ATF value does not initiate a file transfer directly, but it may cause the file to be migrated in a subsequent migration.

- **FORMAT:** /ATF={1,2,3}

A value of 1 indicates that a short access time to the file is very important; 2 indicates that it is moderately important; 3 indicates that it is minimally important. Files are migrated off the performance buffer, beginning with files with the largest ATF values.

- **DEFAULT:** If you omit /ATF when creating a new file version, the default is the current value of the ATF system parameter.
- **ACCESS REQUIREMENTS:** Delete access to the file and ATF privilege.

/BINARY Translates the file into transportable binary bit-stream format (FILE TYPE = 1, FILE SYSTEM TYPE = 66). The host file must be in a standard host format.

- **FORMAT:** /BINARY
- **DEFAULT:** No /BINARY modifier. If you do not specify /BINARY, StorHouse translates the file into a StorHouse format based upon the host and file system types.
- **RESTRICTIONS:** /ASCII and /BINARY are mutually exclusive; that is, you cannot specify both in the same PUT command.
- **HOST DEPENDENCIES:**
 - You must specify /ASCII or /BINARY for IBM MVS files if IBM's DFDSS utility is not installed on the host.
 - UNIX hosts do not support the /BINARY parameter modifier.

/DATA Specifies an IBM MVS host-dependent parameter that controls the compression of data used from the dataset being transferred to StorHouse.

- **FORMAT:** /DATA=COMPRESS

COMPRESS specifies the use of DFDSS's data compression option. When this option is specified, data PUT on StorHouse is compressed. Decompression is automatic when you issue a GET for the file.

If you specify /ASCII or /BINARY, the system ignores /DATA.

- **DEFAULT:** No /DATA modifier.
- **HOST DEPENDENCIES:**
 - UNIX hosts do not use /DATA.
 - IBM's DFDSS utility must be installed on IBM MVS hosts for /DATA to be a valid modifier; on hosts where DFDSS is not installed, /DATA is ignored.

/DELETE Deletes the source file on the host after the file has been transferred to StorHouse and added to the directory successfully. You must have sufficient privilege and access to the file on the host to delete it, or the StorHouse access program will be unable to delete the file.

- **FORMAT:** /DELETE
- **DEFAULT:** If you omit /DELETE, the source file on the host is left intact.

/EDC Controls the generation of error detection codes (EDCs) by the host interface during transfer to StorHouse. /EDC indicates that the host interface must generate error detection codes for a file according to a specified or default algorithm. /NOEDC indicates that EDC codes should not be generated.

- **FORMAT:**
 - /EDC={0,1,2}
 - /EDC
 - /NOEDC
- **DEFAULT:** If you do not specify /EDC, the default is determined by the EDC system parameter. If the value of the EDC system parameter is TRUE, the default is /EDC. If the value of the parameter is FALSE, the default is /NOEDC. If /EDC is the default or you specify /EDC without assigning it a value, the default value is given by the EDC_TYPE system parameter.

/FSET Specifies the primary file set for the file.

If /FSET and /VSET specify or default to a level F file set and volume set, StorHouse does not PUT the file to the performance buffer. The file is PUT directly to the primary file set and volume set on level F, regardless of the value of the /VTF modifier, if specified.

- **FORMAT:** /FSET=fset_name

The file set name can contain 1 to 8 characters, consisting of the following ASCII characters: A-Z (uppercase), 0-9, _ (underscore), and \$ (dollar sign). StorHouse always translates file set names to uppercase characters, even if they are enclosed in quotes.

- **DEFAULT:** If you do not specify /FSET, the default is your current default file set.
- **RESTRICTIONS:** You cannot specify the performance buffer file set.

/GROUP Specifies a file access group name and, optionally, group passwords.

- **FORMAT:**
 - /GROUP=groupname
 - /GROUP=groupname::writepw

- `/GROUP=groupname::writepw:deletepw`
- **FORMAT RESTRICTIONS:** Wild cards are not allowed in the group name.
- **ACCESS REQUIREMENTS:** Write access to the group. If a file parameter modifier requiring delete access is entered, you must also have delete access to the group.

You must specify the write password unless:

- The group is not protected by a write password.
 - Your privilege bypasses write access password checks.
 - Your default access to the group includes write access.
- **DEFAULT:**
 - If you omit `/GROUP`, the default is your current default group and default access rights.
 - If you specify the current default group name and do not specify a write password, the defaults for your group access rights apply.
 - If you specify a group name that is not the current default group and do not specify a write password, the write password defaults to null.
 - **PRIVILEGE:** You must have `SETGROUP` privilege to specify any group except your default group.

/HOSTNAME The combination of the filename and `/HOSTNAME` specify a StorHouse file and a host file to be used in the command.

If you assign the optional `host_filespec` to `/HOSTNAME`, the StorHouse host interface translates or expands the `host_filespec` on the host in your current environment to produce a host file specification. The host interface uses the result to specify a file in the host only. StorHouse uses the `filename` to specify a file in StorHouse. If either file specification is not valid, StorHouse returns an error response.

`/HOSTNAME` always modifies a filename and can have a host file specification assigned to it.

- **FORMAT:**
 - filename `/HOSTNAME`
 - filename `/HOSTNAME=host_filespec`

The filename and `host_filespec` must contain 1 to 56 printable ASCII characters. At least one character must be non-blank.

StorHouse translates lowercase letters to uppercase letters and compresses multiple consecutive spaces to a single space, unless these characters are enclosed in quotes. Because lowercase letters are distinct from uppercase, lowercase letters must be enclosed in quotes if they are part of the filename or host_filespec. All special characters must be enclosed in quotes.

- **FORMAT RESTRICTIONS:** Wild cards are not allowed in the filename or host_filespec.
- **DEFAULT:** If you do not specify /HOSTNAME, StorHouse uses the filename to specify a file in StorHouse. The host interface also uses the filename as if it were a host_filespec assigned to /HOSTNAME, as described above.

If you specify /HOSTNAME without the optional host_filespec, the host interface uses the filename as if it were a host_filespec assigned to /HOSTNAME, as described above. In addition, the interface returns all or part of the translated host file specification to StorHouse to specify a file in StorHouse in place of the filename. (The host interface determines which part is returned.)

- **ACCESS REQUIREMENTS:** Write access to the StorHouse file and read access to the host file. If you specify a modifier that changes passwords or attributes, you must also have delete access to the StorHouse file.
- **HOST DEPENDENCIES:** The IBM MVS Host Interactive Interface uses standard TSO prefixing to translate host_filespec to a dataset specification. This includes the ability to avoid prefixing by enclosing host_filespec in apostrophes. In addition, the Interface translates all host_filespec values to uppercase, and, if you do not specify the StorHouse file name, the Interface passes the entire translated host dataset specification to StorHouse to be used as the file name. Table 2-1 illustrates the StorHouse – IBM MVS dataset name translation. In these examples, your TSO prefix is assumed to be U1.

Table 2-1: StorHouse – IBM MVS Dataset Name Translation

Parameter/Modifier	StorHouse File Name	IBM MVS Data Set
FA.FB /HOSTNAME=F1.F2	FA.FB	U1.F1.F2
FA.FB /HOSTNAME='F1.F2'	FA.FB	F1.F2
FA.FB /HOSTNAME='U1.F1.F2'	FA.FB	U1.F1.F2
U1.F1.F2 /HOSTNAME=F1.F2	U1.F1.F2	U1.F1.F2
'F1.F2' /HOSTNAME=F1.F2	'F1.F2'	U1.F1.F2
F1.F2	F1.F2	U1.F1.F2
U1.F1.F2	U1.F1.F2	U1.U1.F1.F2

Table 2-1: StorHouse – IBM MVS Dataset Name Translation (continued)

Parameter/Modifier	StorHouse File Name	IBM MVS Data Set
'F1.F2'	'F1.F2'	F1.F2
F1.F2. /HOSTNAME	U1.F1.F2	U1.F1.F2
U1.F1.F2 /HOSTNAME	U1.U1.F1.F2	U1.U1.F1.F2
'F1.F2' /HOSTNAME	F1.F2	F1.F2

The UNIX Host Interactive Interface translates file names using the standard UNIX rules of environment variable substitution. No other processing is performed. Any path names specified in `host_filespec`, either explicitly or as the result of variable substitution, are used as part of the file specification. Table 2-2 illustrates UNIX file name translations. The examples assume that environment variable `FNAME1` is defined as `filename1`, `fname2` is defined as `/usr/FILENAME2.ext`, and `PNAME` is defined as `/usr`.

Table 2-2: StorHouse – UNIX File Name Translation

Parameter/Modifier	StorHouse File Name	UNIX File Name
FNAME1	FNAME1	FNAME1
\$FNAME1	\$FNAME1	filename1
"filename1"	filename1	filename1
"fname2"	fname2	fname2
"\$fname2"	\$fname2	/usr/FILENAME2.ext
"/usr/FILENAME2.ext"	/usr/FILENAME2.ext	/usr/FILENAME2.ext
"PNAME/FNAME1"	PNAME/FNAME1	PNAME/FNAME1
"\$PNAME/FNAME1"	\$PNAME/FNAME1	/usr/FNAME1
"PNAME/\$FNAME1"	PNAME/\$FNAME1	PNAME/filename1
"\$PNAME/\$FNAME1"	\$PNAME/\$FNAME1	/usr/filename1
FNAME1 /HOSTNAME="fname2"	FNAME1	fname2
\$FNAME1 /HOSTNAME="\$fname2"	\$FNAME1	/usr/FILENAME2.ext
FNAME1 /HOSTNAME	FNAME1	FNAME1

Table 2-2: StorHouse – UNIX File Name Translation (continued)

Parameter/Modifier	StorHouse File Name	UNIX File Name
\$FNAME1 /HOSTNAME	filename1	filename1
"\$fname2" /HOSTNAME	/usr/FILENAME2.ext	/usr/FILENAME2.ext
"\$PNAME/\$FNAME1" /HOSTNAME	/usr/filename1	/usr/filename1

Unless you specify a path, either explicitly or as part of an environment variable specification, your current directory is used for the source.

/LIMIT Specifies a value for the LIMIT attribute for a file. A new value for /LIMIT that is lower than the current number of versions of an existing file takes effect immediately.

- **FORMAT:** /LIMIT=maximum_versions

The value of maximum_versions can range from 1 through 32768.

- **DEFAULT:** If you omit /LIMIT and the file is new, the default is the value of the LIMIT system parameter. If the file is a new version of an existing file, the default is the value of /LIMIT for the existing file.
- **ACCESS REQUIREMENTS:** Delete access to the file and DELETE privilege.

/LOCK Locks a file version under your account upon completion of the PUT command. Other accounts cannot use the locked file version until it is unlocked.

An explicitly locked file cannot be migrated or written back from the performance buffer to its primary file set.

- **FORMAT:** /LOCK
- **DEFAULT:** No /LOCK modifier.
- **PRIVILEGE:** LOCK privilege.

/NEW Indicates that the file must be created in StorHouse and must not be a new version of an existing file. If a file of the same name already exists in the destination file access group, the PUT will terminate with an error.

- **FORMAT:** /NEW
- **DEFAULT:** No /NEW modifier, which creates a new version of the file if the file name already exists in StorHouse.

/NEWPASSWORDS Specifies all new passwords or replaces all current passwords for a file. You cannot obtain access to current passwords by specifying /NEWPASSWORDS (see /PASSWORDS below).

- **FORMAT:**

- `/NEWPASSWORDS=<:<readpw><:<writepw><:<deletepw>>>>`
- `/NONEWPASSWORDS`

Specifying `/NONEWPASSWORDS` is equivalent to specifying `/NEWPASSWORDS` with null read, write, and delete passwords.

Passwords must be null or contain 1 to 8 characters, and consist of the following ASCII characters: A-Z (uppercase), 0-9, _ (underscore), and \$ (dollar sign). StorHouse always translates passwords to uppercase characters, even if they are enclosed in quotes.

- **DEFAULT:** If you omit `/NEWPASSWORDS` and if the file already exists, the system retains the existing passwords, if any. If you omit the modifier and the file does not exist, the system assigns null passwords to it.

If you specify `/NEWPASSWORDS` but do not specify one or more passwords, the system assigns a null password for each unspecified password.

- **ACCESS REQUIREMENTS:** You must have delete access to the file and `PASSWORD` privilege.

/PASSWORDS Specifies file access passwords to gain write and, if necessary, delete access to the file.

- **FORMAT:** `/PASSWORDS=<:<writepw><:<deletepw>>`

You must specify a write password to obtain write access to the file and a delete password to obtain delete access unless:

- The file is not protected by a delete password.
- Your privilege bypasses delete password checks.
- **DEFAULT:** If you omit `/PASSWORDS`, the passwords default to nulls.

/RETENTION Specifies the retention attribute (retention period) for the file.

- **FORMAT:**

Option	Description
/RETENTION=DEFAULT	Sets the retention period to the default value.
/RETENTION=number_of_days	<p>Sets the file retention period to the specified number of days. The retention period ends when the current date is beyond the file's last_modified date plus the specified retention value.</p> <p>A value of 0 indicates no retention period (same as specifying ZERO).</p> <p>Example: /RETENTION=3</p> <p>In this example, the retention period is 3 days. The retention period for a file that was last modified at 11 p.m. on December 12 expires at 11 p.m. on December 15.</p>
/RETENTION=ZERO	Sets no retention period, which indicates the file may be deleted.
/RETENTION=FOREVER	Sets an infinite retention period, which indicates the file may never be deleted.

- **DEFAULT:** If you omit **/RETENTION** or specify **/RETENTION=DEFAULT**, StorHouse determines the default file retention attribute as follows:
 - If the file's resident file set has a retention attribute equal to **FOREVER**, **ZERO**, or a specified number of days, the file set retention attribute determines the default file retention attribute.
 - If the file's resident file set has a retention attribute of **DEFAULT**, the **RETENTION_MODE** system parameter determines the default file retention attribute. If **RETENTION_MODE** is set to **BASIC**, the default file retention is **ZERO**. If **RETENTION_MODE** is set to **STRICT**, the default file retention is **FOREVER**.
 - **RESTRICTIONS:** None.

/VSET Specifies the primary volume set where the file's destination file set is located.

If **/VSET** and **/FSET** specify or default to a level F volume set and file set, StorHouse does not PUT the file to the performance buffer. The file is PUT directly to the primary file set and volume set on level F, regardless of the value of **/VTF**, if specified.

- **FORMAT:** **/VSET=vset_name**

- **DEFAULT:** If you do not specify the modifier, the default is your current default volume set.
- **RESTRICTIONS:** The volume set must be a primary volume set.

/VTF Specifies the file's Vulnerability Time Factor (VTF) attribute, which determines how long StorHouse can leave new extents of the file version in the performance buffer before copying them to their primary file set.

- **FORMAT:** /VTF=NEXT, /VTF=NOW, or /VTF=DIRECT

Note that NEVER is no longer a valid value for /VTF. If you specify /VTF=NEVER, the value is accepted and converted to NEXT.

If you specify /VTF=NEXT, the file is written to the performance buffer. The next time a backup occurs, the file is copied to its primary file set.

If you specify /VTF=NOW, StorHouse copies the new version from the performance buffer to its primary file set as part of the command.

Note: If you specify /VTF=NOW and the PUT *fails* after the file has been transferred from the host to the performance buffer but before the file is written back to its primary file set, StorHouse keeps the performance buffer copy and notifies you of the error. However, if the PUT is *aborted* after the file has been transferred from the host to performance buffer but before the file is written back to its primary file set, StorHouse does *not* keep the performance buffer copy.

If you specify /VTF=DIRECT, the file bypasses the performance buffer. Extents are written directly to their primary file set.

- **DEFAULT:** If you omit /VTF, the default for a new file is the value of the VTF system parameter. For an existing file, the system uses the VTF value of the latest version of the file.
- **PRIVILEGE:** Requires VTF privilege.
- **ACCESS REQUIREMENTS:** Delete access to the group and file.

Examples

The following host-dependent command examples show typical user applications of PUT.

IBM MVS Example 1. To copy the dataset USERFILE from the host to the StorHouse file, enclosing the dataset name in single quotes and specifying /HOSTNAME to prevent your TSO prefix from being added to StorHouse file name; specifying USERFILE's ATF attribute as 3 (meaning that a short access time is not an important factor for the file); specifying /EDC to indicate that error detection codes are to be generated during file transfer to StorHouse; specifying USERFILE's write

password as WRITEPW and its delete password as DELETEPW; and specifying /VTF=NEXT so that StorHouse will back up USERFILE at the next scheduled backup, enter:

```
? PUT 'USERFILE' /HOSTNAME /ATF=3 /EDC
/PASSWORDS=:WRITEPW:DELETEPW /VTF=NEXT
```

Because you did not specify an EDC type, the system parameter EDC_TYPE determines the type of error detection code algorithm that is used.

