

Disk File Optimizer for OpenVMS

Guide to Operations

Operating System and Version: VSI OpenVMS IA-64 Version 8.4-1H1 or higher
VSI OpenVMS Alpha Version 8.4-2L1 or higher

Disk File Optimizer for OpenVMS Guide to Operations



VMS Software

Copyright © 2026 VMS Software, Inc. (VSI), Boston, Massachusetts, USA

Legal Notice

Confidential computer software. Valid license from VSI required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

The information contained herein is subject to change without notice. The only warranties for VSI products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. VSI shall not be liable for technical or editorial errors or omissions contained herein.

HPE, HPE Integrity, HPE Alpha, and HPE Proliant are trademarks or registered trademarks of Hewlett Packard Enterprise.

Table of Contents

Preface	vii
1. About VSI	vii
2. Purpose of This Guide	vii
3. Intended Audience	vii
4. Structure of This Guide	vii
5. Related Documents	viii
6. References to VSI Products	viii
7. OpenVMS Documentation	viii
8. VSI Encourages Your Comments	viii
9. Conventions	ix
Chapter 1. Overview of the Disk File Optimizer	1
1.1. Introduction	1
1.2. Product Description	1
1.2.1. Operational Goals	1
1.2.2. Features	1
1.2.3. User Interface	2
1.2.4. Programming Interface	5
1.2.5. Operational Overview	5
1.2.5.1. Defragmentation Process	5
1.2.5.2. Prologue and Epilogue Files	6
1.2.5.3. Operational Summary	7
1.2.6. Scheduling Defrag PLUS	7
1.2.6.1. Scheduling Database and Database Interface	7
1.2.6.2. Scheduler	7
1.2.6.3. Defragmentation Process	7
1.3. Basic File and File System Concepts	8
1.3.1. File Header Information	10
1.3.2. Extent Mapping Windows	11
1.3.3. Cathedral Window	11
1.4. Determining File Fragmentation	12
1.4.1. DEFRAGMENT SHOW {volume}/VOLUME Command	12
1.4.2. DUMP Utility	12
1.4.3. Monitor Utility	12
1.4.4. Performance Products	12
1.5. File Defragmentation and VMS Backup	13
Chapter 2. Getting Started	15
2.1. Introduction	15
2.2. Organizing Volumes Efficiently	15
2.2.1. /CLUSTER_SIZE=n	15
2.2.2. /DIRECTORIES=n	15
2.2.3. /HEADERS=n	15
2.2.4. /INDEX=MIDDLE	16
2.3. Improving I/O Performance	16
2.3.1. Move Log Files to One Volume	16
2.3.2. SET VOLUME/NOHIGHWATER_MARKING	16
2.3.3. SET VOLUME/DATA_CHECK=(NOREAD,NOWRITE)	16
2.3.4. SET VOLUME/NOERASE_ON_DELETE	17
2.4. Disk File Optimizer Licensing Considerations	17
2.5. VMScluster System Considerations	17

