



StorHouse/RFS Utilities Reference Manual

StorHouse/RFS Release 4.0

Publication Number
900181 Rev. B

April 6, 2006

The FileTek logo consists of the word "FileTek" in white, sans-serif font, centered within a teal square.



All rights reserved. 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.

Copyright © 2004-2006 FileTek, Inc. As an Unpublished Licensed Work.
Publication Number: 900181 Rev. B

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 Unpublished licensed work property of FileTek, Inc. Unauthorized use, duplication or distribution prohibited.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/manufacture 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. 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

Purpose of this guide vii

Audience..... viii

What’s inside viii

Document conventions..... viii

Recommended reading ix

Chapter 1: Introduction

What are the StorHouse/RFS utilities? 1-2

Where do I run StorHouse/RFS utilities?..... 1-2

When can I run StorHouse/RFS utilities? 1-3



Contents

What are the format conventions?.....	1-3
What and where are log files?	1-5

Chapter 2: The tblgen utility

Purpose of tblgen.....	2-2
How tblgen works	2-2
Before running tblgen	2-3
How to run tblgen.....	2-3
After running tblgen	2-5
Prompts and responses.....	2-5
Example.....	2-8

Chapter 3: The rfsmaint utility

Purpose of rfsmaint.....	3-2
How rfsmaint works	3-2
Identifying collections eligible for deletion or compaction	3-3
Compacting collections and preparing them for deletion	3-4
Removing references to orphan collections.....	3-5
Before running rfsmaint.....	3-6
How to run rfsmaint.....	3-7
After running rfsmaint	3-8
Format of rfsmaint.....	3-8
Examples	3-12



Chapter 4: The rfsdelete utility

Purpose of rfsdelete.....	4-2
Before running rfsdelete.....	4-2
How to run rfsdelete.....	4-3
After running rfsdelete.....	4-4
Format of rfsdelete.....	4-5
Examples	4-6

Chapter 5: The checkfile utility

Purpose of checkfile.....	5-2
Before running checkfile.....	5-2
How to run checkfile.....	5-3
After running checkfile.....	5-4
Format of checkfile.....	5-5
Examples	5-6

Index



Contents



Welcome

StorHouse/Relational File System (RFS) is the FileTek file system interface that enables your applications to store and access a virtually unlimited number of files on *StorHouse*®—FileTek’s data storage and management system for capturing, storing, moving, and accessing gigabytes (GB) to petabytes of relational and non-relational enterprise data.

Purpose of this guide

The *StorHouse/RFS Utilities Reference Manual* provides a reference for the utilities used to maintain StorHouse/RFS. This manual contains utility formats, examples, and guidelines for usage.



Welcome

Audience

The *StorHouse/RFS Utilities Reference Manual* is intended for personnel who are responsible for maintaining StorHouse/RFS systems. This document assumes that you are experienced with the StorHouse operating system (UNIX) and that you understand StorHouse/RFS concepts.

What's inside

This manual is a work-in-progress that will be updated as new utilities are developed. The current version of the document describes four StorHouse/RFS utilities: `tblgen`, `rfsmaint`, `rfsdelete`, and `checkfile`.

Document conventions

The following conventions are used in this manual.

Table i: Document conventions

Convention	Meaning
<i>Italics</i>	Emphasized text, new terms, and document titles
<code>This font</code>	Utility formats and examples and text that you enter
▼	Procedures



Recommended reading

This manual assumes you are familiar with the information in the following publications.

- The *StorHouse/RFS Concepts*, publication number 900159, defines the StorHouse/RFS hardware and software components, storage and retrieval processes, security features, and StorHouse/RFS terminology, such as collection, virtual file system, and staging area.
- The *StorHouse/RFS Administration Guide*, publication number 900178, explains how to set up the storage resources necessary to store data on StorHouse through StorHouse/RFS. It also contains planning and setup information for collecting and storing data.



Welcome

C H A P T E R 1



Introduction

This chapter describes basic information about StorHouse/RFS utilities. It answers the following questions:

- What are the StorHouse/RFS utilities?
- Where do I run StorHouse/RFS utilities?
- When can I run StorHouse/RFS utilities?
- What format conventions are used?
- What and where are log files?



What are the StorHouse/RFS utilities?

StorHouse/RFS currently provides the following utilities.

- **tblgen** - Creates StorHouse database tables necessary for StorHouse/RFS operation. See Chapter 2 for more information about the **tblgen** utility. *The StorHouse/RFS Administration Guide* also explains how to run this utility.
- **rfsmaint** - Manages space in StorHouse collections and in two StorHouse database tables: file locator tables and collections tables. See Chapter 3 for more information about the **rfsmaint** utility.
- **rfsdelete** - Deletes StorHouse collections prepared for deletion by the **rfsmaint** utility. See Chapter 4 for more information about the **rfsdelete** utility.
- **checkfile** - Locates CRC errors in user files and optionally places a copy of a file in a designated directory for further analysis. See Chapter 5 for more information about the **checkfile** utility.
- **rfsrestore** - Recovers out of sync databases after a crash or between StorHouse systems. This utility is run by a FileTek customer support representative and is not described in this manual.

Where do I run StorHouse/RFS utilities?

The **rfsmaint**, **rfsdelete**, and **rfsrestore** StorHouse/RFS utilities reside on the StorHouse system with the StorHouse databases and collections used



by StorHouse/RFS. You issue these utility commands at the StorHouse operating system (UNIX) prompt. Use the StorHouse system operator account and password to run these StorHouse/RFS utilities. You can also run the `rfsmaint` utility and the `rfsdelete` utility on the StorHouse/RFS server platform (Windows or UNIX) or from any system that can access the StorHouse/RFS configuration file or a copy of it and that has ODBC and TCP/IP connectivity to StorHouse.

The `tblgen` and `checkfile` utilities reside on the StorHouse/RFS server platform. You run this utility at a command prompt or terminal window.

When can I run StorHouse/RFS utilities?