If your account does not have ANYFILE privilege, you must specify the write password on a PUT to allow the password to be bypassed. You must have delete access to specify ATF and VTF values.

Because you omitted /FSET, /GROUP, and /VSET on the command, USERFILE is placed in your default access group. StorHouse allocates space for USERFILE in your default file set and volume set.

IBM MVS Example 2. To copy the dataset FILE2 from the host to StorHouse, specifying a HOSTNAME of FILE1; specifying /LIMIT=10 so that StorHouse retains a maximum of 10 versions of FILE2; specifying the file's existing write password, WRITE, because you do not have ANYFILE privilege; specifying its existing delete password, DELETE, to give you delete access in order to specify /LIMIT and /NEWPASSWORDS; and specifying FILE2's new read, write, and delete passwords as NEWREAD, NEWWRITE, and NEWDEL, respectively, enter:

```
? PUT FILE2 /HOSTNAME=FILE1 /LIMIT=10 /PASSWORDS=:WRITE:DELETE
/NEWPASSWORDS=NEWREAD:NEWWRITE:NEWDEL
/RETENTION=FOREVER
```

Because FILE1 is not enclosed in single quotes, the name of the host dataset includes your TSO prefix. Therefore, if your prefix is USER, the dataset name is USER.FILE1. The name of the file on StorHouse is FILE2.

If the number of versions of FILE2 exceeds 10, the oldest version(s) are marked as deleted until only 10 remain.

Because you omitted /FSET, /GROUP, and /VSET on the command, FILE2 is placed in your default file access group. StorHouse allocates space for FILE2 in your default file set and volume set.

The file retention value is FOREVER, which means the file can never be deleted.

UNIX Host Example. To copy to StorHouse the contents of the host file whose name has been defined as the environment variable LOGICALNAME, specifying /HOSTNAME so that the file name on StorHouse will be the same name as the actual host file name; specifying /LIMIT=10 so that StorHouse will retain a maximum of 10 versions of /maindir/userfile.dat; specifying the file's existing write password, WRITE, because you do not have ANYFILE privilege; specifying its existing delete password, DELETE, to give you delete access in order to specify /LIMIT and

/NEWPASSWORDS; and specifying the file's new read, write, and delete passwords as READ, WRITE, and DELETE, respectively, enter:

```
? put $logicalname /hostname /limit=10 /passwords=:write:delete  
/newpasswords=newread:newwrite:newdel
```

If the host file name is /maindir/userfile.dat, the name of the file on StorHouse is /maindir/userfile.dat.

If more than 10 versions of /maindir/userfile.dat are PUT to StorHouse, the oldest versions are moved to the deleted file directory until only 10 remain.

Because you omitted /FSET, /GROUP, and /VSET on the command, /maindir/userfile.dat is placed in your default file access group. StorHouse allocates space for /maindir/userfile.dat in your default file set and volume set.

Because /RETENTION was omitted on the command, the default file retention attribute is DEFAULT. The exact value will be determined by the file set retention attribute or the value of the RETENTION_MODE system parameter.

REPLICATE

The REPLICATE command copies eligible files from one StorHouse system to another.

FORMAT

REPLICATE filename

COMMAND FORMAT SUMMARY					
COMMAND, PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
REPLICATE	SYSTEM	-	-	-	-
/CONFIRM	-	-	-	-	-
/CHECK	-	-	-	-	-
/DEF_RPL_CLASS=...	-	-	-	-	-
/PREVIEW	-	-	-	-	-
/RECORD	-	-	-	-	/RECORD
/REPORT	-	-	-	-	-
/WAIT	-	-	-	-	-
filename	-	-	-	-	(required)
/FSET=...	-	-	-	-	/FSET=*
/GROUP=...	SETGROUP		-	-	/GROUP=*
/VERSION=...	-	-	-	-	/VERSION=*
/VSET=...	-	-	-	-	/VSET=*

Description

REPLICATE copies one or more eligible files from the primary directory on one StorHouse system to the primary directory on a destination, or target, StorHouse system. If the primary copy is unavailable on the source system, StorHouse accesses the duplex copy to create the replica.

Replication works as follows. When an application creates or modifies a replication-eligible file, StorHouse queues the same action to occur on the destination StorHouse system the next time the REPLICATE command executes on the source system and selects that file. When a file is deleted (and removed) from the source system, StorHouse queues the same action to occur on the destination StorHouse the next time the REPLICATE command executes on the source system and selects any file.

For convenience, you can schedule the REPLICATE command to run periodically.

REPLICATE uses the same account, password, and file access group on the source and target systems. The system administrator must manually create the respective

accounts and volume sets on the target location. If the required groups and file sets do not already exist on the destination system, StorHouse creates them automatically. System-created file sets are noncontiguous and use the other CREATE FSET command defaults.

Files are eligible for replication when they have a pre-assigned replication class or an explicitly specified default replication class (see the /DEF_RPL_CLASS modifier on page 2-38). The replication class defines information about the target StorHouse system (for example, the system, file set, volume set, and network link names and the network device identifier).

Note the following:

- REPLICATE requires SYSTEM privilege on both the source and target systems.
- A file and its replica may have different version numbers on the source and destination systems.
- The source and destination systems do not need to run the same StorHouse release as long as both releases support replication.
- Files queued for replication are not written to the destination system in any particular order or within a set time period.
- A file is considered replicated when at least one copy resides on the destination system, including on the performance buffer.
- Replicated files retain many of their original attributes including file name, group, FID, creation date/time, and retention characteristics.
- StorHouse does not copy file and file set replication class attributes from the source system to the target location. A replica inherits the replication class attribute of its target file set.
- StorHouse can replicate any type of primary file (VRAM, sequential, or STORHOUSE).

Parameters

filename Specifies the name of the file or files to be replicated.

- FORMAT:
 - filename
 - partial_filename*
 - *

StorHouse file names must contain 1 to 56 printable ASCII characters. At least one character must be non-blank. Lowercase characters are distinct from uppercase characters. File names must be unique within a file access group.

The wild card is valid only if you specify it as the last or only character in the file name.

- DEFAULT: None, you must specify this parameter.

Command Modifiers

/CHECK Tells StorHouse to ignore a file's replicated flag and check whether every specified file has already been replicated on the destination StorHouse system. Unless necessary, do not use /CHECK because it increases processing time significantly.

- FORMAT: /CHECK
- DEFAULT: If /CHECK is omitted, StorHouse checks the replicated flag of only the file queued for replication to determine whether to copy it to the target StorHouse system.

/CONFIRM Controls whether StorHouse asks you to confirm the replication of each individual file.

- FORMAT: /CONFIRM or /NOCONFIRM
- DEFAULT: /NOCONFIRM

When the system requests a confirmation, enter YES (also Y or YE) or NO (also N). If you press **Enter** or enter any characters other than those described as a YES response, StorHouse interprets them as NO.

If the command is to be performed for more than one item, the system generally requests a confirmation of each item and allows an END (entered as E, EN, or END) response. END directs the system not to perform the command for the current item and any further items.

/DEF_RPL_CLASS Specifies the name of the default replication class for files with no pre-assigned replication class. This feature enables one-time, ad hoc replication of files that otherwise would not be copied to the target StorHouse system. If the specified replication class does not exist, StorHouse generates an error message.

- FORMAT: /DEF_RPL_CLASS=replication_class_name

A replication_class_name must consist of 1 to 8 of the following ASCII characters: A-Z (uppercase), 0-9, _ (underscore), and \$ (dollar sign). StorHouse always translates replication class names to uppercase characters, even if enclosed in quotes.

- **DEFAULT:** If you omit /DEF_RPL_CLASS, StorHouse does not copy any file without a pre-assigned replication class to the target StorHouse system.
- /PREVIEW** Indicates that REPLICATE will display the number of files and bytes that will be replicated when the command is executed. REPLICATE/PREVIEW does not display individual file names or copy files to the target StorHouse system.
- **FORMAT:** /PREVIEW
 - **DEFAULT:** If you omit /PREVIEW, the command copies selected eligible files to the target StorHouse system.
- /RECORD** Indicates whether the command will change the replicated flag in the primary directory of the source StorHouse system when it copies a file version to the target StorHouse system. /RECORD instructs the command to mark each replicated file version as having a current replica. /NORECORD instructs the command not to change the replicated flag.
- **FORMAT:** /RECORD or /NORECORD
 - **DEFAULT:** /RECORD
- /REPORT** Controls the generation of special text responses for the completion of significant actions. /NOREPORT instructs StorHouse not to generate text responses.
- **FORMAT:** /REPORT or /NOREPORT
 - **DEFAULT:** /NOREPORT
- /WAIT** Instructs StorHouse to wait for a locked file to be unlocked before attempting to use it in the command execution.
- **FORMAT:** /WAIT or /NOWAIT
 - **DEFAULT:** /NOWAIT

Parameter Modifiers

- /FSET** Replicates eligible files in the specified primary file set (and volume set) to the target StorHouse system.
- **FORMAT:** /FSET=fset_name or /FSET=*
 - **DEFAULT:** If you omit /FSET, the default is all file sets (/FSET=*).
 - **RESTRICTIONS:**
 - If you specify /FSET, you must also specify /VSET.

- If you specify the performance buffer file set name, StorHouse does not select any files for replication.

/GROUP Replicates eligible files in the specified file access group to the target StorHouse system.

- **FORMAT:**
 - /GROUP=groupname
 - /GROUP=partial_groupname*
 - /GROUP=*
- **DEFAULT:** If you omit /GROUP, the default is all groups (/GROUP=*).
- **PRIVILEGE:** You must have SETGROUP privilege to specify any group except your default group.

/VERSION Specifies the relative version number of files to be replicated.

- **FORMAT:** /VERSION=version or /VERSION=*
- **/DEFAULT:** If you omit /VERSION, the default is all versions (/VERSIONS=*).

/VSET Replicates eligible files in the specified primary volume set to the target StorHouse system.

- **/FORMAT:** /VSET=vset_name or /VSET=*
- **DEFAULT:** If you omit /VSET, the default is all volume sets (/VSET=*).

Examples

- To replicate all eligible files and wait for locked files to be unlocked before attempting to use them in the command execution, enter:

? REPLICATE * /WAIT
- To check whether version 0 of the file USERFILE in the file access group ACCT already exists on the target system and then copy it there, if required, enter:

? REPLICATE USERFILE /CHECK /GROUP=ACCT /VERSION=0
- To preview the number of files and bytes that will be replicated to the target StorHouse system from volume set MAR_2004 and to generate a text response for each processed file, enter:

? REPLICATE * /VSET=MAR_2004 /REPORT /PREVIEW

SET DEVICE

SET DEVICE lets you set or clear the read-only device mode for level F or level L devices. In addition, the command supports dynamic changes to the size of fixed content object-based disk and filesystem level F drives.

Format

SET DEVICE did

COMMAND FORMAT SUMMARY					
COMMAND, PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
SET DEVICE	OPERATOR	-	-	-	-
/CONFIRM	-	-	-	-	-
/REPORT	-	-	-	-	-
did	-	-	-	-	(Required)
/READ_ONLY	-	-	-	-	-
/SIZE=...	SYSTEM	-	-	-	-

Description

SET DEVICE lets you set or clear the read-only device mode of a level F or level L drive. In addition, SET DEVICE supports dynamic changes to the size of fixed content object-based disk or filesystem level F drives.

You can invoke the SET DEVICE command interactively or schedule it to run periodically or at a specific time using the SCHEDULE command.

Parameters

- did Specifies the device identification code for the drive affected by the command.
- FORMAT: Specifies the device identification code (did) for the drive, which must include a device level of F or L and a device unit number. For library device drives, the did must also include a subunit type of D and a subunit number (the braces “{ }” are not part of the specification):

F{unit_number}
 L{unit_number}D{subunit_number}
 - DEFAULT: None; you must specify this parameter.

Command Modifiers

/CONFIRM Controls whether StorHouse asks you to confirm the command.

- **FORMAT:** /CONFIRM or /NOCONFIRM
- **DEFAULT:** /NOCONFIRM

When the system requests a confirmation, enter YES (also Y or YE) or NO (also N). If you press **Enter** or enter any characters other than those described as a YES response, StorHouse interprets them as NO.

/REPORT Controls the generation of special text responses for the completion of significant actions. /REPORT instructs StorHouse to generate a text response. /NOREPORT instructs StorHouse not to generate a text response.

- **FORMAT:** /REPORT or /NOREPORT
- **DEFAULT:** /NOREPORT

Parameter Modifiers

/READ_ONLY Changes the mode of a level F or level L drive. /READ_ONLY instructs StorHouse to set the drive to read-only mode. /NOREAD_ONLY instructs StorHouse to clear read-only mode for the specified drive.

- **FORMAT:** /READ_ONLY or /NOREAD_ONLY
- **DEFAULT:** If you omit this modifier, the read-only mode of the drive is unaffected.

/SIZE Specifies the size of fixed content object-based disk or filesystem level F drives.

- **FORMAT:** /SIZE=number_of_bytes

The value for number_of_bytes can range up to 4294967295 (do not use commas in the specification). Use K to indicate 1,000-byte units, M to indicate 1,000,000-byte units, and G to indicate 1,000,000,000-byte units. If you do not specify K, M, or G, the number defaults to 1-byte units.

- **DEFAULT:** If you omit /SIZE, the size attribute is not changed.
- **RESTRICTIONS:** /SIZE can only be used to specify the size of fixed content object-based disk and filesystem level F drives. It is not valid for other types of level F devices or for level L devices.

The value of /SIZE may not be less than the amount of space currently allocated on the device.

Examples

- To set drive L00D01 to read-only mode, enter:
? SET DEVICE L00D01 /READ_ONLY
- To clear read-only mode for drive L00D01, enter:
? SET DEVICE L00D01 /NOREAD_ONLY
- To set drive F00 to read-only mode, enter:
? SET DEVICE F00 /READ_ONLY
- To set the size of F00 to 500 GB, enter:
? SET DEVICE F00 /SIZE=500G

SET FILE

The SET FILE command changes file attributes and passwords.

Format

SET FILE filename

COMMAND FORMAT SUMMARY					
COMMAND, PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
SET FILE	FILE	-	-	-	-
/REPORT	-	-	-	-	-
/WAIT	-	-	-	-	-
filename	-	D	D	-	(Required)
/ARCHIVED	-	-	-	-	-
/ATF=...	ATF	-	-	-	-
/BACKUP	-	-	-	-	-
/FSET=...	-	-	-	-	-
/GROUP=...	SETGROUP	D	-	-	Current default
/LIMIT=...	DELETE	-	-	-	-
/NEWPASSWORDS=...	PASSWORD	-	-	-	-
/NOREPLICATED	SYSTEM	-	-	-	-
/PASSWORDS=...	-	-	-	-	-
/RELINK=...	-	-	-	-	-
/RETENTION=...	-	-	-	-	-
/RPL_CLASS=...	-	-	-	-	-
/VERSION=...	-	-	-	-	See text
/VOLUME=...	-	-	-	-	-
/VSET=...	-	-	-	-	-
/VTF=...	VTF	-	-	-	-

Description

SET FILE allows you to change attributes and passwords for a file in the primary directory. It does not change directory information in the archive or backup directory.

SET FILE changes to version-dependent attributes only apply to the specified (or default) file version. They have no effect on other versions of the file.

The version-dependent attributes are:

- ARCHIVED
- ATF
- BACKUP (file version is a candidate for being archived or backed up)
- BACKED UP (users cannot directly set this attribute)
- NOREPLICATED
- RETENTION
- RPL_CLASS
- VTF

You can schedule SET FILE using the SCHEDULE command.

Parameters

filename Specifies the file to be modified.

- FORMAT:
 - filename
 - partial_filename*
 - *

The wild card is valid only if you specify it as the last or only character in the name.

- DEFAULT: None; you must specify this parameter.
- ACCESS REQUIREMENTS: Delete access to the file and group.

Command Modifiers

/REPORT Controls the generation of special text responses for the completion of significant actions. /REPORT instructs StorHouse to generate a text response. /NOREPORT instructs StorHouse not to generate a text response.

- FORMAT: /REPORT or /NOREPORT
- DEFAULT: /NOREPORT

/WAIT Instructs StorHouse to wait for a locked file to be unlocked before attempting to use it in the command execution.

- FORMAT: /WAIT or /NOWAIT
- DEFAULT: /NOWAIT

Parameter Modifiers

/ARCHIVED Changes the archived status of a file version.

The command changes the status in the primary directory, but does not verify that the status is consistent with the archive directory.

ARCHIVED is a version-dependent attribute and has no effect on other versions of the file. The /VERSION modifier determines which version is changed.

- **FORMAT:** /ARCHIVED or /NOARCHIVED
- **DEFAULT:** If you omit /ARCHIVED, the archived status of the file version is not changed.