2.5.1. Processes Per Node	17
2.5.2. Common Scheduling Database	18
2.5.3. Locally Mounted Volumes	18
2.6. Creating Scheduling Scripts	18
2.6.1. Gathering Initial Information	19
2.6.2. Creating Scheduling Scripts for Defrag PLUS	19
2.6.3. Modifying Scheduling Scripts for DEFrag Plus	20
2.6.4. Creating Scheduling Scripts for Defrag	21
2.6.5. Checking for the Right Script Interval	21
2.6.6. Modifying Scheduling Scripts	22
2.7. Example Scheduling Scripts for a VMScluster	22
2.8. Creating Scheduling Scripts for a Standalone System	26
Chapter 3. Using Defrag	29
3.1. Introduction	29
3.2. Defrag Capabilities	29
3.2.1. Interface	29
3.2.2. Graphic User Interface	29
3.2.3. Online Help	30
3.2.4. Speed and Effectiveness of Defragmentation Operations	30
3.2.5. Scheduling Defragmentation Operations	30
3.2.5.1. Scheduling Guidelines	30
3.2.5.2. Using DECscheduler	31
3.2.6. Using Hot-files Information	31
3.2.7. Defining Quotas for the Defragmentation Process	31
3.2.7.1. Hard coded values	32
3.2.7.2. SYSGEN Process Quotas	32
3.2.7.3. DFO Logical Process Quotas	33
3.2.7.4. Defining the Transfer Size Logical	33
3.3. Wildcard Character Use	34
3.4. Examples Using Defrag	34
3.4.1. Creating Log Files	34
3.4.2. Showing the State of a Disk	35
3.5. Defrag Command Set	35
3.5.1. The Defrag Commands	36
Chapter 4. Using Defrag PLUS	61
4.1. Introduction	61
4.2. General Guidelines	61
4.2.1. DCL Interface	61
4.2.2. Wildcard Use	61
4.2.3. Graphic User Interface	62
4.2.4. Online Help	62
4.2.5. Quickness and Effectiveness of Defragmentation	62
4.2.6. Cluster Management	62
4.3. Scripts	63
4.3.1. Creating and Using Scripts	63
4.3.2. Script Naming Conventions	64
4.3.3. Examples of Creating and Using Scripts	64
4.3.3.1. Creating Log Files	64
4.3.3.2. Showing the State of a Disk	65
4.3.3.3. Creating a Named, Periodic Script	65
4.3.3.4. Copying Scripts	65

4.3.3.5. Modifying Scripts	66
4.3.3.6. Linking and Modifying Linked Scripts	66
4.3.3.7. Saving Database Contents	67
4.3.3.8. Re-creating the Database	67
4.3.4. Using Hotfiles Information	67
4.3.5. Defining Quotas for the Defragmentation Process	68
4.3.5.1. Hard coded values	68
4.3.5.2. SYSGEN Process Quotas	69
4.3.5.3. DFO Logical Process Quotas	70
4.3.5.4. Defining the Transfer Size Logical	70
4.4. Scheduling Defragmentation Operations	71
4.5. Error Recovery Procedures	71
4.6. Disk File Optimizer Command Set	71
4.6.1. Defrag PLUS Commands	73
Chapter 5. The DECwindows Interface for Defrag	121
5.1. Introduction	121
5.2. File Menu	121
5.3. Optimize Menu	122
5.3.1. Optimize Volume	122
5.3.1.1. Define the Optimize Parameters	123
5.3.1.2. Restrict Special Files	128
5.3.1.3. Schedule Optimization	130
5.3.2. Optimize Selected Files	132
5.3.3. Optimize Volume Index File	132
5.3.4. Abort an Optimize Process	133
5.4. Monitor a Volume	134
5.4.1. Defragmentation Process Statistics Window	136
5.4.2. Full Volume Window	138
5.4.3. Partial Volume Window	140
5.5. Report on the State of a Volume	140
5.5.1. Graphics Format	141
5.5.2. Text Format	143
Chapter 6. The DECwindows Interface for Defrag Plus	147
6.1. Introduction	147
6.2. File Menu	147
6.3. Optimize Menu	148
6.4. Script Menu	149
6.4.1. Create Scripts	150
6.4.1.1. Define a Script	151
6.4.1.2. Restrict Special Files Dialog Box	156
6.4.1.3. Schedule a Script Dialog Box	159
6.4.2. Modify a Script	161
6.4.3. Remove a Script	162
6.4.4. Show a Script	162
6.4.5. Schedule a Script	164
6.4.6. Abort a Script	167
6.5. Monitor a Script or Volume	167
6.5.1. Defragmentation Process Statistics Window	169
6.5.2. Full Volume Window	171
6.5.3. Partial Volume Window	173
6.6. Report on the State of a Volume	173