You may run the StorHouse/RFS utilities described in this manual at any time (24 x 7). (The `rfsrestore` utility is the only exception.) StorHouse/RFS may be running when you execute the utilities.

What are the format conventions?

The `tblgen` utility prompts for input. The other StorHouse/RFS utilities use a command-line format consisting of a program, or utility, name followed by options and arguments. For example, the format of the `rfsdelete` utility is as follows:

```
rfsdelete -c <rfs config file name> -d <storage or  
collection definition name> [-m] [-a] [-b]  
[-w <log directory>]
```



Chapter 1 - Introduction

In this format:

- `rfsdelete` is the program name
- `-c` and `-d` are required options
- `<rfs config file name>` and `<storage or collection definition name>` are required arguments
- `-w`, `-m`, `-a`, and `-b` are optional options

The following conventions are used.

- `< >` – Braces identify arguments. You must provide a value for a required argument. An optional argument has a default value.
- `[]` – Brackets indicate optional command components (options and arguments), which you can omit from the command line.
- Program names – Utility names are not case sensitive for Windows but are for UNIX systems.
- Options – Options begin with a dash and are not case sensitive, for example, `-d` or `-D`.
- Arguments – Argument values follow an option and may be case sensitive. For example, database names and directories are case sensitive.
- Spacing – A space is required after the program name, after each option, and after each argument value. The only exception is that spaces are not allowed when entering multiple table type values for the `tblgen` utility (see page 2-5).



- | – A vertical bar (not shown in the previous example) separates two command components of which at least one is required. For example:

```
-s <since date or newer than days>|-b <before date or  
older than days>
```

Below is an example utility command with values.

```
rfsdelete -c c:\Program Files\RFS\v4r0\rfs.cfg  
-d ST_RFS3l -m -w c:\Program Files\RFS\v4r0\deletelog
```

What and where are log files?

The tblgen and checkfile utilities write messages to a log file located in the same directory as the program executable. The other StorHouse/RFS utilities write messages to a log file located in a directory that you designate when running the utility or in the default directory (same location as the utility program). You can check the log file to determine whether the utility ran successfully. If a log contains an error message, call FileTek customer support for clarification or to report the problem. Your FileTek customer support representative may request the log to assist with problem resolution.



Chapter 1 - Introduction

C H A P T E R

2



The tblgen utility

This chapter describes the tblgen utility, including:

- Purpose of tblgen
- How tblgen works
- Before running tblgen
- How to run tblgen
- After running tblgen
- Prompts and responses
- Example



Purpose of `tblgen`

The *tblgen utility* creates the StorHouse tables necessary for StorHouse/RFS to function correctly. These tables, which reside in the system tablespace of a StorHouse database, are as follows:

- Aliases table – Contains old and new user and group names, or aliases, to enable access to files using either the old or new name.
- Collections table – Identifies the StorHouse files, or StorHouse collections, that have been written to StorHouse. Once this table is created, StorHouse/RFS creates a table array, or set of 32 tables, for storing file locator data. StorHouse/RFS subsequently creates up to a total of 255 sets of 32 tables (as required) to store file locator information.
- Security table – Contains security information for n levels of directories below each collector's staging directory. Currently, the security table is supported for Windows installations only.
- Statistics tables – Contains information for monitoring and managing staging, cache space, and local collection space; collection and search activity; and StorHouse writes and reads. The `tblgen` utility creates three statistics tables.

How `tblgen` works

The `tblgen` utility prompts for input, including the type of table to create, host name, database name, table name, table owner, and password. The security table requires additional input: user directory (`UserDir` parameter in the StorHouse/RFS configuration file) and number of subdirectories (`KeepSubdirectories` parameter). After you provide the input, the `tblgen` utility connects to the specified database on the



specified StorHouse system using the provided account and password and then creates the tables. With the exception of the security table, if the tables already exist, the utility prompts you to re-run the utility with a different table name.

Before running tblgen

Before running tblgen, you must create the StorHouse database, the StorHouse account that will be used as the owner of the tables, and the account password with StorHouse/Admin. At a minimum, the account must have SQLEXECUTE and RESOURCE privileges. If the same account will be used for creating StorHouse collections, it must additionally have ATF, DELETE, GET, PUT, RECORD, SETGROUP, SHOW, and VTF privileges.

The user directory you specify when running the tblgen utility must match the UserDir value in the collector definition in the StorHouse/RFS configuration file. You can create the collector definition before or after creating the security table. Just be sure the user directories match.

How to run tblgen

You run the tblgen utility on the system where StorHouse/RFS is installed from a command prompt on Windows or a terminal window on UNIX.

▼ To run tblgen on Windows

1. Log in to the StorHouse/RFS server platform with the administrator account.
2. From the Start menu, select Run.



Chapter 2 - The tblgen utility

3. Enter `CMD` and click OK.
4. In the **DOS** box enter the drive letter where StorHouse/RFS is installed, for example:

`C:`

5. Change to the StorHouse/RFS executable directory, for example:

```
CD "\Program Files\RFS\v4r0"
```

The quotes are necessary if StorHouse/RFS is installed in the Program Files directory.

6. Start the tblgen utility by typing:

`tblgen`
7. Respond to the prompts. See “Prompts and responses” on page 2-5 for more information.

▼ To run tblgen on UNIX

1. If necessary, log in to the StorHouse/RFS server as `root`. You should be running bourne shell.
2. Execute the following command to get the necessary definitions.

```
. /rfs/v4r0/bin/defs
```

3. Change to the StorHouse/RFS configuration directory:

```
cd /rfs/v4r0/conf
```

4. Start the tblgen utility:

```
./tblgen.exe
```



5. Respond to the prompts. See “Prompts and responses” on page 2-5 for more information.

After running tblgen

The tblgen utility creates a log in the directory where the tblgen executable resides. You can access this log to verify work performed. The utility messages indicate the location of the log. See page 2-10 for an example of the tblgen log.