/ATF Specifies a value for the ATF (Access Time Factor) attribute for a file version. The ATF attribute indicates the importance of access time for the file. Setting an ATF value does not initiate a file transfer directly, but it may cause the file to be moved in a subsequent migration.

ATF is a version-dependent attribute and has no effect on other versions of the file. The /VERSION modifier determines which version is changed.

- **FORMAT:** /ATF={1,2,3}
- A value of 1 indicates that a short access time for the file is very important; 2 indicates that access time is moderately important; 3 indicates that it is minimally important. Files are migrated off the performance buffer based on their ATF attributes, beginning with files with the largest ATF values.
- **DEFAULT:** If you omit /ATF, the value of the file's ATF attribute is not changed.
- **ACCESS REQUIREMENTS:** Delete access to the file.

/BACKUP Controls whether the file version is to be selected for backup by the CREATE BACKUP and ARCHIVE commands. /BACKUP indicates that the version is to be backed up. /NOBACKUP indicates that the version is not to be backed up.

BACKUP is a version-dependent attribute and has no effect on other versions of the file. The value of the /VERSION modifier determines which version is changed.

- **FORMAT:** /BACKUP or /NOBACKUP
- **DEFAULT:** If you omit /BACKUP, the BACKUP attribute of the file version is not changed.

/FSET Indicates that the command will only select files in the specified file set (and volume set).

- **FORMAT:** /FSET=fset_name

- **FORMAT RESTRICTIONS:** Wildcards are not allowed.
- **DEFAULT:** If you omit /FSET, StorHouse does not use the file set name to select files.
- **RESTRICTIONS:** If you specify /FSET, you must also specify /VSET.

/GROUP Specifies the file's access group name and, optionally, the group's delete password.

- **FORMAT:**
 - /GROUP=groupname<:::deletepw>
 - /GROUP=partial_groupname*
 - /GROUP=*
- **ACCESS REQUIREMENTS:** Delete access to the group.

You must specify the delete password unless:

- The group is not protected by a delete password.
- Your privilege bypasses delete access password checks.
- Your default access to the group includes delete access.

Passwords are not allowed if the group name contains a wild card.

- **DEFAULT:** If you omit /GROUP, the default is your current default group.
- **PRIVILEGE:** You must have SETGROUP privilege to specify any group except your default group.

/LIMIT Specifies a value for the LIMIT attribute for a primary file. A new value for /LIMIT that is less than the existing value takes effect the next time a PUT or PURGE of the file is executed. However, SHOW FILE displays the new LIMIT attribute for all file versions immediately.

LIMIT applies to primary files only. (The limit for archive and backup copies of files is always 65536.)

- **FORMAT:** /LIMIT=maximum_versions

The value of maximum_versions can range from 1 through 32768.

- **DEFAULT:** If you omit /LIMIT, the default is to keep the current value of /LIMIT for the file.

/NEWPASSWORDS Replaces all current passwords for a file. Note that you cannot gain access to passwords by specifying /NEWPASSWORDS (see /PASSWORDS).

- **FORMAT:**

- /NEWPASSWORDS=<readpw>:<writepw>:<deletepw>
- /NEWPASSWORDS=<readpw>:<writepw>
- /NEWPASSWORDS=readpw
- /NONEWPASSWORDS

Specifying /NONEWPASSWORDS is equivalent to specifying /NEWPASSWORDS with null read, write, and delete passwords.

A file password can be null or contain 1 to 8 characters, consisting of the following ASCII characters: A-Z (uppercase), 0-9, _ (underscore), and \$ (dollar sign). StorHouse always translates passwords to uppercase characters, even if they are enclosed in quotes.

- DEFAULT: If you omit /NEWPASSWORDS, the system retains the existing passwords, if any.

If you specify /NEWPASSWORDS but do not specify one or more passwords, the system assigns a null password for each unspecified password.

- ACCESS REQUIREMENTS: Delete access to the file and PASSWORD privilege.

/NOREPLICATED Clears the replicated status of a file version.

/NOREPLICATED is a version-dependent attribute. The /VERSION modifier determines which version is changed.

- FORMAT: /NOREPLICATED
- DEFAULT: If you omit /NOREPLICATED, the replicated status of the file version is not changed.

PRIVILEGE: You need SYSTEM privilege to specify /NOREPLICATE.

/PASSWORDS Specifies the file's delete password.

- FORMAT: /PASSWORDS=::deletepw

You must specify a delete password unless:

- The file is not protected by a delete password.
- Your privilege bypasses delete password checks.
- DEFAULT: If you omit /PASSWORDS, the password defaults to nulls.

/RELINK Indicates that the command will check the BACKUP or ARCHIVE directory for copies of the specified file versions and mark the file versions in the PRIMARY directory as backed up or archived where appropriate. If the backup (or archive) copy exists, the command will mark its primary as backed up (or archived) even if the primary has the NOBACKUP attribute.

The value of the /VERSION modifier determines which version or versions are checked.

- **FORMAT:** /RELINK=ARCHIVE, /RELINK=BACKUP, or /RELINK
- **DEFAULT:**
 - If you omit /RELINK, the system does not attempt to relink primary with backup and archive copies of file versions.
 - If you specify /RELINK without giving a directory, the command uses the directory specified by the DUPLEX_DIRECTORY system parameter.
- **RESTRICTION:** /RELINK is mutually exclusive with /NOARCHIVE.
-

/RETENTION Specifies the retention attribute (retention period) for the file.

RETENTION is a version-dependent attribute. The /VERSION modifier determines which version is changed.

- **FORMAT:**

Option	Description
/RETENTION=DEFAULT	Sets the retention period of the specified file version to the default value.
/RETENTION=number_of_days	Sets the retention period of the specified file version to the specified number of days. The retention period ends when the current date is beyond the file's last_modified date plus the specified retention value. A value of 0 indicates no retention period (same as specifying ZERO). Example: /RETENTION=3 In this example, the retention period is 3 days. The retention period for a file that was last modified at 11 p.m. on December 12 expires at 11 p.m. on December 15.
/RETENTION=ZERO	Sets no retention period, which indicates the specified file version may be deleted.
/RETENTION=FOREVER	Sets an infinite retention period, which indicates the specified file version may never be deleted.

- **DEFAULT:** If you omit /RETENTION, there is no change to the file's current retention value.

- **RESTRICTIONS:** You may only specify a more restrictive value. For example, if the current file retention is 100 days, you may specify 200 days but not 50 days.

/RPL_CLASS Specifies the name of the replication class for the specified file. A replication class is a collection of replication-related information (system, file set, volume set, and link names and network device identification) about the target StorHouse system. **/NORPL_CLASS** specifies that the file has no replication class. If the specified replication class does not exist, StorHouse generates a warning message.

/RPL_CLASS is a version-dependent attribute. The **/VERSION** modifier defines the version that is assigned the specified replication class.

- **FORMAT:** **/RPL_CLASS=replication_class_name** or **/NORPL_CLASS**

A **replication_class_name** can consist of from 1 to 8 of the following ASCII characters: A-Z, 0-9, _, and \$. StorHouse always forces replication class names to uppercase, even when enclosed in quotes.

- **DEFAULT:** If **/RPL_CLASS** is omitted, the default is not to change the replication class.

/VERSION Specifies the relative version number of the files selected to be modified.

- **FORMAT:** **/VERSION=version** or **/VERSION=***

The value of **version** can range from 0 through -32767.

- **DEFAULT:** **/VERSION=0**
- **RESTRICTIONS:** **/VERSION** can be specified only with the following version-dependent attributes:
 - ARCHIVED
 - ATF
 - BACKUP (file version is a candidate for being archived or backed up)
 - BACKED UP (users cannot directly set this attribute)
 - RETENTION
 - REPLICATED
 - RPL_CLASS
 - VTF

/VOLUME Indicates that the command will select only files with one or more extents on the volume specified by this modifier.

- **FORMAT:**
 - **/VOLUME={media_type}{recording_type}{volume_label}:{side}**
 - **/VOLUME={media_type}{recording_type}{volume_label}:***
- **DEFAULT:** If you omit **/VOLUME**, the system does not select files by volume.

- RESTRICTION: /VOLUME and /VSET are mutually exclusive.
- /VSET** Indicates that the command will select only files in the volume set specified by this modifier.
- FORMAT: /VSET=vset_name
 - FORMAT RESTRICTIONS: Wild cards are not allowed.
 - DEFAULT: If you omit /VSET, the system does not use the volume set name to select files.
 - RESTRICTION: /VOLUME and /VSET are mutually exclusive.
- /VTF** Specifies a file version's VTF (Vulnerability Time Factor) attribute, which determines how long StorHouse can leave new extents of the file version in the performance buffer before copying them to their primary file set.
- VTF is a version-dependent attribute and has no effect on other versions of the file. The value of the /VERSION modifier determines which version is changed.
- FORMAT: /VTF=NEXT, /VTF=NOW, or /VTF=DIRECT
- If you specify /VTF=NEXT, the file is written to the performance buffer. The next time a backup occurs, the file is copied to its primary file set.
- If you specify /VTF=NOW, StorHouse copies the new version from the performance buffer to its primary file set as part of the command.
- If you specify /VTF=DIRECT, the file bypasses the performance buffer. Extents are written directly to their primary file set.
- DEFAULT: If you omit /VTF, the value of the file's VTF attribute is not changed.
 - PRIVILEGE: This modifier requires VTF privilege.

Examples

- Assume you have sufficient access to bypass specifying file and group passwords. To replace USERFILE's existing file passwords with new read, write, and delete passwords (READ, WRITE, and DELETE) or establish these passwords if USERFILE has none, enter:


```
? SET FILE USERFILE /NEWPASSWORDS=READ:WRITE:DELETE
```
- If you do not have sufficient access to bypass specifying file and group passwords and the passwords already exist, you must specify them as follows:

```
? SET FILE USERFILE /PASSWORDS=::OLDDEL  
/GROUP=USERGRP::GRPDEL  
/NEWPASSWORDS=READ:WRITE:DELETE
```

If you must specify a file password in order to gain file access, only the delete password is required. In this instance, the password is OLDDDEL. Likewise, the delete password for USERGRP is GRPDEL.

- To specify that relative version 0 (the default version) of the file USERFILE is not to be archived or backed up if an ARCHIVE or CREATE BACKUP command is executed, enter:

```
? SET FILE USERFILE /NOBACKUP
```

- To specify that all files in volume set MAY00 will be written to the performance buffer and then copied to their primary file sets the next time a backup occurs, enter:

```
? SET FILE * /VSET=MAY00 /VTF=NEXT /GROUP=* /VERSION=*
```

- To specify that version -1 of USERFILE will be assigned the STANDARD replication class and a retention period of 60 days, enter:

```
? SET FILE USERFILE /RPL_CLASS=STANDARD /RETENTION=60  
/VERSION=-1
```

SET FSET

The SET FSET command changes the attributes of a file set.

Format

SET FSET fset_name

COMMAND FORMAT SUMMARY					
COMMAND, PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
SET FSET	ALLOCATION +SYSTEM	-	-	-	-
/REPORT	-	-	-	-	-
fset_name	-	-	-	-	(Required)
/AUTO_STAGE	-	-	-	-	-
/FORCE_ RETENTION	-	-	-	-	-
/LIMIT=...	-	-	-	-	-
/RELEASE	-	-	-	-	-
/RETENTION=...	-	-	-	-	-
/RPL_CLASS=...	-	-	-	-	-
/SIZE=...	-	-	-	-	-
/UPDATE=...	-	-	-	-	-
/VSET=...	-	-	-	-	See text

Description

SET FSET changes the attributes of a file set.

Parameters

fset_name Specifies the name of the file set to be changed.

- **FORMAT:** fset_name
- **DEFAULT:** None; you must specify this parameter.

Command Modifiers

/REPORT Controls the generation of special text responses for the completion of significant actions. /REPORT instructs StorHouse to generate a text response. /NOREPORT instructs StorHouse not to generate a text response.

- FORMAT: /REPORT or /NOREPORT
- DEFAULT: /NOREPORT

Parameter Modifiers

All parameter modifiers except /VSET are mutually exclusive. You can only specify one in each command.

/AUTO_STAGE Assigns or clears the file set AUTO_STAGE attribute. The system administrator sets this attribute to enable StorHouse to automatically stage (copy) the files in the specified file set from Level L storage to the performance buffer for faster access when read from the host.

- FORMAT: /AUTO_STAGE or /NOAUTO_STAGE
- DEFAULT: No change.
- RESTRICTIONS: The file set must be a primary file set on Level L storage.
- PRIVILEGES: None.

/FORCE_RETENTION Specifies whether the file set retention value overrides the file retention value explicitly supplied by an application at file create time. This modifier affects only new files in the file set. /NOFORCE_RETENTION indicates that the file set retention value does not override the file retention value.

- FORMAT: /FORCE_RETENTION or /NOFORCE_RETENTION
- DEFAULT: If you omit /FORCE_RETENTION, there is no change.
- RESTRICTIONS: /FORCE_RETENTION can only be used when specifying a primary file set.

/LIMIT Specifies the maximum number of bytes this file set can contain. A LIMIT of 0 indicates that the maximum size is not limited.

When changing the size of the performance buffer file set, /SIZE and /LIMIT should be set to the same value. Note that this requires two SET FSET commands, and the /SIZE value cannot be greater than the /LIMIT value. When increasing /SIZE, set /LIMIT first. When decreasing /SIZE, set /SIZE first.

- FORMAT: /LIMIT=number_of_bytes or /LIMIT=0

The value for number_of_bytes can range from 0 up to 4294967295 (do not use commas in the specification). Use K to indicate 1,000-byte units, M to indicate 1,000,000-byte units, and G to indicate 1,000,000,000-byte units. If you do not specify K, M, or G, the number defaults to 1-byte units.

- **DEFAULT:** If you omit /LIMIT, the LIMIT attribute of the file set is not changed.
- **RESTRICTIONS:** If the value of /LIMIT is not zero, it must be greater than or equal to the value of the file set's /SIZE, that is, the number of bytes in the file set.

/RELEASE

Indicates that StorHouse is to deallocate free storage from the file set. StorHouse deallocates all unnecessary general-use free storage. It also deallocates all free storage reserved for updates except for any that must be retained to meet the requirements of the update attribute. If the value of the update attribute is zero, StorHouse releases all unnecessary free storage.

The system returns the deallocated storage to the volume set's free storage. If the system deallocates all storage allocated to the file set (none was used, so the size becomes zero), the system removes the file set.

- **FORMAT:** /RELEASE
- **DEFAULT:** If you omit /RELEASE, other modifiers control any changes to file set storage.
- **RESTRICTIONS:**
 - You cannot release the performance buffer FSET (VSET=MAGDISK, FSET=\$\$BUFFER)
 - You cannot release the checkpoint file set if the checkpointing feature is enabled (in other words, if CHKP_ON is set to TRUE).
 - SET FSET/RELEASE does remove file sets with non-zero retention values when the RETENTION_MODE system parameter is set to STRICT.

/RETENTION

Specifies the file set retention attribute (retention period). The file set retention attribute applies only to new files (in the file set) that are created without a specified retention value, or whenever /FORCE_RETENTION is in effect on the file set.

- **FORMAT:**

Option	Description
/RETENTION=DEFAULT	Sets the retention period to the default value.
/RETENTION=number_of_days	<p>Sets the retention period to the specified number of days. The retention period ends when the current date is beyond the file's last_modified date plus the specified retention value.</p> <p>A value of 0 indicates no retention period (same as specifying ZERO).</p> <p>Example: /RETENTION=3</p> <p>In this example, the retention period is 3 days. The retention period for a file that was last modified at 11 p.m. on December 12 expires at 11 p.m. on December 15.</p>
/RETENTION=ZERO	Sets no retention period.
/RETENTION=FOREVER	Sets an infinite retention period.

- **DEFAULT:** If you omit /RETENTION, the default is to keep the current file set retention attribute.
- **RESTRICTIONS:**
 - The RETENTION_MODE system parameter determines which retention settings are valid. If the RETENTION_MODE is STRICT, you may set /RETENTION only to a more restrictive value than its current setting (for example, from a current retention of 100 days to a new retention of 200 days). If the RETENTION_MODE is BASIC, you may set /RETENTION to any valid value (for example, from a current retention of 60 days to a retention of 30 days or 90 days).
 - When you specify /RETENTION, the specified file set must be a primary file set.

/RPL_CLASS Specifies the name of the replication class for new files written to this file set. A replication class is a collection of replication-related information (network device identification and system, file set, volume set, and link names) about the target StorHouse system. If the specified replication class does not exist, StorHouse generates a warning message.

- **FORMAT:** /RPL_CLASS=replication_class_name or /NORPL_CLASS

A replication_class_name can consist of from 1 to 8 of the following ASCII characters: A-Z, 0-9, _, and \$. StorHouse always forces replication class names to uppercase, even when enclosed in quotes.

- **DEFAULT:** If you omit /RPL_CLASS, the default is no change.
- **RESTRICTIONS:** When you specify /RPL_CLASS, the specified file set must be a primary file set.

/SIZE Specifies the number of bytes in the file set.

When changing the size of the performance buffer file set, /SIZE and /LIMIT should be set to the same value. Note that this requires two SET FSET commands, and the /SIZE value cannot be greater than the /LIMIT value. When increasing /SIZE, set /LIMIT first. When decreasing /SIZE, set /SIZE first.

- **FORMAT:** /SIZE=number_of_bytes or /SIZE=0

The value for number_of_bytes can range from 0 up to 4294967295 (do not use commas in the specification). Use K to indicate 1,000-byte units, M to indicate 1,000,000-byte units, and G to indicate 1,000,000,000-byte units. If you do not specify K, M, or G, the number defaults to 1-byte units.

- If the value of /SIZE is larger than the current size of the file set, the system allocates space to the set until the desired size is reached.
- If the value of /SIZE is smaller than the current size of the file set, the system deallocates all unnecessary free space from the file set.
- If the specified size is less than the size already allocated to files, the system rejects the request.
- If the file set is contiguous, the system allocates entire volume sides unless the file set's LIMIT attribute restricts an allocation to less than a full side. For noncontiguous file sets, the system tries to extend to contiguous space (on the same volume side) if it is available, but any available space can be used if contiguous space is not available.

If level F consists of only one volume, the LIMIT of a contiguous level F file set must be smaller than the size of one volume side. Otherwise, the level F file set must be noncontiguous. This restriction does not apply to file sets on optical storage.

- **DEFAULT:** If you omit /SIZE, the size attribute is not changed, or changes are controlled by other modifiers.
- **RESTRICTIONS:** The value of /SIZE must be less than or equal to the value of the limit attribute for the file set.

/UPDATE Specifies a percentage of the file set size that is to be set aside for VRAM file updates. The percentage can range from 0 to 99.

- **FORMAT:** /UPDATE=percentage
 - If the **percentage** value is larger than the previous value, the file set may not have enough free space on one or more volume sides to meet the requirement. In these cases, the system sets aside what space there is and ignores the deficiencies.
 - If the size of the updates written already exceeds the amount to be set aside, the system reserves no additional space.
 - If the **percentage** value is smaller than the previous value, the system frees for general use any excess unused space reserved for updates.
- **DEFAULT:** If you omit /UPDATE, the system does not change the update percentage.

/VSET Specifies the volume set on which the file set exists.

- **FORMAT:** /VSET=vset_name
- **FORMAT RESTRICTIONS:** Wild cards are not allowed.
- **DEFAULT:** If you omit /VSET or specify /VSET without the vset_name, the system uses the default volume set name from your account.

Examples

- To release all unused partitions (in other words, the part of the file set on a volume side) if USERFSET is a contiguous file set, or to release all free storage from the file set if USERFSET is a noncontiguous file set, enter:

```
? SET FSET USERFSET /RELEASE
```

- Assume the following:
 - USERFSET is a contiguous file set with a current allocation of one entire volume side (3500 MB).
 - USERFSET has a LIMIT of zero, which indicates no limit.
 - USERFSET belongs to the volume set USERVSET, which has a current allocation of three additional empty sides (3500 MB per side).

To cause StorHouse to allocate all free space on one entire volume side to USERFSET, enter:

```
? SET FSET USERFSET /VSET=USERVSET /SIZE=4000MB
```

- To assign the AUTO_STAGE attribute to the FSET00 file set, enter:

? SET FSET FSET00 /AUTO_STAGE

SHOW FILE

The SHOW FILE command displays information for a file version.

Format

SHOW FILE filename

COMMAND FORMAT SUMMARY					
COMMAND PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
SHOW FILE	SHOW or FILE	-	-	-	-
/BRIEF	-	-	-	-	-
/EXTENT	-	-	-	-	-
/FULL	-	-	-	-	-
/NAME	-	-	-	-	-
filename	-	R	R	-	(Required)
/ARCHIVE_EXISTS	-	-	-	-	-
/BACKUP_EXISTS	-	-	-	-	-
/BEFORE=...	-	-	-	-	-
/BKP_ATTR	-	-	-	-	-
/BUFFERED	-	-	-	-	-
/DAMAGED	-	-	-	-	-
/DELETED	-	-	-	-	-
/DIRECTORY=...	-	-	-	-	/DIRECTORY= PRIMARY
/FSET=...	-	-	-	-	-
/GROUP=...	SETGROUP	R	-	-	Current default
/LEVEL=...	-	-	-	-	-
/ORDER=...	-	-	-	-	/ORDER= NATURAL
/PASSWORDS=...	-	-	-	-	-
/PRIMARY_EXISTS	-	-	-	-	-
/RESIDENT	-	-	-	-	-
/SAFE_COPIES=...	-	-	-	-	-
/SINCE=...	-	-	-	-	-
/UNUSED=...	-	-	-	-	-
/VERSION=...	-	-	-	-	/VERSION=0
/VOLUME=...	-	-	-	-	-
/VSET=...	-	-	-	-	-

Description

SHOW FILE produces one display record for each selected file version. The default display record contains the following information:

- FILE=file_name
- GROUP=group_name
- VERSION=relative_version_number
- FID=system_identifier.file_number

The /BRIEF, /EXTENT, and /FULL modifiers add more information to the default display.

The default display lists records alphabetically by group name and file name, and then from most recent to oldest version (if more than one version is displayed). If /EXTENT is specified, the command displays extents in highest to lowest extent sequence number order. You can override the default display order by specifying /ORDER=ANY.

At the end of the SHOW FILE information, StorHouse lists the total number of displayed files in the format:

Total files displayed=nnnn

where nnnn is expressed in decimal digits.

Selecting files for display. You can use the following parameter modifiers to select files for display:

/ARCHIVE_EXISTS	/DIRECTORY	/SINCE
/BACKUP_EXISTS	/FSET	/UNUSED
/BEFORE	/GROUP	/VERSION
/BKP_ATTR	/LEVEL	/VOLUME
/BUFFERED	/PRIMARY_EXISTS	/VSET
/DAMAGED	/RESIDENT	
/DELETED	/SAFE_COPIES	

If you specify a value for one of the modifiers in the preceding list, SHOW FILE selects only files matching that value. If you specify multiple modifiers, SHOW FILE selects only files matching all specified values.

Some modifiers are mutually exclusive (that is, you cannot specify them together in a single command).

Parameters

filename Specifies the StorHouse file for which information will be displayed.

- **FORMAT:** filename

You can use multiple wildcards anywhere in the filename specification.

- **DEFAULT:** None; you must specify this parameter.
- **ACCESS REQUIREMENTS:** Read access to the file and group.

Command Modifiers

/BRIEF Displays the following fields for each file version:

- **FILE=**file_name
- **GROUP=**group_name
- **VERSION=**relative_version_number
- **FID=**system_identifier.file_number
- **DATE=**creation_date (always for the primary)
- **SIZE=**number_of_bytes
- **FORMAT:** /BRIEF
- **DEFAULT:** If you omit /BRIEF, the display is controlled by other modifiers.
- **RESTRICTIONS:** /FULL overrides /BRIEF.

/EXTENT Displays the following extent information for each selected extent of the specified file version.

Extent Field	Definition
EXTENT_NUMBER	Extent sequence number.
EXTENT_SID	Extent system identifier.
EXTENT_DATE	Date and time the extent was first created.
EXTENT_WRITTEN	Date and time the extent was written to its current location. For example, if StorHouse initially creates an extent in the performance buffer, the extent's EXTENT_DATE and EXTENT_WRITTEN are the same. If StorHouse subsequently writes back the extent to level L, EXTENT_DATE remains the same, but EXTENT_WRITTEN changes to the time the new copy was written to level L.
EXTENT_REVISION	Revision number of the file version for which the extent was created. An extent can be used in more than one revision.
EXTENT_SIZE	Size of the extent in bytes.

Extent Field	Definition
EXTENT_LOCATION	VID of the volume where the extent resides.
EXTENT_LEVEL	Storage level that contains the resident file set for the extent. Valid values are F, L, and S. (A value of F indicates a level F file set, not the performance buffer.)
EXTENT_STATUS	Status of the extent. Valid values are: <ul style="list-style-type: none"> • BUFFERED - indicates there is a copy of the extent in the performance buffer. • LAST - indicates the extent is the last extent of a revision or checkpoint. • NEW - indicates the extent has not been copied from the performance buffer to its resident file set. • NONE - indicates none of the other statuses. • UPDATE - indicates the extent was created in a VRAM update operation. • WRITEBACK_DISABLED - indicates the extent cannot be backed up to its destination file set.
EXTENT_MF	Migration factor of the extent. This field is valid only for extents in the performance buffer. For non-performance buffer extents, EXTENT_MF is displayed as "none."
EXTENT_RETENTION_DATE	Date until which the system will attempt to retain the extent in the performance buffer. <ul style="list-style-type: none"> • A value of "none" indicates no retention date. • A value of "expired" indicates that the retention period has expired.

You can use /EXTENT with the default display, with /BRIEF, or with /FULL.

At the end of the display record, /EXTENT adds the following:

EXTENTS_DISPLAYED=number_of_extents_displayed

- **FORMAT:** /EXTENT
- **DEFAULT:** If you omit /EXTENT, the command does not add extent information.

/FULL Adds the following fields to the default display for each file version:

- **REVISION**=highest_revision_number.
- **DIRECT_REC**=directory_record_number in the directory used by SHOW FILE. The value of DIRECT_REC can range from 2 through 4294967295. The value for any one file version may differ for copies of the version in other directories.
- **LASTACCESS**=last_access_date/time the file version was read or written.

- MODIFIED_DIR=last_modification_date/time of the directory entry for the specified file (for example, when a file was relocated).
- MODIFIED_FILE=last_modification_date/time of the file's data (for example, the creation time of the last extent).
- BACKUP=backup_date or "none".
- LIMIT=limit_attribute (always 65536 for archive and backup files).
- PROTECTION=protection_indicators.
- VTF=vtf_attribute.
- ATF=atf_attribute.
- ORGANIZATION=organization_code. Valid values are:
 - KEYED
 - KEYSEQUENTIAL
 - RECORD
 - SEQUENTIAL
 - STORHOUSE
 - VRAM (RECORD, KEYSEQUENTIAL, or KEYED file created before Release 3.0).
- FILE_SYSTEM=file_system_code.
- HOST_TYPE=host_type_code.
- FRAME_VER=frame_structure_version.
- FRAME_SIZE=bytes_per_frame.
- FRAME_HDR=frame_header_size (in bytes).
- RECORD_HDR=record_header_size.
- HOST_UNIT=host_data_unit_size.
- MAX_LENGTH=maximum_record_size.
- RECORD_COUNT=number_of_records (number of user records written; meaningful only for VRAM files).
- ATTRIBUTES=attribute_code. Valid values are:
 - 20 - Print format (machine carriage control).
 - 40 - Print format (ANSI carriage control).

- 80 - Fixed record length).
- ACCESES=number_of_accesses.
- EDC=edc_code.
- STATUS=status. Valid values are:
 - COMPLETE
 - PARTIAL
 - TRUNCATED.
- DESCRIPTOR=flags. Valid values are:
 - ARCHIVED – current revision was archived.
 - BACKED_UP – current revision was backed up.
 - CATALOGING – being cataloged.
 - DELETED – deleted.
 - HARD_DISABLED – hardware disabled.
 - NAMED – named file.
 - NOBACKUP – do not back up file version.
 - PENDING – file copy/transfer in progress.
 - SOFT_DISABLED – software disabled.
 - REPLICATED – a replica of the current file version exists on another StorHouse system.
- VSET=vset_name.
- FSET=fset_name.
- RETENTION=retention_attribute.
- RPL_CLASS=rpl_class_name.
- EXTENT_COUNT=number_of_extents (in version).

StorHouse identifies file and data record formats using the file type, the file system type, the host type, and the file's organization. See the *StorHouse Concepts and Facilities Manual* for definitions of these fields.

- FORMAT: /FULL
- DEFAULT: If you omit /FULL, the display is controlled by other modifiers.
- RESTRICTIONS: /FULL overrides /BRIEF.

/NAME Displays the file name in the format:

“filename” /GROUP=group_name /VERSION=version_number

The output of a SHOW FILE/NAME can be directed to a file and subsequently edited and used in other file-related commands such as DELETE.

- **FORMAT:** /NAME
- **DEFAULT:** If you omit /NAME, the command does not display the file name in the specified format.
- **RESTRICTION:** /NAME is mutually exclusive with /BRIEF, /EXTENT, and /FULL.

Parameter Modifiers

/ARCHIVE_EXISTS Selects files based on whether they have an archive copy. /NOARCHIVE_EXISTS selects files that do not have an archive copy.

- **FORMAT:**
 - /ARCHIVE_EXISTS
 - /NOARCHIVE_EXISTS
- **DEFAULT:** If you omit /ARCHIVE_EXISTS, the system does not select files based on whether an archive copy exists.
- **RESTRICTION:** Do not use /ARCHIVE_EXISTS when displaying information for an archive file.

/BKP_ATTR Selects files based on whether their BACKUP attribute is set. /NOBKP_ATTR selects files that do not have their BACKUP attribute set.

- **FORMAT:**
 - /BKP_ATTR
 - /NOBKP_ATTR
- **DEFAULT:** If you omit /BKP_ATTR, the system does not select files based on whether the BACKUP attribute is set.
- **RESTRICTION:** Do not use /BKP_ATTR when displaying information for a backup file.

/BACKUP_EXISTS Selects files based on whether they have a backup copy. /NOBACKUP_EXISTS selects files that do not have a backup copy.

- **FORMAT:**
 - /BACKUP_EXISTS

- /NOBACKUP_EXISTS
 - DEFAULT: If you omit /BACKUP_EXISTS, the system does not select files based on whether a backup copy exists.
 - RESTRICTION: Do not use /BACKUP_EXISTS when displaying information for a backup file.
- /BEFORE** Selects file versions with a creation date and time value that is the same as or earlier than a specified date and time. The version creation date and time value is the date and time the primary version was first created in StorHouse.
- FORMAT: /BEFORE=absolute_time or /BEFORE=delta_time
- For a full description of the absolute time and delta time data fields, see “Appendix B” of the *Command Language Reference Manual*.
- DEFAULT: If you omit /BEFORE and /SINCE, the system does not select files based on creation date and time.
- /BUFFERED** Selects files for display based on whether copies of all file extents reside in the performance buffer. /NOBUFFERED selects files that do not have copies of all extents in the performance buffer.
- FORMAT:
 - /BUFFERED
 - /NOBUFFERED
 - DEFAULT: If you omit /BUFFERED, the command does not select files based on whether they have copies of all extents in the performance buffer.
 - RESTRICTIONS: /BUFFERED is meaningful only for files in the PRIMARY directory.
- /DAMAGED** Selects files for display based on whether they are damaged. Damaged files are files marked PENDING, SOFT_DISABLED, or HARD_DISABLED, or that are missing one or more extents, that is, marked as truncated or partial. /DAMAGED checks for problems with the file’s directory information, not for problems with the physical media where the file resides.
- FORMAT:
 - /DAMAGED
 - /NODAMAGED
 - DEFAULT: If you omit /DAMAGED, the command does not select files based on whether they are damaged.

/DELETED Displays a list of files that are deleted, but not yet removed. If multiple deleted versions of the file exist, the most recently deleted version is version 0, and the previously deleted version is version -1, and so on.

Note: Deleted file version numbers are not related to the version numbers assigned to files in their original directory. Deleted file versions correspond to the order that files were deleted rather than the order that they were created.

- **FORMAT:** /DELETED
- **DEFAULT:** If you omit /DELETED, the command does not select deleted files.
- **RESTRICTIONS:** /DELETED is mutually exclusive with /PASSWORDS.

/DIRECTORY Specifies the directory used to select files for display. If you specify /DIRECTORY and /LEVEL, the command only selects files that are listed in the specified directory and that reside on the specified storage level.

- **FORMAT:**
 - /DIRECTORY=ARCHIVE
 - /DIRECTORY=BACKUP
 - /DIRECTORY=PRIMARY
- **DEFAULT:** /DIRECTORY=PRIMARY (only if you do not specify /VOLUME and /VSET)
- **RESTRICTIONS:** /DIRECTORY is mutually exclusive with /VOLUME and /VSET.

/FSET Selects for display file versions located in the specified file set.

- **FORMAT:** /FSET=fset_name
- **FORMAT RESTRICTIONS:** Wild cards are not allowed in the file set name.
- **DEFAULT:** If you omit /FSET, the system does not use the file set name to select files.
- **RESTRICTIONS:**
 - If you specify /FSET, you must also specify /VSET.
 - The performance buffer file set may not be specified.