Prompts and responses

Table 2-1 lists the prompts that the tblgen utility issues and the responses that you provide.

Table 2-1: Prompts and responses for the tblgen utility

Prompt	Response
Enter the type of RFS tables to create	Any combination of the following letters: <ul style="list-style-type: none">■ S - Statistics tables■ L - Collections table■ A - Aliases table■ B - Security table When specifying multiple letters, do not use spaces or commas. Example: SLAB
Enter the host name	The DNS name of the host (StorHouse system) where the tables will reside. Example: alpha2



Chapter 2 - The tblgen utility

Table 2-1: Prompts and responses for the tblgen utility (continued)

Prompt	Response
Enter the database name	<p>The name of the StorHouse database that will contain the tables used by StorHouse/RFS. The database name is case sensitive.</p> <p>Example: JOURNALDATABASE</p>
Enter the base table name	<p>The part of the table name common to the type(s) of table(s) being created. StorHouse/RFS table names have the following format:</p> <p>base_table_name_<value></p> <p>where base_table_name is a name that you provide and <value> is a suffix provided by StorHouse/RFS to identify the purpose of the table. Values may be:</p> <ul style="list-style-type: none">■ _C for the collections table■ _S for the security table■ _RFS, _SYSTEMS, and _TABLES for statistics tables■ _ALIASES for the aliases table <p>The simplest RFS configuration would use the same base table name for all tables. However, you may designate any base table for any type of table by running the tblgen utility multiple times with different table types and different base table names.</p> <p>Example: If you are creating all table types (slab) and specify a base table name of RFSTABLE, then the collections table would be named RFSTABLE_C; the security table would be named RFSTABLE_S; and so on.</p>
Enter the owner for the tables	<p>The StorHouse account that is authorized to create the tables for StorHouse/RFS.</p> <p>Example: SYSADM</p>
Enter the owner password	<p>The corresponding password for the StorHouse account. The password is not displayed.</p>

If you are creating a security table (table type B), the tblgen utility issues the additional prompts listed in Table 2-2.

Table 2-2: Additional prompts and responses for the security table

Prompt	Response
Enter a collector's UserDir or type 'Q' to quit	The UserDir path as specified in a collector definition in the StorHouse/RFS configuration file. Examples: \\mailboxes /journal
Enter KeepSubdirectories value for this directory	The number of subdirectory levels below the staging directory on which StorHouse/RFS will retain security information for this collector's UserDir. Example: 4



Example

The following tblgen example creates all of the StorHouse tables with the same base table name of RFSPROD. The example also creates multiple security tables.

```
tblgen

*** RFS Table Generator for RFS 4.0 ***
*** Copyright 2005, FileTek, Inc. ***
*** As an unpublished licensed work.***
*** Log File is C:\Rufus-VC\TblGen\v4.0\tblgen.log

Enter the type of RFS tables to create.
Choices are S - statistics,
L - locator,
A - aliases,
B - security
or combination without spaces: slab

Enter the host name: alpha2

Enter the database name (case sensitive): DEVRFS4

Enter the base table name: RFSPROD

Enter the owner for the tables: SYSADM

Enter the owner password: *****

Creating statistics tables SYSADM.RFSPROD in database
DEVRFS4 on alpha2

Continue? (Y or N) y

Statistics tables successfully created

Creating locator table SYSADM.RFSPROD_C in database
DEVRFS4 on alpha2
```


Example



```
Continue? (Y or N) y

Locator table successfully created

Creating alias table SYSADM.RFSPROD_ALIASES in database
DEVRF4 on alpha2

Continue? (Y or N) y

Aliases table successfully created

Creating/updating security table SYSADM.RFSPROD_S in
database DEVRF4 on alpha2

Continue? (Y or N) y

Security table successfully created

Enter a collector's UserDir or type 'Q' to quit

\Publications

Enter the KeepSubdirectories value for this directory: 5

Create entry for directory \Publications
keeping 5 levels of directory below it?

(Enter 'y' or 'n'): y

Enter a collector's UserDir or type 'Q' to quit

\Other

Enter the KeepSubdirectories value for this directory: 1

Create entry for directory \Other
keeping 1 levels of directory below it?
```



Chapter 2 - The tblgen utility

```
(Enter 'y' or 'n'): y
```

```
Enter a collector's UserDir or type 'Q' to quit
```

```
q
```

The following example illustrates the log messages located in the following directory:

```
C:\Rufus-VC\TblGen\v4.0\tblgen.log
```

```
2005-12-06 13:38:00.414 I- 2856: New log file started for  
dev-elf.filetek.com
```

```
2005-12-06 13:38:00.414 I- 2856: *** RFS Table Generator  
for RFS 4.0 ***
```

```
2005-12-06 13:38:00.414 I- 2856: *** Copyright 2005,  
FileTek, Inc. ***
```

```
2005-12-06 13:38:36.706 I- 2856: Creating statistics  
tables SYSADM.RFSPROD in database DEVRFS4 on alpha2
```

```
2005-12-06 13:38:39.020 I- 2856: Statistics tables  
successfully created
```

```
2005-12-06 13:38:43.326 I- 2856: Creating locator table  
SYSADM.RFSPROD_C in database DEVRFS4 on alpha2
```

```
2005-12-06 13:38:44.688 I- 2856: Locator table  
successfully created
```

```
2005-12-06 13:38:46.600 I- 2856: Creating alias table  
SYSADM.RFSPROD_ALIASES in database DEVRFS4 on alpha2
```

```
2005-12-06 13:38:47.261 I- 2856: Aliases table  
successfully created
```

Example



```
2005-12-06 13:38:53.410 I- 2856: Creating/updating
security table SYSADM.RFSPROD_S in database DEVRFS4 on
alpha2

2005-12-06 13:38:54.983 I- 2856: Security table
successfully created

2005-12-06 13:39:20.870 I- 2856: Added security table
entry. SeqNo: 1 KeepLevel: 5 Directory: \Publications

2005-12-06 13:39:32.888 I- 2856: Added security table
entry. SeqNo: 2 KeepLevel: 1 Directory: \Other

2005-12-06 13:39:36.302 I- 2856: Completed adding entries
for SYSADM.RFSPROD_S

2005-12-06 13:39:36.302 I- 2856: *** RFS Table Generator
ended ***
```



Chapter 2 - The tblgen utility



The rfsmaint utility

This chapter describes the rfsmaint utility, including:

- Purpose of rfsmaint
- How rfsmaint works
- Before running rfsmaint
- How to run rfsmaint
- After running rfsmaint
- Format of rfsmaint
- Examples



Purpose of rfsmaint

The *rfsmaint utility* is a space management tool that:

- Removes rows from file locator tables for user files that have been marked deleted by a user or application. A *file locator table* contains one row for each user file in a StorHouse collection.
- Removes rows from the *collections table*, called *_C table*, for each deleted StorHouse collection. A *_C table* contains one row for each StorHouse collection in a collection set.
- Consolidates, or *compacts*, the space occupied by the deleted files in the StorHouse collections. Or marks an entire StorHouse collection as deleted.

How rfsmaint works

The rfsmaint utility performs the following functions:

- Identifies collections eligible for compaction or deletion
- Compacts collections and prepares them for deletion
- Removes references to orphan collections



Identifying collections eligible for deletion or compaction

You specify criteria that the rfsmaint utility uses to determine which collections are eligible for deletion or compaction. This criteria includes:

- The age of the data, for instance, collections written before a certain date, or older than a number of days, or since a certain date, or newer than a number of days. For example, process only those collections written since one year ago but not newer than 30 days.
- The number of days after a retention period has expired, for example, process collections that have been expired for at least 30 days.
- The percentage of deleted space in collections, for instance, process collections with 75% of deleted space.
- A specific collection name to process only one StorHouse collection. This is useful, for instance, when a user file is stored inadvertently in a StorHouse collection and must be removed for security reasons.

The rfsmaint utility first searches for collections that might be eligible to be deleted or compacted. It checks the `_C` table for collections that match the age criteria.

From that list, the utility checks retention requirements. Any collection that has not passed its retention period (specified by the Retention parameter in the StorHouse/RFS configuration file) is not eligible for deletion or compaction regardless of the age criteria. If a collection's retention period has expired and all utility criteria are met, then a collection is eligible for consideration.



Compacting collections and preparing them for deletion

Your criteria may include a percentage of deleted collection space required before the rfsmaint utility compacts or deletes a StorHouse collection. The default is 100%, which means all user files in a collection must be marked deleted before the utility takes action.

Compaction is the process of moving undeleted files in a StorHouse collection to a new, smaller collection. This occurs when you specify percentage criteria between 1% and 99% (for example, 75%) and that criteria is met.

Deletion is the process of marking an entire StorHouse collection as deleted. This occurs when you specify percentage criteria of 100%.

After compaction, the original StorHouse collection is also marked for deletion. (The rfsmaint utility does not delete StorHouse collections but rather prepares them for deletion.) The utility creates a *_D table* (delete table) in the StorHouse database with a row for each collection that can be deleted. The rfsdelete utility described in Chapter 4 then uses the *_D* table to perform the actual deletes.

The rfsmaint utility compacts collections and prepares them for deletion as follows. After compiling a list of potentially eligible collections, the rfsmaint utility searches the file locator tables for rows that have data located in the collections. The utility determines the number of deleted and undeleted user files and the amount of deleted space in each collection. If the percentage of space deleted exceeds the percentage specified, then the rfsmaint utility executes according to the following rules:

- If the specified percentage is 100, then all user files in the eligible StorHouse collections must be marked deleted before the collections are eligible for deletion. If all files are marked deleted, then the



rfsmaint utility deletes all rows in the file locator tables and all rows in the `_C` table for the collections. The utility adds the collection names to the `_D` table.

- If the specified percentage is between 1 and 99, then that percentage of space in the eligible StorHouse collections must be occupied by files that are marked deleted in the file locator data. The rfsmaint utility writes the undeleted user files to new StorHouse collections, updates the file locator data and `_C` table for the new collections, and adds the names of the original StorHouse collections to the `_D` table.

You can also perform an *unconditional delete* where all eligible StorHouse collections are deleted. None of the user files need to be marked deleted. The rfsmaint utility deletes all rows in the file locator tables and all rows in the `_C` table for the collections. The utility adds the collection names to the `_D` table.

Removing references to orphan collections

Authorized users can delete collections directly on StorHouse using the StorHouse DELETE and REMOVE commands or through StorHouse/Control Center. Collections with retention, however, cannot be removed from StorHouse until the retention has expired. This method, however, produces *orphan collections* because rows for the deleted user files exist in the file locator data and rows for the deleted collections exist in the `_C` table.

You may request the rfsmaint utility to remove all references to orphan collections. The default is to keep the references. The rfsmaint utility searches for rows corresponding to the deleted collections in the file locator tables and deletes those rows. The utility then removes the collection names from the `_C` table.



Before running rfsmaint

Before executing rfsmaint, you can run the utility in *report mode* to list the actions performed when the utility executes. No deletion or compaction occurs in report mode. The listing includes the following items.