/GROUP Specifies a file access group name and, optionally, the group's read password.

- **FORMAT:**

- /GROUP=groupname<:readpw>
 - /GROUP=partial_groupname*
 - /GROUP=*
- DEFAULT: If you omit /GROUP, the default is your default file access group.
 - ACCESS REQUIREMENTS: Read access to the group.

You must specify the read password unless:

- The group is not protected by a read password.
 - Your privilege bypasses read access password checks.
 - Your default access to the group includes read access.
- PRIVILEGE: You must have SETGROUP privilege to specify any group except your default group.

/LEVEL Selects for display files whose resident file sets are located on the specified storage level.

SHOW FILE /LEVEL bases selection on a file's destination, which is not necessarily its current physical location. For files that have already been backed up, SHOW FILE displays the level where the volume containing the resident copy of the file is located.

For files that are still in the performance buffer, SHOW FILE bases selection on the current level of the volume that will contain the resident copy of the file after a backup.

- FORMAT:
 - /LEVEL=F for fixed storage
 - /LEVEL=L for library storage
 - /LEVEL=S for shelf storage
- DEFAULT: If you omit /LEVEL, the system does not use the level for selection.
- RESTRICTIONS: /LEVEL is mutually exclusive with /VOLUME and /VSET.

/ORDER Specifies the order in which files are displayed.

- FORMAT: /ORDER=NATURAL or /ORDER=ANY

NATURAL indicates that files are displayed in their natural order, which is alphabetically by group name and file name, and most recent version to oldest version.

ANY indicates that you do not have an order preference; as a result, the system displays the files in any order.

With /ORDER=ANY, if you specify a wildcard for the filename (either full or partial) and a value for /VOLUME or /VSET, the system will use an algorithm to produce the list of files faster than if it had to produce the list in natural order. However, if the wildcard specifies a very restrictive search, the system may produce the list of files faster if you specify /ORDER=NATURAL (the default) rather than /ORDER=ANY.

- DEFAULT: /ORDER=NATURAL

/PASSWORDS Specifies the file's read password.

- FORMAT: /PASSWORDS=readpw

You must specify a read password for each file in the command unless:

- The file is not protected by read passwords.
- Your privilege bypasses read access password checks.

File passwords are not allowed if filename or /GROUP includes a wild card.

- DEFAULT: If you omit /PASSWORDS, the file password defaults to null unless bypassed by a privilege.
- RESTRICTIONS: /PASSWORDS is mutually exclusive with /DELETED.

/PRIMARY_EXISTS Selects files based on whether they have a primary copy. /NOPRIMARY_EXISTS selects files that do not have a primary copy.

- FORMAT:
 - /PRIMARY_EXISTS
 - /NOPRIMARY_EXISTS
- DEFAULT: If you omit /PRIMARY_EXISTS, the system does not select files based on whether a primary copy exists.
- RESTRICTION: This modifier is meaningful only when you specify /DIRECTORY=ARCHIVE or /DIRECTORY=BACKUP.

/RESIDENT Selects files for display based on whether all extents have been written to their resident file sets. /NORESIDENT selects files whose extents have not all been written to their resident file sets.

- FORMAT:
 - /RESIDENT
 - /NORESIDENT
- DEFAULT: If you omit /RESIDENT, the command does not select files based on whether all extents have been written to their resident file sets.

- **RESTRICTIONS:** /RESIDENT is meaningful only for files in the PRIMARY directory.

/SAFE_COPIES Selects files for display with at least the specified number of safe copies in the PRIMARY, BACKUP, and ARCHIVE directories. For a file to be considered safe, it must be complete (that is, no missing extents) and usable (that is, not marked PENDING, SOFT_DISABLED, or HARD_DISABLED). In addition, for files in the PRIMARY directory, all extents must have been written to their resident file sets.

- **FORMAT:** /SAFE_COPIES=number_of_copies

The number_of_copies can range from 1 through 3.

- **DEFAULT:** If you omit /SAFE_COPIES, the system does not select files based on the number of safe copies that exist.

/SINCE Selects file versions with a creation date and time value that is the same as or later than a specified date and time. The version creation date and time are the date and time the primary version was first created in StorHouse.

- **FORMAT:** /SINCE=absolute_time or /SINCE=delta_time

For a full description of the absolute time and delta time data fields, see “Appendix B” of the *Command Language Reference Manual*.

- **DEFAULT:** If you omit /SINCE and /BEFORE, the system does not select files based on creation date and time.

/UNUSED Selects for display files that have not been accessed since the specified time.

- **FORMAT:** /UNUSED=absolute_time or /UNUSED=delta_time

Absolute time specifies a calendar date and clock time. For a full description of the absolute time and delta time data fields, see “Appendix B” of the *Command Language Reference Manual*.

- **DEFAULT:** If you omit /UNUSED, the default is to select files with any last-access date.

/VERSION Selects the specified file versions for display. The versions are relative version numbers in the directory selected by /DIRECTORY or, if you omit /DIRECTORY, in the primary directory.

- **FORMAT:** /VERSION=version or /VERSION=*

The value of version can range from 0 through -32767 for primary files. For files in the archive and backup directories, the range is 0 through -65535.

- **DEFAULT:** If you omit /VERSION, the default is /VERSION=0.

/VOLUME Selects for display files whose resident file sets are located on the specified volume.

SHOW FILE /VOLUME bases selection on a file's destination, which is not necessarily its current location.

With /ORDER=ANY, if you specify a wildcard for the filename (either full or partial) and a value for /VOLUME, the system will use an algorithm to produce the list of files faster than if it had to produce the list in natural order. However, if the wildcard specifies a very restrictive search, the system may produce the list of files faster if you specify /ORDER=NATURAL (the default) rather than /ORDER=ANY.

- **FORMAT:**

`/VOLUME={media_type}{recording_type}{volume_label}:{side}`

The braces “{ }” are not part of the specification.

- **FORMAT RESTRICTIONS:** The wild card is not allowed.
- **DEFAULT:** If you omit /VOLUME, the system does not select files by volume.
- **RESTRICTIONS:** /VOLUME is mutually exclusive with /DIRECTORY, /VSET, and /LEVEL.

/VSET Selects for display files that reside on the specified volume set.

With /ORDER=ANY, if you specify a wildcard for the filename (either full or partial) and a value for /VSET, the system will use an algorithm to produce the list of files faster than if it had to produce the list in natural order. However, if the wildcard specifies a very restrictive search, the system may produce the list of files faster if you specify /ORDER=NATURAL (the default) rather than /ORDER=ANY.

- **FORMAT:** /VSET=vset_name
- **FORMAT RESTRICTIONS:** Wild cards are not allowed in the volume set name.
- **DEFAULT:** If you do not specify /VSET, the system does not use the vset_name to select files.
- **RESTRICTIONS:** /VSET is mutually exclusive with /DIRECTORY, /VOLUME and /LEVEL.

Examples

- To display directory information for the latest version of all files in your default file access group to which you have read access, enter:

? SHOW FILE *

The default display includes the file name, group name, relative version number, and fid.

```
FILE="EXTKEY" GROUP=USERGRP VERSION=0 FID=234.118
FILE="LOGFILE" GROUP=USERGRP VERSION=0 FID=234.8
FILE="USERFILE" GROUP=USERGRP VERSION=0 FID=234.201
```

Total files displayed=3

- To display all information available for version 0 of the file SAMPLE, enter:

```
? SHOW FILE SAMPLE /FULL
```

The system displays the following information:

```
FILE="SAMPLE" GROUP=SERVICE VERSION=0 FID=4444.2 DATE=26-
MAR-2004:14:05:06 SIZE=401 REVISION=1 DIRECT_REC=2
LASTACCESS=26-MAR-2004:14:05:06 MODIFIED_DIR=26-MAR-
2004:14:05:29 MODIFIED_FILE=26-MAR-2004:14:05:06 BACKUP=none
LIMIT=128 PROTECTION=none VTF=NEXT ATF=3
ORGANIZATION=SEQUENTIAL FILE_SYSTEM=67 HOST_TYPE=33
FRAME_VER=1 FRAME_SIZE=31744 FRAME_HDR=20 RECORD_HDR=5
HOST_UNIT=8 MAX_LENGTH=16384 RECORD_COUNT=0
ATTRIBUTES=0 ACCESSES=1 EDC=2 STATUS=COMPLETE
DESCRIPTOR=(NAMED, REPLICATED) VSET=P FSET=P
RETENTION=365 RPL_CLASS=STANDARD EXTENT_COUNT=1
```

Total files displayed=1

Because you omitted the /GROUP modifier on the command, USERFILE must be located in your default access group, or the command returns an error message.

- To display the file name, access group name, relative version number, fid, creation date, size, and last access date for all files in the access group USERGRP that were created before midnight of March 31, 2000, enter:

```
? SHOW FILE * /GROUP=USERGRP /BEFORE=31-MAR-2000 /BRIEF
```

The system displays the following information:

```
FILE="EXTKEY" GROUP=USERGRP VERSION=0 FID=234.105
DATE=29-MAR-2000:19:03:25 SIZE=398

FILE="USERFILE" GROUP=USERGRP VERSION=0 FID=234.2
DATE=28-MAR-2000:18:43:45 SIZE=1310
```

Total files displayed=2

- To display the file name, access group name, relative version number, fid, creation date, and size for the current version of all files in the access group USERGRP that were created before two days ago (delta time), enter:

```
? SHOW FILE * /GROUP=USERGRP /BEFORE=D2- /BRIEF
```

- To display all information available for version 0 of the file VRAMFILE, including file extent information, enter:

```
? SHOW FILE VRAMFILE /EXTENT /FULL
```

The system displays the following information:

```
FILE="VRAMFILE" GROUP=SERVICE VERSION=0 FID=234.8
DATE=16-NOV-1999:14:38:59 SIZE=75282 REVISION=2 DIRECT_REC=9
LASTACCESS=16-NOV-1999:14:47:14 MODIFIED_DIR=16-NOV-
1999:14:47:14 MODIFIED_FILE=16-NOV-1999:14:42:20 BACKUP=none
LIMIT=128 PROTECTION=none VTF=DIRECT ATF=3
ORGANIZATION=KEYSEQUENTIAL FILE_SYSTEM=66 HOST_TYPE=17
FRAME_VER=1 FRAME_SIZE=31744 FRAME_HDR=20 RECORD_HDR=5
HOST_UNIT=8 MAX_LENGTH=80 RECORD_COUNT=500 ATTRIBUTES=0
ACCESSES=7 EDC=2 STATUS=COMPLETE DESCRIPTOR=(NAMED,
NOBACKUP, REPLICATED) VSET=SYSTEM FSET=SERVICE
RETENTION=365 RPL_CLASS=SMBACKUP EXTENT_COUNT=3
```

```
EXTENT_NUMBER=1000002 EXTENT_SID=234
EXTENT_DATE=16-NOV-1999:14:41:48
EXTENT_WRITTEN=16-NOV-1999:14:42:19 EXTENT_REVISION=2
EXTENT_SIZE=5154 EXTENT_LOCATION=OAD"2523A699":A
EXTENT_LEVEL=L EXTENT_STATUS=(LAST) EXTENT_MF=none
EXTENT_RETENTION_DATE=16-NOV-2000:11:59:59
```

```
EXTENT_NUMBER=1000001 EXTENT_SID=234
EXTENT_DATE=16-NOV-1999:14:41:40
EXTENT_WRITTEN=16-NOV-1999:14:41:46 EXTENT_REVISION=2
EXTENT_SIZE=67022 EXTENT_LOCATION=OAD"2523A699":A
EXTENT_LEVEL=L EXTENT_STATUS=(none) EXTENT_MF=none
EXTENT_RETENTION_DATE=16-NOV-2000:11:59:59
```

```
EXTENT_NUMBER=1000000 EXTENT_SID=234
EXTENT_DATE=16-NOV-1999:14:39:01
EXTENT_WRITTEN=16-NOV-1999:14:39:02 EXTENT_REVISION=1
EXTENT_SIZE=3106 EXTENT_LOCATION=OAD"2523A648":A
EXTENT_LEVEL=L EXTENT_STATUS=(LAST) EXTENT_MF=none
EXTENT_RETENTION_DATE=16-NOV-2000:11:59:59
```

```
EXTENTS_DISPLAYED=3
```

```
Total files displayed=1
```

- To display version 0 of all deleted files in the current file access group, enter:

? SHOW FILE * /DELETED

The system displays the following information:

```
FILE="CHCONFIG" GROUP=SERVICE VERSION=0 FID=100.5
FILE="NETCONFIG" GROUP=SERVICE VERSION=0 FID=100.1
FILE="SMCONFIG" GROUP=SERVICE VERSION=0 FID=100.4
```

- To display all files in the primary directory and the current file access group that begin with "AMMO" and whose extensions begin with "A", enter:

? SHOW FILE AMMO*.A*

```
FILE="AMMO.ABEND" GROUP=TEST VERSION=0 FID=4444.115345
```

```
FILE="AMMO30.AMMXB268" GROUP=TEST VERSION=0
FID=2062.2005387
```

```
FILE="AMMO30.AMMCS860" GROUP=TEST VERSION=0
FID=2062.1003465
```

```
FILE="AMMO30.AMMCS861" GROUP=TEST VERSION=0
FID=2062.1003466
```

```
FILE="AMMO30.AMMCS880" GROUP=TEST VERSION=0
FID=2062.1151369
```

```
FILE="AMMOII.ABENDOC4" GROUP=TEST VERSION=0 FID=4444.115352
```

Total files displayed=6

- To display information for the latest version of all files in your default file access group for which you have read access whose names contain the numbers "99", enter:

? SHOW FILE *99*

The default display for each file includes the file name, group name, relative version number, and fid.

```
FILE="99FILE1" GROUP=USERGRP VERSION=0 FID=234.201
FILE="EXT99" GROUP=USERGRP VERSION=0 FID=234.118
FILE="LOG99" GROUP=USERGRP VERSION=0 FID=234.8
```

Total Files Displayed=3

- To display the files in your default file access group that have archive copies, enter:

? SHOW FILE * /ARCHIVE_EXISTS

SHOW FSET

The SHOW FSET command displays information about a file set.

Format

SHOW FSET fset_name

COMMAND FORMAT SUMMARY					
COMMAND PARAMETER, OR MODIFIER	REQUIRED COMMAND PRIVILEGE	REQUIRED GROUP ACCESS	REQUIRED FILE ACCESS	MINIMUM ACCOUNT ACCESS	DEFAULT
SHOW FSET	SHOW or ALLOCATION or SYSTEM	-	-	-	-
fset_name	-	-	-	-	Current default
/AUTO_STAGE					See text
/FULL	-	-	-	-	-
/VSET=...	-	-	-	-	Current default

Description

SHOW FSET displays file set information. If you do not specify any modifiers, the system displays the following information:

- FSET=fset_name indicates the name of the file set.
- VSET=vset_name indicates the name of the volume set.
- DIRECTORY=directory_name indicates the name of the directory to which the volume set belongs.
- SIZE=size displays the total size of the file set.
- GENERAL_FREE=size displays the number of bytes available for allocation to files. (This does not include UPDATE_FREE storage.)
- UPDATE_FREE=size displays the number of bytes set aside for allocation to file updates.
- STATE=(CONTIGUOUS, AUTO_STAGE, FORCE_RETENTION) indicates that the file set has the contiguous storage allocation attribute, the AUTO_STAGE attribute, the FORCE_RETENTION attribute, or some combination of the three. If a file set has no assigned attributes, STATE does not display.

- **RETENTION**=(number_of_days, DEFAULT, FOREVER) indicates the file set retention attribute. The number of days can range from 0-65000. StorHouse displays RETENTION only for primary file sets.
- **RPL_CLASS**=(rpl_class_name,none) indicates the name of the replication class assigned to the file set or none if the file set does not have a replication class. StorHouse displays this field only for primary file sets.

Sizes are displayed in 1000-byte units and followed by the letters KB. If a size is not a multiple of 1000 bytes, the system rounds it down for display.

Parameters

fset_name Specifies the name of the file set for which information is to be displayed.

- **FORMAT:**
 - fset_name
 - partial_fset_name*
 - *
- **DEFAULT:** If you omit this parameter, the default is your default file set name.

Parameter Modifiers

/AUTO_STAGE Displays file sets with or without the AUTO_STAGE attribute.

- **FORMAT:** /AUTO_STAGE or /NOAUTO_STAGE
- **DEFAULT:** The default is to display only those file sets with /NOAUTO_STAGE set.
- **RESTRICTIONS:** None.
- **PRIVILEGES:** None.

/FULL Displays all available information for a file set.

/FULL displays the same information as the default display, plus the following:

- **GENERAL_ALLOCATED=size** displays the number of bytes allocated to files for general usage. (This does not include UPDATE_ALLOCATED storage.)
- **UPDATE_ALLOCATED=size** displays the number of bytes allocated to file updates.
- **UPDATE_PERCENT=percentage** displays the value of the update attribute for the file set.