- Your specified criteria, for instance:

```
Purging collections written before 2003-12-01 12:00:01
Collection space must be at least 100% deleted
Orphaned rows for missing collections will be deleted.
```

- The list of eligible collections:

```
DIRECT20031117173719(A).DEV-ELF.FILETEK.COM
DIRECT20031117174733(A).DEV-ELF.FILETEK.COM(1)
DIRECT20031118084334(A).DEV-ELF.FILETEK.COM
...
There are 11 collections to process.
```

- A compaction evaluation for each eligible collection:

```
Processing collection 1 of 11:
DIRECT20031117173719(C).DEV-ELF.FILETEK.COM

Beginning deletion check

Space [ Total: 9000000 Deleted: 0 ]   Rows [ Total:
1000 Deleted: 0 ]

% Deleted: 0

-**- Collection not eligible for compacting **
```



- A deletion check in the file locator tables and the _C table:

```
Beginning deletion check

Space [ Total: 0 Deleted: 0 ]   Rows [ Total: 30592
Deleted: 0 ]

% Deleted: 0

SM files to be deleted: 0

Rows to be deleted: 0
```

Also, if you wish to process a specific collection (see the `-f` option on page 3-11), you must first obtain the collection name by submitting a FIND or XMLFIND command through StorHouse/RFS. Refer to the *StorHouse/RFS Administration Guide* for more information about obtaining a collection name for a specific user file.

How to run rfsmaint

You run the rfsmaint utility at the StorHouse operating system prompt or from any system that can access the StorHouse/RFS configuration file or a copy of it and that has ODBC and TC P/IP connectivity to StorHouse.

▼ To run the rfsmaint utility at the StorHouse operating system prompt

1. Log in to the StorHouse system using the operator account and password.
2. At the StorHouse operating system prompt, type the utility command, and press **Enter**. See “Format of rfsmaint” on page 3-8 for more information about the rfsmaint utility command.



After running rfsmaint

After running rfsmaint, examine the log file located in the default directory or the alternate directory (if specified on the command line) to ensure that the utility ran successfully.

You may run the rfsmaint utility as often as is necessary to clear space in StorHouse databases and to prepare StorHouse collections for deletion. The StorHouse collections are not deleted, however, until after you run the metadata backup utility followed by the rfsdelete utility. See Chapter 4, “The rfsdelete utility,” for more information about removing StorHouse collections.

Format of rfsmaint

The rfsmaint utility consists of the following options and arguments.

```
rfsmaint -c <rfs config file name> -d <storage or  
collection definition name> -s <since date or newer than  
days>|-b <before date or older than days> [-p <1 - 100>]  
[-u] [-f <collection name>] [-m] [-x] [-r <number of  
days>] [-w <log directory>] [-o ] [-l]
```



Table 2-1: Options/Arguments for the rfsmaint utility

Option/Argument	Description
-c <rfs config file name>	<p>(required) The fully qualified path of the StorHouse/RFS configuration file that contains the storage definition or collection definition with the information required to connect to StorHouse and perform the necessary StorHouse database and maintenance tasks. The configuration file may be on the machine running the rfsmaint utility or on a machine running StorHouse/RFS that the rfsmaint utility can access, or just a copy of the StorHouse/RFS configuration file with the relevant information.</p> <p>Example: -c c:\Program Files\RFS\v4r0\rfs.cfg</p>
-d <storage or collection definition name>	<p>(required) The name of the storage definition or collection definition identifying the StorHouse database name, table name, system definition name, account ID, and password. For StorHouse/RFS 3.0, specify the collection name. For StorHouse/RFS 4.0, specify the storage definition name.</p> <p>Example: -d ST_RFS31</p>
-s <since date or newer than days>	<p>(required if the -b option is omitted) The written since date (yyyy-mm-dd:hh:mm:ss) or newer than days age criteria used to evaluate StorHouse collections for compaction or deletion. Collections written since the specified date or that are newer than the specified number of days are considered for maintenance. You can also use both the -s and -b options.</p> <p>Examples:</p> <ul style="list-style-type: none"> ■ -s 2003-12-01 requests to consider collections written from now through December 1, 2003. ■ -s 365 -b 7 requests to consider collections written in the last 365 days but older than 7 days. <p>Note the following:</p> <ul style="list-style-type: none"> ■ The default is 0 which means consider all collections ever written to StorHouse/RFS. ■ No spaces are allowed in the date. ■ The yyyy-mm-dd part of the date, if specified, is required. ■ If you omit the time (hh:mm:ss), the default is 00:00:00.



Chapter 3 - The rfsmaint utility

Table 2-1: Options/Arguments for the rfsmaint utility (continued)

Option/Argument	Description
-b <before date or older than days>	<p>(required if the -s option is omitted) The before date (yyyy-mm-dd:hh:mm:ss) or older than days age criteria used to evaluate StorHouse collections for compaction or deletion. Collections written before the specified date or that are older than the specified number of days are considered for maintenance.</p> <p>Examples:</p> <p>-b 2005-08-31:23:00:00 requests to consider collections written before August 31, 2005, 11 p.m.</p> <p>-b 30 requests to consider collections written before the last 30 days.</p> <p>Note the following:</p> <ul style="list-style-type: none">■ The default is now (when the program is run) which means consider all collections written before the current date and time.■ The same format rules as the -s option apply to the -b option.
-p <1 - 100>	<p>(optional) The minimum percentage of deleted space in a StorHouse collection before it may be deleted or compacted.</p> <p>Example: -p 50 means compact eligible collections when at least half of the space occupied by files is marked deleted.</p> <p>Note the following:</p> <ul style="list-style-type: none">■ The default is 100, which means delete eligible collections only when all files are marked deleted.■ A value between 1 and 99 means compact eligible collections when the specified percentage is met.■ The value 0 is not permitted.■ You cannot use the -p parameter with the -u parameter.
-u	<p>(optional) An option to perform an unconditional delete of all eligible collections within the age range. The rfsmaint utility deletes all database rows for the eligible collections, removes the entries from the _C table, and deletes the StorHouse files. You cannot use the -u parameter with the -p parameter.</p>



Table 2-1: Options/Arguments for the rfsmaint utility (continued)