- LIMIT=size displays the value of the limit attribute for the file set.
- CREATED=date_time displays the absolute date and time the file set was created.
- MODIFIED=date_time displays the absolute date and time the file set was last modified.

Sizes are displayed in 1000-byte units and followed by the letters KB. If a size is not a multiple of 1000 bytes, the system rounds it down for display.

- FORMAT: /FULL
- DEFAULT: If you omit /FULL, the command displays a basic set of information described at the beginning of the SHOW FSET section.

/VSET Displays file sets located in the specified volume set.

- FORMAT:
 - /VSET=vset_name
 - /VSET=partial_vset_name*
 - /VSET=*
- DEFAULT: If you omit /VSET, the default is your default volume set.

Examples

- To display basic information for file set P in volume set P, enter:

```
? SHOW FSET P /VSET=P
```

```
FSET=P VSET=P DIRECTORY=PRIMARY SIZE=10444KB
GENERAL_FREE=10433KB UPDATE_FREE=0KB STATE=(CONTIGUOUS,
FORCE_RETENTION) RETENTION=365 RPL_CLASS=STANDARD
```

```
Total fsets displayed=1
```

- To show all information for file set P in volume set P, enter:

```
? SHOW FSET P /VSET=P /FULL
```

```
FSET=P VSET=P DIRECTORY=PRIMARY SIZE=10444KB
GENERAL_FREE=10433KB UPDATE_FREE=0KB
GENERAL_ALLOCATED=11KB UPDATE_ALLOCATED=0KB
UPDATE_PERCENT=00 LIMIT=0KB CREATED=26-MAR-2004:13:59:59
MODIFIED=26-MAR-2004:14:05:57 STATE=(CONTIGUOUS,
FORCE_RETENTION) RETENTION=365 RPL_CLASS=STANDARD
```

```
Total fsets displayed=1
```

- To display all file sets in your default volume set with the AUTO_STAGE attribute:

```
? SHOW FSET * /AUTO_STAGE
```

The system displays the following information:

```
FSET=USERFSET VSET=USERVSET DIRECTORY=PRIMARY  
SIZE=1853KB GENERAL_FREE=2KB UPDATE_FREE=0KB  
STATE=(AUTO_STAGE,REPLICATED) RETENTION=FOREVER  
/RPL_CLASS=STANDARD
```

Total fsets displayed=1

Generic Callable Interface

This section provides the changes to the LSMCO and LSMOS Generic Callable Interface Functions.

LSMCO - Create Open

LSMCO creates a new VRAM file on StorHouse and establishes a data transfer link for writing information to that file. LSMCO is equivalent to issuing a StorHouse Command Language CREATE FILE command followed by LSMOV in mode APPEND.

LSMCO requires RECORD privilege to specify the function. For more information about StorHouse privileges, refer to the *Command Language Reference Manual* in the StorHouse User Document Set.

LSMCO also requires the StorHouse VRAM component.

Function Prototype Definition

```
extern int LSMCO ( struct LSMS_TOKEN *c_token,
                  int message_flag,
                  struct LSMS_TOKEN *o_token,
                  char *file_name,
                  char *file_password,
                  char *group_name,
                  char *group_password,
                  char *model_file_name,
                  struct LSMS_FLOC *file_location,
                  struct LSMS_CATTR *file_attr
                  );
```

Data Structures

```
struct LSMS_FLOC
{
    char volumeset_name[ 9 ];
    char fileset_name[ 9 ];
};

struct LSMS_CATTR
{
    long list_size;
    long block_size;
    long checkpoint;
    long file_size;
```



```

    long data_xlate_flag;
    long atf;
    long cache;
    long edc;
    long limit;
    long vtf;
    long retention_interval;
};

```

Argument Description

c_token	A pointer to a session identifier token (connect token).
message_flag	An integer that indicates how text messages are handled. If non-zero (defined value LSMF_MSGHOLD), then text messages from all data transfer errors, including LSMCLO (close) errors, are retained in the message stack. If zero (defined value LSMF_NOMSGHOLD), messages are not retrievable after LSMCLO has been called.
o_token	A pointer to a token (LSMS_TOKEN) structure. It is not necessary to initialize this structure, because LSMCO sets it to the file identifier. The application program should not manipulate the members of this structure. It should be used only as the o_token parameter to other LSMxxx calls for this transfer operation.
file_name	A 56-byte character string containing the StorHouse file name or the (operating system dependent) symbolic variable to be referenced. If the symbolic variable is specified, the string must begin with the characters DD=. In this case, the file name used is the operating system's translation of the string following DD=. If file_name is longer than 56 characters, LSMCO fails with a return code of 2949.
file_password	<p>An 8-character variable containing the write password for the newly created file. The value of file_password must match the model file's write password (see model_file_name). The read and delete passwords for the new file are copied from the model file's read and delete passwords.</p> <p>If the model file has no write password, file_password should point to a null or all-blanks string.</p> <p>If no model file name is specified, the file_password value becomes the read, write, and delete passwords for the newly created file. The file_password value also supplies the write and delete passwords for any existing version of that file.</p>
group_name	<p>An 8-byte character string that identifies the file access group name for the newly created file and for the model file (see model_file_name). If the file is stored under the account's default group, group_name may point to a null or all-blanks string.</p> <p>SETGROUP privilege is required to specify any group other than the default group.</p>
group_password	An 8-character variable containing the write password for the StorHouse file access group. If no write password is defined for the group, group_password should point to a null or all-blanks string.

model_file_name A 56-byte character string containing the StorHouse model file name or the (operating system dependent) symbolic variable to be referenced. If the symbolic variable is specified, the string must begin with the characters DD=. In this case, the file name used is the operating system's translation of the string following DD=. If **model_file_name** is longer than 56 characters, LSMCO fails with a return code of 2949.

The model file must already exist on the StorHouse system. File characteristics for **file_name** are determined by copying the characteristics of the model file. Non-default values in the **file_attrb** list override these characteristics.

Only RECORD files can be created without a model file specification. If **model_file_name** is specified as a blank or null string, then file attributes are determined from the **file_attrb** list and from system/user defaults.

A prior version of a file cannot be used as a model for a new version of that same file; in other words, **model_file_name** may not be the same as **file_name**.

file_location A pointer to a file location structure containing the identifiers for the file's destination volume set and file set. The file location structure members are:

- **volumeset_name** – an 8-byte character string that supplies the name of the file's destination volume set.
- **fileset_name** – an 8-byte character string that supplies the name of the file's destination file set.

If a volume set or file set member is blank or null, the default for the StorHouse account is used. If the account default is also blank, then the specification is copied from the model file.

file_attrb A pointer to a structure containing a list of long integers that provide file attributes. The caller specifies a value for the first member in the structure, **list_size**, which contains the number of other members in the structure. To supply or retrieve all available file attributes, set **list_size** to 10. The value of **block_size** should be set to 0. The **file_size** member is required. All other attributes are optional. The caller must supply a value for all members that are included in the list. Attributes not included assume a value of 0.

The values for **file_attrb** members are either integers or flags. Integer-valued members are either positive or zero. A positive value supplies the specific value used for the parameter. Zero indicates use of the default value.

Note The actual value of the default may not equal 0.

Flags have one of three values:

- Positive indicates true (the option is selected).
- Negative indicates false (the option is not selected).

- Zero indicates use of the default value.

Non-default values override attributes that are determined from the model file. If no model file name is specified, non-default values override normal StorHouse file attribute defaults.

The members of the LSMS_CATTR structure are:

- `list_size` – the number of other members in the list.
- `block_size` – a user-defined value that specifies the size in bytes of a buffer area used by the Callable Interface. This member should be set to 0.
- `ckpt` – a caller-supplied value that indicates the checkpoint number where file processing should be restarted. A value of 0 indicates normal (non-restart) operations. If a non-zero value is specified, the remaining structure members are ignored.
- `file_size` – the number of bytes of storage space (in units of 1000 bytes) allocated whenever a file is opened for an append operation and whenever a checkpoint is issued. The value must contain enough space for the largest extent set that is written. This extent set includes a data extent, a DF extent, and for KEYED files, a K extent. A file size must always be specified (non-zero) for file creation (in other words, `ckpt=0`).
- `data_xlate_flag` – a flag value. If positive, data is stored as ASCII characters. Data is translated from the native host character set to ASCII when the file is stored and translated from ASCII to the native character set of the receiving host when data is retrieved. This allows use of character files on host systems with non-ASCII character sets.
- `atf` – the StorHouse Access Time Factor (ATF). Values may equal 1, 2, or 3. Refer to the *Command Language Reference Manual* in the StorHouse User Document Set for more information about ATF.
- `cache` – the number of sequential records that VRAM caches for an LSMRS, LSMRR, or an LSMRK function for this file when it is opened with an access mode of READ or UPDATE and an access method including RECORD and/or KEYED. The system can use this cache to optimize subsequent sequential reads.
- `edc` – a flag value. A positive value indicates that error detection coding (EDC) is enabled. A negative value indicates that error detection coding is disabled.
- `limit` – file version limit value, which may equal a positive number between 1 and 32768. A value of 0 indicates use of the default limit value.
- `vtf` – Vulnerability Time Factor (VTF), which may equal 2, 3, or 4. A value of 2 indicates VTF=NEXT; 3 indicates VTF=NOW; and 4 indicates VTF=DIRECT.

Refer to the *Command Language Reference Manual* in the StorHouse User Document Set for more information about VTF.

- `retention_interval` – the retention attribute for the file. Valid values are:
 - Number of days specified as a non-zero, positive integer (for example, 60).
 - `LSMV_RETEN_FOREVER`, which indicates infinite retention.
 - `LSMV_RETEN_ZERO`, which indicates the file has no retention period and can be deleted.
 - `LSMV_DEFAULT`, which indicates the retention period is not specified at the file level and assumes the default value.
 - If the file's resident file set has a retention attribute equal to `FOREVER`, `ZERO`, or a specified number of days, the file set retention attribute determines the file retention attribute.
 - If the file's resident file set has a retention attribute of `DEFAULT`, the `RETENTION_MODE` system parameter determines the file retention attribute. If `RETENTION_MODE` is set to `BASIC`, the file retention is `ZERO`. If `RETENTION_MODE` is set to `STRICT`, the file retention is `FOREVER`.

Return Codes

2629 Specifies that the caller supplied an invalid checkpoint number.

2635 May be caused by the following errors:

- `LSMCO` was used to create a new version of the model file.
- The specified model file is open for write or update by another user.
- A user tried to create-open a file whose highest version was already in use.

Refer to the message text retrieved by `LSMMSG` to identify the specific cause of error.

Any Other Non-Zero
Value

Indicates that the file was not created and is not open. Any other StorHouse functions relating to this file should not be issued. In particular, `LSMCLO` will fail because of an invalid `o_token`.

Function Description

`LSMCO` creates a VRAM file on the StorHouse system and builds an open data transfer path to allow write operations to this new file. The value of `file_name` identifies the VRAM file. The `c_token` is the session identifier returned by `LSMCON`. `LSMCO` returns a file identifier in the `o_token` structure. After a successful `LSMCO`, users may perform operations for the newly created file.

Notes

- Each LSMCO establishes another transfer link and returns another file identifier (o_token). It is the user's responsibility to maintain the integrity of the open tokens.
- If the amount of space indicated by file_size cannot be allocated, StorHouse returns an error code. Refer to the *Command Language Reference Manual* (CREATE FILE command description) for information about how to estimate VRAM file sizes.
- A non-zero return code indicates that there may be associated error messages. These messages may be retrieved using LSMMSG. Specify the c_token rather than the o_token in the LSMMSG call.
- Generally model files should be created only for use as models, not for use as data files. When a file is used as a model, it is referenced (mounted) as part of LSMCO processing. If the model is on optical, a physical platter mount may be required. Allocating models as empty files on level F storage prevents this extra platter mount.
- Appendix A in the *Generic Callable Interface Programmer's Guide* contains programming guidelines for using multiple open statements and checkpoints. These guidelines also apply to LSMCO.

Cross-Reference to Sample Program

There is no cross-reference to the sample program contained in Chapter 6, "Sample Program," in the *Generic Callable Interface Programmer's Guide*.

LSMOS - Open Sequential

LSMOS establishes a data transfer link between the user program and StorHouse, sets the direction of data flow, and identifies the StorHouse file being opened. Sequential record transfer operations (read or write) may then be performed on the file.

SETGROUP privilege is required to use LSMOS. For more information about StorHouse privileges, refer to the *Command Language Reference Manual*.

Function Prototype Definition

```
extern int LSMOS ( struct LSMS_TOKEN *c_token,
                  int message_flag,
                  struct LSMS_TOKEN *o_token,
                  char *mode,
                  char *file_name,
                  long version,
                  struct LSMS_FPW *file_passwords,
                  char *group_name,
                  struct LSMS_FPW *group_passwords,
                  struct LSMS_FLOC *file_location,
                  struct LSMS_ATTR *file_attr,
                  struct LSMS_OPTS *file_options
                );
```

Data Structures

```
struct LSMS_FPW
{
    char  read_password[ 9 ];
    char  write_password[ 9 ];
    char  delete_password[ 9 ];
};
```

```
struct LSMS_FLOC
{
    char  volumeset_name[ 9 ];
    char  fileset_name[ 9 ];
};
```

```
struct LSMS_ATTR
{
    long  list_size;
    long  file_size;
    long  max_record_len;
    long  transport_flag;
    long  data_xlate_flag;
    long  fixed_record_fl;
    long  cc_ansi_flag;
```

```

        long  cc_mach_flag;
        long  block_size;
        long  retention_interval;
    };

    struct LSMS_OPTS
    {
        long  list_size;
        long  lock;
        long  wait;
        long  atf;
        long  edc;
        long  limit;
        long  new; /* This will be newx if compiled with C++. */
        long  unlock;
        long  vtf;
    };

```

Argument Description

c_token	A pointer to a session identifier token (connect token).
message_flag	An integer that indicates how indicative text messages are handled. If non-zero (defined value LSMF_MSGHOLD), then text messages from all data transfer errors, including close function errors, are retained in the message stack. If zero (defined value LSMF_NOMSGHOLD), messages are not retrievable after LSMCLO (Close) has been called.
o_token	A pointer to a token (LSMS_TOKEN) structure. The structure need not be initialized. LSMOS fills in this structure with transfer operation identification and clears the asynchronous processing indicator. The application program should not manipulate the members of this structure. It should only be used as the open token (o_token parameter) to other LSMxxx calls for this transfer operation.
mode	A character string that identifies the file reference mode. The acceptable values are READ and WRITE. These values may be specified in upper or lower case.
file_name	A 56-byte character string containing the StorHouse file name or the (operating system dependent) symbolic variable to be referenced. If the symbolic variable is specified, the string must begin with the characters DD=. In this case, the file name used is the operating system's translation of the string following DD=. If the file_name is longer than 56 characters, LSMOS fails with a return code of 2949.
version	A long integer containing the file's version number. This argument applies only to READ operations. Zero is the default (most current) version. A negative value indicates a relative version number. Positive values are not supported.
file_passwords	A pointer to a structure containing three 9-character (eight data characters plus one byte for null-termination) variables that contain the read, write, and delete passwords, respectively, associated with the file name. The structure member for a

password that is not supplied should be set to null. The length of a file/group password (eight characters) is defined value LSML_FILEPW.

group_name	The identifier for the file access group. If the file is stored under the user's default group, this parameter may be omitted; that is, its value may be null. The maximum length for this string is 8 (defined value LSML_GROUPNAME).
group_passwords	A pointer to a password structure containing the read, write, and delete passwords for the group. The structure format is the same as the structure format used for file_passwords.
file_location	A pointer to a file location structure containing the identifiers for the file's destination volume set and file set. This argument applies only to WRITE operations. The file location structure members are: <ul style="list-style-type: none"> • volumeset_name – a character string that supplies the name of the file's destination volume set. • filesset_name – a character string that supplies the name of the file's destination file set.
file_attrb	A pointer to a structure containing a list of long integers that provide file attributes. The caller specifies a value for the first member in the list, list_size, which contains the number of other members in the structure. To supply or retrieve all available file attributes, set list_size to 9.

The caller must supply a value for all members included in the structure. Attributes not included in the structure assume a value of 0, which indicates use of the default. (The actual default value may not equal zero.)

Note the following:

- The value of block_size should be set to 0.
- For a mode of WRITE, the caller specifies file attributes. The file_size member is required. All other attributes are optional.
- For a mode of READ, all file attributes, except block_size and retention_interval, are returned to the caller.
- For flag values, a negative value implies the opposite of the positive value. Zero indicates that the default is used.

The members of the file attribute list are:

- list_size – number of other members in the structure.
- file_size – the total file size in bytes. This is an estimate that must be larger than the actual number of bytes being transferred.

- `max_record_len` – the maximum length for any record in the file.
- `transport_flag` – a flag value. If positive, the file is in a transportable format that can be retrieved by dissimilar host systems.
- `data_xlate_flag` – a flag value. If positive, the data is stored as ASCII characters. The data is translated from the native host character set to ASCII when the file is stored. The data is translated from ASCII to the native character set of the receiving host when retrieved. This allows the use of character files on host systems with non-ASCII character sets.
- `fixed_record_fl` – a flag value. If positive, the records are fixed-length records.
- `cc_ansi_flag` – a flag value. If positive, the first character of each record is a print carriage control character of the FORTRAN (or ANSI) type.
- `cc_mach_flag` – a flag value. If positive, the first character of each record is a machine-specific print carriage control character.
- `block_size` – a user-defined value that specifies the size in bytes of a buffer area used by the Callable Interface. This element is not used and should be set to 0.
- `retention_interval` – the retention period for the file. Valid values are:
 - Number of days specified as a non-zero, positive integer (for example, 60).
 - `LSMV_RETEN_FOREVER`, which indicates infinite retention.
 - `LSMV_RETEN_ZERO`, which indicates the file has no retention period and can be deleted.
 - `LSMV_RETEN_DEFAULT`, which indicates the retention period is not specified at the file level and assumes the default value.
- If the file's resident file set has a retention attribute equal to `FOREVER`, `ZERO`, or a specified number of days, the file set retention attribute determines the file retention attribute.
- If the file's resident file set has a retention attribute of `DEFAULT`, the `RETENTION_MODE` system parameter determines the file retention attribute. If `RETENTION_MODE` is set to `BASIC`, the file retention is `ZERO`. If `RETENTION_MODE` is set to `STRICT`, the file retention is `FOREVER`.

`file_options` A pointer to a structure containing a list of long integers that provide file transfer options. These options correspond to the StorHouse GET and PUT command modifiers. For more information about GET and PUT, refer to the *Command Language Reference Manual*.

Each member is either an integer or a flag value. Integers are positive or zero. Zero indicates that the default value is used. (Note that the actual default value may not equal zero.) Flags are any positive non-zero value (indicates “true” and the option is selected), any negative value (indicates the opposite of “true”), or zero (indicates use of the default).

The caller must supply a value for all attributes included in the structure. Attributes not included in the structure assume a value of 0, which indicates use of the default.

The first member in the structure, `list_size`, must contain the number of the other members in the structure. To provide all options, set `list_size` to 8.

The structure members are:

- `list_size` – number of other members in the structure.
- `lock` – lock flag. A positive value indicates that the file is to remain explicitly locked after the file operation completes.
- `wait` – wait for file lock flag.
 - For READ operations, a positive value indicates that the data transfer operation should wait for a locked file to be unlocked.
 - For WRITE operations, this field is no longer used. It is not necessary to change existing code. For new programs, set this field to 0.
- `atf` – Access Time Factor, a positive integer equal to 1, 2, or 3. This field is used only for WRITE operations.
- `edc` – Error Detection Code (EDC) identifier, an integer equal to: 1 or 2 to explicitly select the coding algorithm; 0 to indicate use of the default; or negative to generate no codes. A value of zero is recommended. This field is used only for WRITE operations.
- `limit` – file version LIMIT value, a positive integer between 1 and 32768. This field is used only for WRITE operations.
- `new` – new file flag. A positive value indicates that a previous version of the file (same group and file name) must not exist in StorHouse. This field is used only for WRITE operations. This will be `newx` if compiled with C++.
- `unlock` – unlock flag. This field is obsolete. It is not necessary to change existing code. For new programs, set `unlock` to 0.
- `vtf` – Vulnerability Time Factor, an integer equal to 2, 3, or 4. A value of 2 indicates `/VTF=NEXT`; 3 indicates `/VTF=NOW`; and 4 indicates `/VTF=DIRECT`. This field is used only for WRITE operations.

Refer to the *Command Language Reference Manual* for information about the VTF attribute.

Return Codes

Any Non-Zero Value The file was not opened. No other Callable Interface functions relating to this file should be issued. In particular, LSMCLO will fail due to an invalid o_token.

Detailed Function Description

LSMOS initiates a file transfer operation between the host and StorHouse. The StorHouse file is identified by the file_name and group_name identifiers. The mode parameter determines the type of processing, either READ or WRITE. The transfer is performed during the session identified by the connect token (c_token).

After the file has been opened successfully (return code zero), read or write functions may be called. The transfer operation is ended by issuing LSMCLO (Close).

LSMOS returns an open token (o_token), which identifies the transfer operation path to subsequent data transfer operations such as read and write. A StorHouse file opened with LSMOS can only be processed sequentially. Facilities implemented by the optional VRAM component, such as reading a record by relative record number, require that the file be created as a VRAM file and that transfer operations be initiated with LSMOV (Open VRAM).

If the message flag is set (non-zero), then LSMMSG must be called after the transfer operation ends (in other words, after LSMCLO [Close] has been called). The dynamic memory allocated for the operation is not released until all messages have been retrieved; that is, until a return code of 3065, indicating no more messages, is received from LSMMSG.

Notes

- If LSMOS returns a non-zero status code, then any associated error messages can be retrieved with the LSMMSG function. The token parameter for LSMMSG should be the connect token, not the open token.
- Appendix D of the *Generic Callable Interface Programmer's Guide* contains information about using multiple open statements.

Cross-Reference to Sample Program

See steps 2 and 5 in the sample program in Chapter 6, "Sample Program," in the *Generic Callable Interface Programmer's Guide*.

Mainframe Callable Interface

This section contains changes to the mainframe Callable Interface CREATE-OPEN and OPEN-SEQ functions.

CREATE-OPEN

CREATE-OPEN creates a new VRAM file on StorHouse, and then establishes a data transfer link for writing data to that file. CREATE-OPEN is equivalent to issuing a StorHouse Command Language CREATE FILE command followed by OPEN-VRAM in mode APPEND. CREATE-OPEN requires the StorHouse VRAM component.

CREATE-OPEN requires RECORD privilege. For more information about StorHouse privileges, refer to the *Command Language Reference Manual*.

Statement Format for COBOL

TSO/Batch/IMS Environment

```
CALL 'LSMCALL' USING CREATE-OPEN, C-TOKEN, R-CODE,
                     MESSAGE-FLAG, O-TOKEN, FILE-NAME
                     FILE-PASSWORD, GROUP-NAME,
                     GROUP-PASSWORD, MODEL-FILE-NAME,
                     FILE-LOCATION, FILE-ATTRIB.
```

CICS Environment

```
CALL 'LSMCICS' USING DFHEIBLK, COMMAREA,
                     CREATE-OPEN, C-TOKEN, R-CODE,
                     MESSAGE-FLAG, O-TOKEN, FILE-NAME,
                     FILE-PASSWORD, GROUP-NAME,
                     GROUP-PASSWORD, MODEL-FILE-NAME,
                     FILE-LOCATION, FILE-ATTRIB.
```

Working Storage Section for COBOL Program

```
01 CREATE-OPEN          PIC X(16)  VALUE 'CREATE-OPEN'.
01 C-TOKEN              PIC S9(8)   COMP SYNC.
01 R-CODE               PIC S9(8)   COMP SYNC.
01 MESSAGE-FLAG         PIC S9(8)   COMP SYNC.
01 O-TOKEN              PIC S9(8)   COMP SYNC.
01 FILE-NAME            PIC X(56).
```

```

01 FILE-PASSWORD          PIC X(8) .
01 GROUP-NAME             PIC X(8) .
01 GROUP-PASSWORD         PIC X(8) .
01 MODEL-FILE-NAME        PIC X(56) .
01 FILE-LOCATION.
    05 VOLUMESET-NAME      PIC X(8) .
    05 FILESET-NAME        PIC X(8) .
01 FILE-ATTRIB.
    05 FATTR-LIST-SIZE     PIC S9(8)  COMP SYNC VALUE 10 .
    05 FATTR-BLOCK-SIZE   PIC S9(8)  COMP SYNC .
    05 FATTR-CHECKPOINT   PIC S9(8)  COMP SYNC .
    05 FATTR-FILE-SIZE    PIC S9(8)  COMP SYNC .
    05 FATTR-DATA-XLATE   PIC S9(8)  COMP SYNC .
    05 FATTR-ATF          PIC S9(8)  COMP SYNC .
    05 FATTR-CACHE        PIC S9(8)  COMP SYNC .
    05 FATTR-EDC          PIC S9(8)  COMP SYNC .
    05 FATTR-LIMIT        PIC S9(8)  COMP SYNC .
    05 FATTR-VTF          PIC S9(8)  COMP SYNC .
    05 FATTR-RETENTION-INTERVAL PIC S9(8)  COMP SYNC .

```

Parameter Overview

C-TOKEN	The session identifier returned by CONNECT.
R-CODE	Final status from the requested operation; see the following section “Return Codes.”
MESSAGE-FLAG	An integer set to zero or non-zero. If non-zero, this flag indicates that the caller requires text messages from all data transfer errors including CLOSE function errors. If zero, messages may not be retrievable after CLOSE has been issued.
O-TOKEN	Variable set by CREATE-OPEN to the file identifier. The application program should not manipulate (in particular, not cause arithmetic conversion to) the result; it should only be used as the O-TOKEN parameter to other function calls for this file.
FILE-NAME	A 56-byte character string that contains either the StorHouse file name or the DDname to be referenced. If the DDname is specified, the string must begin with the characters DD=. In this case, the file name used will be the DSNAMES specified on the named DD statement. File names shorter than 56 characters must be padded on the right with blanks.
FILE-PASSWORD	An 8-character variable containing the write password for the file. This password must match the write password for the model file (see MODEL-FILE-NAME) unless the user has the privilege to bypass file passwords. If the user has the privilege to bypass file passwords, this value is not used unless there is no model file. All other passwords for the new file will be copied from the passwords defined for the model file. If no write password is defined for the model file, this variable should be set to all blanks.

If no model file name is provided, the FILE-PASSWORD value becomes the new read, write, and delete passwords for the newly created file. The file password value also supplies the write and delete passwords for any existing version of that file.

GROUP-NAME An 8-byte character string that identifies the file access group for the new file and for the model file (see MODEL-FILE-NAME). If the file is stored under the account's default group, this parameter may be specified as all blanks. SETGROUP privilege is required to specify any group other than the user's default.

GROUP-PASSWORD An 8-character variable containing the write password for the file access group. If no write password is defined for the group, this variable should be set to all blanks.

MODEL-FILE-NAME A 56-byte character string that contains either the StorHouse file name or the DDname to be referenced. If the DDname is specified, the string must begin with the characters DD=. In this case, the file name used will be the DSNNAME specified on the named DD statement. File names shorter than 56 characters must be padded on the right with blanks.

The model file must already exist on StorHouse. File characteristics for the new file (whose name is given by FILE-NAME) are determined by copying the characteristics of the model file. These characteristics are overridden by non-default values in the FILE-ATTRIB array.

Only RECORD type files can be created without a model file specification. If blanks are specified for the model file name, then file attributes are determined only from the FILE-ATTRIB array.

MODEL-FILE-NAME must not be the same as FILE-NAME. That is, a prior version of a file cannot be used as a model for a new version of the same file.

FILE-LOCATION An array of two, 8-character variables containing the file's destination volume set name and file set name. If a variable contains all blanks, the default value associated with the StorHouse account is used. If the account's default value is also blank, the value is copied from the model file.

FILE-ATTRIB An array of 32-bit integers that provide file attributes. The first entry in the array must be set to the number of other elements in the array. To provide all attributes, set FATTR-LIST-SIZE to 10.

Other entries in the array are either integers or flag values.

- Integers are either positive or 0. Zero indicates that the default is used.

Note: The actual default value may not equal zero.

- Flags have one of three values:
 - Positive indicates true (the option is selected).
 - Negative indicates false (the option is not selected).
 - Zero indicates use of the default value.

The caller must supply a value for all attributes included in the array. Attributes not included in the array assume a value of 0. Either a file size or checkpoint must be supplied.

Non-default values override attributes determined from the model file. If no model file name is specified, non-default values override normal StorHouse file attribute defaults.

The elements of the FILE-ATTRIB array are:

- FATTR-LIST-SIZE – the number of other elements in the array.
- FATTR-BLOCK-SIZE – the size in bytes of a buffer area used by the Callable Interface to block user records prior to moving data to the StorHouse Subsystem. The caller does not have to reserve this area because it is GETMAINED and FREEMAINED by the StorHouse software.

The caller supplies the value for block size. A value of 0 causes a site-selected value to be used for block size. A value of 1 to 256 causes buffering to be bypassed. The recommended block size is between 32,000 and 100,000 bytes and should be large enough to contain two or more records plus four bytes.

- FATTR-CHECKPOINT – a checkpoint number at which file processing should be restarted. For normal (non-restart) operations, 0 must be specified. If a non-zero checkpoint value is specified, then the remaining entries in this attribute array are ignored.
- FATTR-FILE-SIZE – the number of bytes of storage space (in units of 1000 bytes) allocated whenever a file is opened for an append operation and whenever a checkpoint is issued. The value must contain enough space for the largest extent set that is written. This extent set includes a data extent, a DF extent, and for KEYED files, a K extent. A file size must always be specified (non-zero) for file creation (in other words, FATTR-CHECKPOINT value is zero). Refer to the *Command Language Reference Manual* for more information about specifying file size.
- FATTR-DATA-XLATE – a flag value; if positive, data is stored as ASCII characters. The data is translated from EBCDIC to ASCII as it is transferred to StorHouse and is translated from ASCII to the local code for the host as it is transferred to the host (for IBM mainframes the local code is EBCDIC).
- FATTR-ATF – Access Time Factor, a positive integer equal to 1, 2, or 3. Refer to the *Command Language Reference Manual* for additional information.
- FATTR-CACHE – the number of records cached by StorHouse during read operations for files opened with a mode of READ or UPDATE and a method including RECORD or KEYED. The cache value may be specified as a negative value. A negative value turns off caching and ignores the cache specification for the model file.

- FATTR-EDC – a flag value; if positive error detection coding is enabled. If negative, EDC is disabled.
- FATTR-LIMIT – the file version limit value, a positive integer between 1 and 32768, or 0 for default.
- FATTR-VTF – Vulnerability Time Factor, an integer equal to 2, 3, or 4.
 - 2 indicates a VTF of NEXT
 - 3 indicates a VTF of NOW
 - 4 indicates a VTF of DIRECT.

Refer to the *Command Language Reference Manual* for additional information about the VTF attribute.

- FATTR-RETENTION-INTERVAL – the retention period for the file. Valid values are:
 - Non-zero positive integer (for example, 60), which indicates the number of days to retain the file.
 - -2, which indicates to retain the file forever.
 - -1, which indicates zero days, or no retention.
 - 0, which indicates the retention period is not specified at the file level and assumes the default value. StorHouse determines the default value as follows:

If the file's resident file set has a retention attribute equal to -2, -1, or a specified number of days, the file set retention attribute determines the file retention attribute.

If the file's resident file set has a retention attribute of 0, the RETENTION_MODE system parameter determines the file retention attribute. If RETENTION_MODE is set to BASIC, the file retention is ZERO. If RETENTION_MODE is set to STRICT, the file retention is FOREVER.

Return Codes

2629 Indicates that the caller supplied an invalid checkpoint number.

2635 May be caused by the following errors:

- CREATE-OPEN was used to create a new version of the model file.
- The specified model file is open for write or update by another user.
- A user tried to CREATE-OPEN a file whose highest version was already in use.

Refer to the error message text retrieved by EMSG to identify the specific cause of error.

**Any Other Non-Zero
Code**

Indicates that the file was not created and is not open. Any other StorHouse functions relating to this file should not be issued. In particular, CLOSE will fail because of an invalid O-TOKEN.

Detailed Function Description

CREATE-OPEN creates a VRAM file on StorHouse and builds an open data transfer path to allow WRITE operations to that file. The VRAM file is identified by the value of FILE-NAME. C-TOKEN is the session identifier returned by CONNECT. CREATE-OPEN returns a file identifier in the O-TOKEN variable. After a successful CREATE-OPEN (a return code of zero), operations for this file can be performed.