Option/Argument	Description
-f <collection name>	<p>(optional) An option to process a specific collection. The -f option is useful when a user file is inadvertently stored in a StorHouse collection and must be removed for security reasons. You can determine the collection name for the file by doing a FIND or XMLFIND command in StorHouse/RFS. Then after deleting the file through the StorHouse/RFS file system interface, run the rfsmaint utility with the -f parameter and -p 1. The rfsmaint utility deletes the row in the file locator data for the deleted file and rewrites the StorHouse collection without the data for the deleted file.</p> <p>Example: -f DIRECT20050203170548(A).DEV-ELF.FILETEK.COM</p>
-m	<p>(optional) An option to also perform maintenance on the mirror system specified in the StorHouse/RFS configuration file (identified by the -d parameter). If you omit the -m parameter, the rfsmaint utility performs maintenance on the primary system only.</p>
-x	<p>(optional) An option to exclude StorHouse collections consisting only of metadata (.ldr file). This might happen when many files are deleted or their permissions are updated or when a collection has had all of its data deleted but the metadata contains rows for files in other collections that are not yet deleted. Determining whether the metadata is no longer needed requires much database activity while the space gained by deleting the collection is very small. Therefore excluding collections with metadata only and then processing them on a less frequent basis than collections with data saves time.</p>
-r <number of days>	<p>(optional) The number of days a collection with a retention period must be expired before it will be considered for purging.</p> <p>Example: -r 30 means a collection's retention period must be expired by a minimum of 30 days before the collection is eligible for deletion.</p> <p>The default is 0, which means a collection may be eligible for deletion on the retention expiration date.</p>



Table 2-1: Options/Arguments for the rfsmaint utility (continued)

Option/Argument	Description
-w <log directory>	(optional) The name of an alternate directory where the log file will be written. This value is case sensitive. If you omit the -w option, the default location is the directory where the rfsmaint program resides. Example: -w ./rfsutils
-o	(optional) Option to remove rows from file locator tables and the _C table for orphan collections (StorHouse collections deleted outside of StorHouse/RFS). The default is to keep the references to orphan collections.
-l	(optional) Option to run the utility in report mode only, that is, to list the actions that will be performed when rfsmaint executes.

Examples

This section contains examples using different rfsmaint options and arguments. The examples use the StorHouse/RFS configuration file located in `c:\Program Files\RFS\v4r0\rfs.cfg` and the collection definition named `ST_RFS31`.

- Delete everything (-u unconditional delete) older than three years (-b 1095 days), remove references to orphan collections (-o), and exclude metadata-only collections (-x):

```
rfsmaint -c c:\Program Files\RFS\v4r0\rfs.cfg  
-d ST_RFS31 -b 1095 -u -o -x
```

- Compact collections written since January 1, 2003 (-s 2003-01-01) that have over 50% (-p 50) of their files deleted by users or



applications, and perform the same maintenance on the mirror StorHouse system (-m):

```
rfsmaint -c c:\Program Files\RFS\v4r0\rfs.cfg  
-d ST_RFS31 -s 2003-01-01 -p 50 -m
```

- Same as the previous example, with the additional criteria that any collection with a retention period must have expired by 30 days (-r 30):

```
rfsmaint -c c:\Program Files\RFS\v4r0\rfs.cfg  
-d ST_RFS31 -s 2003-01-01 -p 50 -r 30
```

- Same as the previous example, except run in report mode to display the actions that will be taken without doing the compacting:

```
rfsmaint -c c:\Program Files\RFS\v4r0\rfs.cfg  
-d ST_RFS31 -s 2003-01-01 90 -p 50 -r 30 -l
```

- Compact the specific collection named
DIRECT20050203170548(A).DEV-ELF.FILETEK.COM:

```
rfsmaint -c c:\Program Files\RFS\v4r0\rfs.cfg  
-f DIRECT20050203170548(A).DEV-ELF.FILETEK.COM -p 1 -b
```



Chapter 3 - The rfsmaint utility

C H A P T E R

4



The rfsdelete utility

This chapter describes the rfsdelete utility, including:

- Purpose of rfsdelete
- Before running rfsdelete
- How to run rfsdelete
- After running rfsdelete
- Format of rfsdelete
- Examples



Purpose of rfsdelete

The *rfsdelete utility* deletes the StorHouse collections identified by the rfsmaint utility. Specifically, this utility:

- Reads the StorHouse table (_D table) containing the list of StorHouse collections to be deleted
- Marks the collections in the primary, or resident, directory as deleted

You also have the option of deleting collection copies in the archive and backup directories.

Before running rfsdelete

Before running rfsdelete, you must run the rfsmaint utility and the metadata backup utility. You can run the rfsmaint utility any number of times before running rfsdelete. For example:

Run rfsmaint.

Run rfsmaint.

...

Run metadata backup.

Run rfsdelete.

Run rfsmaint.

...



Refer to the *StorHouse Database Administration Guide* or the *StorHouse/RFS Administration Guide* for more information about running a metadata backup. Refer to the *StorHouse Database Administration Guide* for more information about running a journal archive utility.

How to run rfsdelete

You run the rfsdelete utility at the StorHouse operating system prompt, or on the StorHouse/RFS server platform, or from any system that can access the StorHouse/RFS configuration file or a copy of it and that has ODBC and TCP/IP connectivity to StorHouse.

▼ To run the rfsdelete utility at the StorHouse operating system prompt

1. Log in to the StorHouse system using the `operator` account and password.
2. At the StorHouse operating system prompt, type the utility command, and press `Enter`. See “Format of rfsdelete” on page 4-5 for more information about the utility command.

▼ To run the rfsdelete utility on a Windows StorHouse/RFS server

1. Log in to the StorHouse/RFS server platform with the administrator account.
2. From the **Start** menu, select **Run**.
3. Enter `CMD` and click **OK**.



Chapter 4 - The rfsdelete utility

4. In the DOS box enter the drive letter where StorHouse/RFS is installed, for example:

```
C:
```

5. Change to the StorHouse/RFS executable directory, for example:

```
CD "\Program Files\RFS\v4r0"
```

The quotes are necessary if StorHouse/RFS is installed in the Program Files directory.