Notes

- Each CREATE-OPEN establishes another transfer link and returns another file identifier (O-TOKEN). It is the responsibility of the user to maintain the integrity of the open tokens.
- If the amount of space indicated by the FATTR-FILE-SIZE variable cannot be allocated, StorHouse returns an error code. Refer to the *Command Language Reference Manual* (CREATE FILE command) for information about how to estimate VRAM file sizes.
- If the return code is non-zero, there may be associated error messages. These messages can be retrieved using the EMSG function. The C-TOKEN (not the O-TOKEN) must be specified in the EMSG call.
- A session can be established in one task (under one TCB) and then used in another task; however, only one session-related function can be performed at one time for one session. Serialization between multiple tasks is the user's responsibility. CREATE-OPEN must be considered a session-related function.
- If a file is opened and closed under one TCB and written from another TCB, the two tasks must share Subpool 0 storage. If one of these tasks is a subtask of the other, this is accomplished by the SZERO=YES operand on the ATTACH MACRO (this is the default value).
- Generally, model files should be created only for use as models, not for use as data files. When a file is used as a model, it is referenced (mounted) as part of CREATE-OPEN processing. If the model is on optical storage, a physical platter mount may be required. Allocating models as empty files on level F storage prevents this extra platter mount.
- The additional technical information about programming guidelines for using multiple open statements and checkpoints in Appendix C of the *Callable Interface Programmer's Guide* also applies to CREATE-OPEN.

OPEN-SEQ

OPEN-SEQ establishes a data transfer link between the user program and StorHouse, sets the direction of the data flow, and identifies the file that will be referenced. This function allows sequential transfer of complete files, using the read or write functions. OPEN-SEQ requires StorHouse standard features.

Statement Format for COBOL

TSO/Batch/IMS Environment

```
CALL 'LSMCALL' USING OPEN-SEQ, C-TOKEN, R-CODE, MESSAGE-FLAG,
                     O-TOKEN, MODE, FILE-NAME, VERSION,
                     FILE-PASSWORDS, GROUP-NAME,
                     GROUP-PASSWORDS, FILE-LOCATION,
                     FILE-ATTRIB, FILE-OPTIONS.
```

CICS Environment

```
CALL 'LSMCICS' USING DFHEIBLK, COMMAREA,
                     OPEN-SEQ, C-TOKEN, R-CODE, MESSAGE-FLAG,
                     O-TOKEN, MODE, FILE-NAME, VERSION,
                     FILE-PASSWORDS, GROUP-NAME,
                     GROUP-PASSWORDS, FILE-LOCATION,
                     FILE-ATTRIB, FILE-OPTIONS.
```

Working Storage Section for COBOL Program

```
01 OPEN-SEQ                      PIC X(16)  VALUE 'OPEN-SEQ'.
01 C-TOKEN                      PIC S9(8)   COMP SYNC.
01 R-CODE                      PIC S9(8)   COMP SYNC.
01 MESSAGE-FLAG                PIC S9(8)   COMP SYNC.
01 O-TOKEN                    PIC S9(8)   COMP SYNC.
01 MODE                       PIC X(6).
01 FILE-NAME                  PIC X(56).
01 VERSION                   PIC S9(8)   COMP SYNC.
01 FILE-PASSWORDS.
   05 FILE-READ-PASSWORD      PIC X(8)    VALUE SPACES.
   05 FILE-WRITE-PASSWORD     PIC X(8)    VALUE SPACES.
   05 FILE-DELETE-PASSWORD    PIC X(8)    VALUE SPACES.
01 GROUP-NAME                 PIC X(8).

01 GROUP-PASSWORDS.
   05 GROUP-READ-PASSWORD     PIC X(8)    VALUE SPACES.
   05 GROUP-WRITE-PASSWORD    PIC X(8)    VALUE SPACES.
```

```

    05 GROUP-DELETE-PASSWORD    PIC X(8)    VALUE SPACES.
01 FILE-LOCATION.
    05 VOLUMESET-NAME           PIC X(8).
    05 FILESET-NAME             PIC X(8).
01 FILE-ATTRIB.
    05 FATTR-LIST-SIZE          PIC S9(8)    COMP SYNC VALUE 9.
    05 FATTR-FILE-SIZE          PIC S9(8)    COMP SYNC.
    05 FATTR-MAX-RECORD-LEN     PIC S9(8)    COMP SYNC.
    05 FATTR-TRANSPORT-FLAG     PIC S9(8)    COMP SYNC.
    05 FATTR-DATA-XLATE-FLAG    PIC S9(8)    COMP SYNC.
    05 FATTR-FIXED-RECORD-FL    PIC S9(8)    COMP SYNC.
    05 FATTR-CC-ANSI-FLAG       PIC S9(8)    COMP SYNC.
    05 FATTR-CC-MACH-FLAG       PIC S9(8)    COMP SYNC.
    05 FATTR-BLOCK-SIZE         PIC S9(8)    COMP SYNC.
    05 FATTR-RETENTION-INTERVAL PIC S9(8)    COMP SYNC.
01 FILE-OPTIONS.
    05 FOPTS-LIST-SIZE          PIC S9(8)    COMP SYNC VALUE 8.
    05 FOPTS-LOCK               PIC S9(8)    COMP SYNC.
    05 FOPTS-WAIT               PIC S9(8)    COMP SYNC.
    05 FOPTS-ATF                PIC S9(8)    COMP SYNC.
    05 FOPTS-EDC                PIC S9(8)    COMP SYNC.
    05 FOPTS-LIMIT              PIC S9(8)    COMP SYNC.
    05 FOPTS-NEW                PIC S9(8)    COMP SYNC.
    05 FOPTS-UNLOCK             PIC S9(8)    COMP SYNC.
    05 FOPTS-VTF                PIC S9(8)    COMP SYNC.

```

Parameter Overview

C-TOKEN	Session identifier (connect token).
R-CODE	Final status from the requested operation; see the following section “Return Codes.”
MESSAGE-FLAG	An integer set to zero or non-zero. If non-zero, MESSAGE-FLAG indicates that the caller requires indicative text messages from all data transfer operation errors including CLOSE errors. If zero, messages may not be retrievable if the data transfer has terminated.
O-TOKEN	A variable that is set to the data transfer operation identifier (open token). The application program should not manipulate (in particular, not cause arithmetic conversion to) the result; it should only be used as the O-TOKEN parameter to other function calls for this file.
MODE	A 6-byte character string that identifies the file reference mode. Valid MODE values are READ and WRITE.
FILE-NAME	A 56-byte character string that contains either the StorHouse file name or the DDname to be referenced. If the DDname is specified, the string must begin with the characters “DD=”. In this case, the file name used will be the DSNNAME specified on the named DD statement. File names shorter than 56 characters must be padded on the right with blanks.

VERSION	File version number, which applies only to READ operations. Zero is the default (most current) version. A negative value indicates a relative version number. Positive values are not supported.
FILE-PASSWORDS	An array of three 8-character variables containing the read, write, and delete passwords associated with the file name. The array entry for a password that is not supplied must be all blanks.
GROUP-NAME	An 8-byte character string that identifies the file access group. If the file is stored under the user's default group, this parameter need not be supplied; that is, its value must be all blanks.
GROUP-PASSWORDS	An array of three 8-character variables containing the read, write, and delete passwords for the group. The GROUP-PASSWORDS array has the same format as the FILE-PASSWORDS array.
FILE-LOCATION	An array of two 8-character variables containing the file's destination volume set name and file set name, respectively. If a default is used, FILE-LOCATION should contain all blanks. This parameter applies to WRITE operations only.
FILE-ATTRIB	<p>An array of 32-bit integers that provides file attributes. The caller specifies values for the first entry in the array, FATTR-LIST-SIZE, and for FATTR-BLOCK-SIZE. FATTR-LIST-SIZE contains the number of the other elements in the file attributes array. To supply or retrieve all available file attributes, set FATTR-LIST-SIZE to 9.</p> <p>For MODE=WRITE, the caller specifies file attributes. FATTR-FILE-SIZE is required. All other attributes are optional. The caller must supply a value for all attributes included in the array. Attributes not included in the array assume a value of 0, which indicates use of the default. (The actual default value may not equal zero.)</p> <p>For MODE=READ, all file attributes, except for FATTR-BLOCK-SIZE, are returned to the caller.</p> <p>For flag values, a negative value implies the opposite of the positive value; zero indicates that the default is used.</p> <p>The elements in the FILE-ATTRIB array must be listed in the following order:</p> <ul style="list-style-type: none"> • FATTR-LIST-SIZE – number of other elements in the array. • FATTR-FILE-SIZE – total file size in bytes. This estimate must be larger than the actual number of bytes that will be transferred. • FATTR-MAX-RECORD-LEN – maximum length for any record in the file. • FATTR-TRANSPORT-FLAG – flag value; if positive, the file is in a transportable format that can be retrieved by dissimilar host systems.

- FATTR-DATA-XLATE-FLAG – flag value; if positive, data will be stored as ASCII characters. The data is translated from EBCDIC to ASCII when the file is stored on StorHouse and translated from ASCII to EBCDIC when retrieved by the host.
- FATTR-FIXED-RECORD-FL – flag value; if positive, the records are fixed length.
- FATTR-CC-ANSI-FLAG – flag value; if positive, the first character of each record is a print carriage control character of the FORTRAN (or ANSI) type.
- FATTR-CC-MACH-FLAG – flag value; if positive, the first character of each record is a print carriage control character of “machine” type.
- FATTR-BLOCK-SIZE – size in bytes of a buffer area used by the StorHouse software to block user records prior to moving data to or from the StorHouse Subsystem. The caller does not have to reserve this area because it is GETMAINED and FREEMAINED by the StorHouse software.
- FATTR-RETENTION-INTERVAL – the retention period for the file. Valid values are:
 - Non-zero positive integer (for example, 60), which indicates the number of days to retain the file.
 - -2, which indicates to retain the file forever.
 - -1, which indicates zero days, or no retention.
 - 0, which indicates the retention period is not specified at the file level and assumes the default value. StorHouse determines the default value as follows:

If the file's resident file set has a retention attribute equal to -2, -1, or a specified number of days, the file set retention attribute determines the file retention attribute.

If the file's resident file set has a retention attribute of 0, the RETENTION_MODE system parameter determines the file retention attribute. If RETENTION_MODE is set to BASIC, the file retention is ZERO. If RETENTION_MODE is set to STRICT, the file retention is FOREVER.

The caller specifies the value for block size. A value of zero defaults to the site-selected value for block size. A value of 1 to 256 causes buffering to be bypassed.

The recommended block size is between 32,000 and 100,000 bytes and should contain 2 or more records plus 4 bytes.

FILE-OPTIONS

An array of 32-bit integers that provide file options. These options correspond to the StorHouse GET and PUT command modifiers. For information about GET and PUT, refer to the *Command Language Reference Manual*.

The caller sets the first entry in the FILE-OPTIONS array, FOPTS-LIST-SIZE, to the number of the other elements in the array. To access all options, set FOPTS-LIST-SIZE to 8.

Other entries are either integer or flag values.

- Integers are either positive or 0. Zero indicates that the default value is used.

Note: The actual default value may not equal zero.

- Flags are any positive value (indicates “true” and the option is selected), any negative value (indicates the opposite of “true”), or zero (indicates use of the default).

The caller must supply a value for all attributes included in the array. Attributes not included in the array assume a value of 0, which indicates use of the default.

The elements in the FILE-OPTIONS array are:

- FOPTS-LIST-SIZE – number of other elements in the array.
- FOPTS-LOCK – lock flag. A positive value indicates that the file is to be explicitly locked; it will remain locked after the file operation completes.
- FOPTS-WAIT – wait for file lock flag.
 - For READ operations, a positive value indicates that the data transfer operation should wait for a locked file to be unlocked.
 - For WRITE operations, this field is no longer used. It is not necessary to change existing code. For new programs, set this field to 0.
- FOPTS-ATF – Access Time Factor (ATF). ATF can be 1, 2, or 3. This field is used for WRITE operations only.
- FOPTS-EDC – error detection code identifier. FOPTS-EDC can be a positive integer equal to 1 or 2; zero to indicate use of the default (which is recommended); or negative to indicate that no EDC will be generated for data in the file. This field is used for WRITE operations only.
- FOPTS-LIMIT – file version LIMIT value. FOPTS-LIMIT can be a positive integer between 1 and 32768. This field is used for WRITE operations only.
- FOPTS-NEW – new file flag. A positive value indicates that a previous version of the file (same group and file name) must not exist in StorHouse. This field is used for WRITE operations only.
- FOPTS-UNLOCK – unlock flag. This field is obsolete. It is not necessary to change existing code. For new programs, set FOPTS-UNLOCK to 0.

- FOPTS-VTF – Vulnerability Time Factor (VTF). VTF is an integer equal to 1, 2, 3, or 4. A value of 1 indicates /VTF=NEVER; 2 indicates /VTF=NEXT; 3 indicates /VTF=NOW; and 4 indicates /VTF=DIRECT. This field is used for WRITE operations only.

Refer to the *Command Language Reference Manual* for information about the VTF attribute.

Return Codes

Any non-zero value indicates that the file was not opened. In this case, any other StorHouse functions relating to this file should not be issued. In particular, CLOSE will fail due to an invalid O-TOKEN.

Detailed Function Description

OPEN-SEQ is used to open a non-VRAM file on StorHouse. The StorHouse file name is identified by the value of FILE-NAME, and the type of processing is provided by the value of MODE. C-TOKEN contains the session identifier returned by CONNECT. OPEN-SEQ returns a file identifier in the O-TOKEN variable. After a successful OPEN-SEQ (that is, a return code of zero), other StorHouse functions relating to this file can be performed.

A StorHouse file opened with OPEN-SEQ can only be processed sequentially. Facilities implemented by the optional VRAM component, such as reading a record by record number, cannot be used.

If MESSAGE-FLAG is set (non-zero), the application must call EMSG after the file is closed. The dynamic memory allocated for the transfer operation is not released until all messages have been returned; that is, a 3065 return code, indicating no more messages, has been received from EMSG.

Notes

- Each OPEN-SEQ establishes a transfer link and returns a file identifier (O-TOKEN). It is the user's responsibility to maintain the integrity of the open tokens.
- A session can be established in one task (under one TCB) and then used in another task; however, only one session-related function can be performed at one time for one session. Serialization between multiple tasks is the user's responsibility.

OPEN-SEQ must be considered a session-related function.

- If a file is opened and closed under one TCB and read or written from another TCB, the two tasks must share Subpool 0 storage. If one of these tasks is a subtask

of the other, this is accomplished by the SZERO=YES operand on the ATTACH MACRO (this is the default value).

- If the return code is not zero and the associated messages (if any) are to be retrieved, EMSG should be called specifying the C-TOKEN rather than the O-TOKEN.
- Refer to the *Callable Interface Programmer's Guide* for a discussion of programming guidelines for using multiple open statements.

Documentation Updates

This chapter contains documentation updates to the MIGRATE, RELOCATE, and SHOW ACCOUNT commands in the StorHouse 5.4 *Command Language Reference Manual* since its last publication (March 28, 2002).

MIGRATE Command

On page 3-153 of the *Command Language Reference Manual*, the second paragraph should read:

StorHouse moves each selected file to a destination file set in a destination volume set. If the destination volume set does not exist, StorHouse returns an error. If the destination file set does not exist, StorHouse creates one with the same name as the source file set using the /CONTIGUOUS or /NONCONTIGUOUS setting of the source file set and the CREATE FSET command default values for other attributes (see page 3-68 for those defaults). If you want the destination file set to have different values from the defaults, you must create it yourself before you run the MIGRATE /BY_VSET command. FileTek recommends that you always use the default file set LIMIT of 0.

RELOCATE Command

The /FSET parameter modifier documentation for the RELOCATE command should read:

- /FSET Specifies the file set containing the files to be copied.
- FORMAT: /FSET=fset_name or /FSET=*
 - DEFAULT: If you omit /FSET, the default is FSET=*

- **RESTRICTIONS:**
 - If you specify /FSET, you must also specify /VSET or /VOLUME.
 - The source and destination file sets must be in the same directory but in different volume sets.
 - If you specify the performance buffer file set name, no files will be selected for relocation.

The /VSET parameter modifier documentation should read:

/VSET Specifies the volume set that contains the source file to be relocated.

- **FORMAT:** /VSET=vset_name or /VSET=*
- **DEFAULT:** If you omit /VSET, the default is VSET=*
- **RESTRICTIONS:**
 - /VSET is mutually exclusive with /VOLUME and /DIRECTORY.
 - The source and destination volume sets must be different; however, they must be located in the same directory.

In the examples section of the RELOCATE command, the third example should read:

- To relocate all versions of USERFILE in the archive directory to the file set ARCHFS1 in the volume set ARCHVS1, enter:

```
? RELOCATE USERFILE /VERSION=* /DIR=ARCHIVE
/TO_VSET=ARCHVS1 /TO_FSET=ARCHFS1
```

SHOW ACCOUNT Command

The information for the aid parameter should read:

aid Specifies the account identification code (aid) of the account for which information is to be displayed.

- **FORMAT:**
 - aid
 - partial_aid*
 - *
- **DEFAULT:** If you omit aid, the system uses your own account.

ACCESS REQUIREMENTS: In addition to SHOW, you must have SHOACCOUNT or ANYACCOUNT privilege to specify an account other than your own.