6. Enter the rfsdelete command. See “Format of rfsdelete” on page 4-5 for more information about the utility command.

▼ To run the rfsdelete utility on a UNIX StorHouse/RFS server

1. If necessary, log in to the StorHouse/RFS server as root. You should be running bourne shell.
2. Execute the following command to get the necessary definitions.

```
. /rfs/v4r0/bin/defs
```

3. Enter the rfsdelete command. See “Format of rfsdelete” on page 4-5 for more information about the utility command.

After running rfsdelete

After running rfsdelete, examine the log file located in the default directory or the alternate directory (if specified on the command line) to ensure that the utility ran successfully.

After the rfsdelete utility marks StorHouse files for deletion, you must issue a StorHouse REMOVE command to release the space used by the



deleted collections. You do this using StorHouse/Control Center or the StorHouse Interactive Interface. You can also schedule the REMOVE command to run periodically. Refer to the StorHouse *Command Language Reference Manual* or the *StorHouse/Admin System Administrator's Quick Reference* for more information about removing StorHouse files.

Format of rfsdelete

The rfsdelete utility consists of two required and two optional options.

```
rfsdelete -c <rfs config file name> -d <storage or
collection definition name> [-m] [-a] [-b]
[-w <log directory>]
```

Table 3-1: Options/Arguments for the rfsdelete utility

Option/Argument	Description
-c <rfs config file name>	(required) The fully qualified path of the StorHouse/RFS configuration file that contains the storage definition or collection definition with the information required to connect to StorHouse and perform the necessary StorHouse tasks. The configuration file may be on the machine running the rfsdelete utility or on a machine running StorHouse/RFS that the rfsdelete utility can access, or just a copy of the StorHouse/RFS configuration file with the relevant information. Example: -c c:\Program Files\RFS\v4r0\rfs.cfg
-d <storage or collection definition name>	(required) The name of the storage definition or collection definition identifying the StorHouse database name, table name, system definition name, account ID, and password. For StorHouse/RFS 3.0, specify the collection name. For StorHouse/RFS 4.0, specify the storage definition name. Example: -d ST_RFS31



Chapter 4 - The rfsdelete utility

Table 3-1: Options/Arguments for the rfsdelete utility

Option/Argument	Description
-m	(optional) An option to also delete collections on the mirror system specified in the StorHouse/RFS configuration file (identified by the -d parameter). If you omit the -m parameter, the rfsdelete utility deletes collections on the primary system only.
-a	(optional) An option to delete collections in the archive directory. No error is reported if collections do not exist in the archive directory. If you run the rfsdelete utility without this option, then you must manually delete any collections in the archive directory.
-b	(optional) An option to delete collections in the backup directory. No error is reported if collections do not exist in the backup directory. If you run the rfsdelete utility without this option, then you must manually delete any collections in the backup directory.
-w <log directory>	(optional) The name of an alternate directory where the log file will be written. This value is case sensitive. If you omit the -w option, the default location is the same directory where the rfsdelete program resides.

Examples

This section contains two examples of the rfsdelete utility. The examples use the StorHouse/RFS configuration file located in `c:\Program Files\RFS\v4r0\rfs.cfg` and the collection definition named `ST_RFS31`.

- Delete collections in the primary, archive, and backup directories and write the log to the default directory:

```
rfsdelete -c c:\Program Files\RFS\v4r0\rfs.cfg  
-d ST_RFS31 -a -b
```




- Delete collections in the primary directory on both the primary and mirror StorHouse systems (-m) and write the log to `c:\Program Files\RFS\v4r0\deletelog`

```
rfsdelete -c c:\Program Files\RFS\v4r0\rfs.cfg  
-d ST_RFS31 -m -w c:\Program Files\RFS\v4r0\deletelog
```



Chapter 4 - The rfsdelete utility

C H A P T E R

5



The checkfile utility

This chapter describes the checkfile utility, including:

- Purpose of checkfile
- Before running checkfile
- How to run checkfile
- After running checkfile
- Format of checkfile
- Examples



Purpose of checkfile

When StorHouse/RFS detects a CRC error in a user file, it isolates the file by renaming it into an RFS_ISOLATED directory at the same level as the staging directory. The *checkfile utility* determines where in an isolated file the error was detected and creates a log identifying the beginning and ending addresses of the 2048 byte block(s) where the CRC error(s) was detected. The checkfile utility can optionally place the original user file, with errors, in a specified directory. The file creator can then examine the file for errors.

Before running checkfile

Before running checkfile, go to the RFS_ISOLATED directory to obtain the names of the files with CRC errors. You need to provide the file name as input to the checkfile utility. If the file has not yet been collected, then it will be in the same path as it was originally created but in the RFS_ISOLATED directory. For example, an uncollected file named `/a/b/c/d.txt` with a detected CRC error would be found in the following directory:

```
/RFS_ISOLATED/a/b/c/d.txt
```

If the file has been collected and the error occurred prior to writing the data to StorHouse, the isolated path includes the collection name and the logical block number where the data would normally reside in the collection. For example, if the above file had been collected and started at logical block 123 in the collection, then the isolated file name might look like this:

```
/RFS_ISOLATED/COL20051118084517(C).SYSNAME/a/b/c/d.txt.123
```



How to run checkfile

You run the checkfile utility on the system where StorHouse/RFS is installed.

▼ To run checkfile on Windows

1. Log in to the StorHouse/RFS server platform with the administrator account.
2. From the Start menu, select Run.
3. Enter `CMD` and click OK.
4. In the **DOS** box enter the drive letter where StorHouse/RFS is installed, for example:

`C:`

5. Change to the StorHouse/RFS executable directory, for example:

```
CD "\Program Files\RFS\v4r0"
```

The quotes are necessary if StorHouse/RFS is installed in the Program Files directory.

6. Enter the checkfile command. See “Format of checkfile” on page 5-5 and “Examples” on page 5-6 for more information.

▼ To run checkfile on UNIX

1. If necessary, log in to the StorHouse/RFS server as root. You should be running bourne shell.
2. Execute the following command to get the necessary definitions.

```
. /rfs/v4r0/bin/defs
```



3. Enter the checkfile command. See “Format of checkfile” on page 5-5 and “Examples” on page 5-6 for more information.

After running checkfile

After running the checkfile utility, access the log that identifies the location of the CRC errors. The log is located in the same directory as the checkfile program. Below is an example of a checkfile log.

```
2005-11-22 15:24:45.043 I- 2440: *****
StorHouse/RFS *****

2005-11-22 15:24:45.043 I- 2440: *****
CheckFile utility *****

2005-11-22 15:24:45.043 I- 2440: Copyright(c) 2001-2005,
FileTek, Inc.

2005-11-22 15:24:45.043 I- 2440: As an Unpublished
Licensed Work. All rights reserved.

2005-11-22 15:24:45.043 I- 2440: CheckFile Version
1.0.0.1 Prerelease

2005-11-22 15:24:45.043 I- 2440: Computer name: dev-
elf.filetek.com

2005-11-22 15:24:45.043 I- 2440: - - - - -
- - - - -

2005-11-22 15:24:45.043 I- 2440: Checking source file:
d:\RFS_ISOLATED\RFSFILE_00062043.TXT.1

2005-11-22 15:24:45.043 I- 2440: Writing output to:
d:\crctest1.txt
```



```

2005-11-22 15:24:45.043 E- 2440: CRC error between
offsets 14336 and 16384

2005-11-22 15:24:45.043 E- 2440: CRC error between
offsets 38912 and 40960

2005-11-22 15:24:45.043 E- 2440: CRC error between
offsets 55296 and 57344

2005-11-22 15:24:45.043 I- 2440: --- CheckFile complete -
--

```

Additionally, if you specify a destination file name, the file creator can check the original file to determine whether the file itself contains errors or whether the error occurred in the bits of the stored CRC.

Format of checkfile

The checkfile utility consists of one required and one optional option/argument.

```
checkfile -s <source file> [-d <destination file>]
```

Table 3-1: Options/Arguments for the checkfile utility

Option/Argument	Description
-s <source file>	(required) The path and file name of the isolated file. Example: d:\RFS_ISOLATED\RFSFILE_00062043.TXT
-d <destination file>	(optional) A fully qualified file name where the checkfile utility writes a copy of the original file. Example: d:\filewithcrc.txt



Examples

The following Windows example checks the file named `d:\RFS_ISOLATED\RFSFILE_00062043.TXT` and copies the original user file, with errors, into `d:\filewithcrc.txt`.

```
checkfile -s d:\RFS_ISOLATED\RFSFILE_00062043.TXT  
-d d:\filewithcrc.txt
```

The following UNIX example checks the file named `/rfs/collectors/RFS_ISOLATED/compliance` and copies the original file into a file called `compliance` in the current directory.

```
$BIN/checkfile -s /rfs/collectors/RFS_ISOLATED/compliance  
-d compliance
```


I N D E X



Symbols

_C table 2-2, 3-2
_D table 3-4
_S table 2-2

A

account privileges 2-3
account, operator 1-3
aliases table 2-2
arguments 1-3, 1-4

B

-b option, rfsmaint 3-10

C

checkfile utility
 -d option 5-5
 examples 5-6
 format 5-5
 -s option 5-5
collections table 2-2, 3-2

compaction 3-4
conventions
 command format 1-3
 document viii

D

-d option
 checkfile utility 5-5
 rfsdelete utility 4-5
database, StorHouse 2-6
deletion 3-4
destination file 5-5
DNS 2-5
document conventions viii

E

E- message 1-5
examples
 checkfile utility 5-6
 rfsdelete utility 4-6
 rfsmaint utility 3-12
 tblgen utility 2-8



Index

F

- f option, rfsmaint utility 3-11
- file locator tables 3-2
- format conventions 1-3
- formats
 - checkfile utility 5-5
 - date in rfsmaint utility 3-10
 - rfsdelete utility 4-5
 - rfsmaint utility 3-8
 - tblgen utility 2-5

I

- identifying collections eligible for deletion or compaction 3-3

L

- l option, rfsmaint utility 3-12
- log files 1-5
- logging in to StorHouse operating system 3-7, 4-3

M

- m option, rfsmaint utility 3-11
- metadata backup 4-2

O

- o option, rfsmaint utility 3-12
- operator account 1-3
- options 1-3, 1-4
- orphan collections 3-5

P

- p option, rfsmaint utility 3-10
- privileges, account 2-3

R

- r option, rfsmaint utility 3-11
- REMOVE command 4-4
- report mode, rfsmaint utility 3-6
- retention 3-3
- rfsdelete utility
 - d option 4-5
 - examples 4-6
 - format 4-5
 - purpose 4-2
 - w option 4-6
- rfsmaint utility
 - b option 3-10
 - compacting collections 3-4
 - criteria 3-3
 - examples 3-12
 - f option 3-11
 - format 3-8
 - functions 3-2
 - l option 3-12
 - m option 3-11
 - o option 3-12
 - p option 3-10
 - preparing collections for deletion 3-4
 - purpose 3-2
 - r option 3-11
 - removing orphan collections 3-5
 - report mode 3-6
 - s option 3-9
 - u option 3-10
 - w option 3-12
 - x option 3-11
- rfsrestore utility 1-2

S

- s option
 - checkfile utility 5-5



- rfsmaint utility 3-9
- security table 2-2
- source file 5-5
- statistics tables 2-2
- StorHouse vii
- StorHouse database 2-6
- StorHouse/RFS vii

T

- tblgen utility
 - after running 2-5
 - before running 2-3
 - example 2-8
 - how to run 2-3
 - log messages 2-10
 - prompts and responses 2-5
 - purpose 2-2

U

- u option, rfsmaint utility 3-10

W

- w option
 - rfsdelete utility 4-6
 - rfsmaint utility 3-12

X

- x option, rfsmaint utility 3-11