6.6.1. Graphics Format	174
6.6.2. Text Format	176
6.7. Options Menu	179
6.7.1. Setting Process Limits	180
6.7.1.1. The Set Process Limits Procedure	181
6.7.1.2. Procedure for Removing Nodes	182
6.7.2. Showing Defragmentation Process Limits Procedure	182
Chapter 7. Callable Interface	185
7.1. Introduction	185
7.2. Using the Callable Interface	185
7.2.1. Including Files for DFG\$DEFRAGMENT and DFG\$RELEASE_MEMORY	185
7.2.2. Linking Files with DFG\$DEFRAGMENT and DFG\$RELEASE_MEMORY	185
7.3. DFG\$DEFRAGMENT Routine	185
7.4. DFG\$RELEASE_MEMORY Routine	215
Appendix A. Answers to Frequently Asked Questions	217
Appendix B. Disk File Optimizer Logicals and Files	223
B.1. Files Added to System by the Disk File Optimizer	223
B.2. Logical Names	224
Appendix C. DEFRAGMENT SHOW Command Report	227
C.1. Introduction	227
C.2. Output from the SHOW Command	227
C.3. Examples of the Output from the SHOW Command	228
C.3.1. System Volume	228
C.3.2. User Volume With System Log Files	230
C.3.3. User Volume	233
C.4. Calculation of Fragmentation Index	235
Appendix D. Brief Log File Example	237
Appendix E. Full Log File Example	239
Appendix F. Defragmentation Messages	245
F.1. General Message Information	245
F.1.1. Linked Messages	245
F.1.2. Recovery Procedures	245
F.1.3. Recovering from Command Entry Errors	246
F.1.4. Reporting System Software or Hardware Errors	246
F.2. Disk File Optimizer Messages	246

Preface

1. About VSI

VMS Software, Inc. (VSI) is an independent software company licensed by Hewlett Packard Enterprise to develop and support the OpenVMS operating system.

2. Purpose of This Guide

The *Disk File Optimizer for OpenVMS Guide to Operations* provides information on the Disk File Optimizer software. This guide also contains information about the DCL command, graphic user interfaces, and the callable interface, and list of the messages generated by the software.

3. Intended Audience

The *Disk File Optimizer for OpenVMS Guide to Operations* is intended for the system administrator. This guide also contains information that may be of interest to programmers who wish to use the callable interface.

4. Structure of This Guide

The following list presents the structure of this document:

- *Chapter 1, "Overview of the Disk File Optimizer"* provides an overview of file fragmentation, performance degradation due to file fragmentation, and how the Disk File Optimizer can correct these situations.
- *Chapter 2, "Getting Started"* provides information on getting started with the Disk File Optimizer.
- *Chapter 3, "Using Defrag"* explains how to use the Defrag variant of the Disk File Optimizer software.
- *Chapter 4, "Using Defrag PLUS"* explains how to use the Defrag PLUS variant of the Disk File Optimizer software.
- *Chapter 5, "The DECwindows Interface for Defrag"* describes how to use the Disk File Optimizer DECwindows interface to initiate, control, and end defragmentation operations.
- *Chapter 7, "Callable Interface"* describes the callable interface.
- *Appendix A, "Answers to Frequently Asked Questions"* contains frequently asked questions.
- *Appendix B, "Disk File Optimizer Logicals and Files"* provides a list of files and logicals created during a Disk File Optimizer software installation.
- *Appendix C, "DEFRAGMENT SHOW Command Report"* describes the report you receive from the SHOW command.
- *Appendix D, "Brief Log File Example"* shows an example of a brief log file.
- *Appendix E, "Full Log File Example"* shows an example of a full log file.

- *Appendix F, "Defragmentation Messages"* lists and describes the error messages generated by the Disk File Optimizer software.

5. Related Documents

- *VSI OpenVMS License Management Utility Guide*
- *VSI OpenVMS Performance Management Manual*
- *VSI OpenVMS DCL Dictionary*
- *VSI OpenVMS System Manager's Manual*
- *VSI OpenVMS System Management Utilities Reference Manual*
- *Guide to OpenVMS File Applications*
- *VSI OpenVMS Record Management Utilities Reference Manual*

6. References to VSI Products

The following VSI products are mentioned in this manual:

Product	Formal Product Name
Disk File Optimizer	Disk File Optimizer refers to the Disk File Optimizer software.
Defrag	Defrag refers to a variation of the Disk File Optimizer software that provides basic file and volume defragmentation capabilities.
Defrag PLUS	Defrag PLUS refers to the variation of the Disk File Optimizer software that supports all of the features of Defrag, plus scheduling and command storing capabilities.
VPA	VPA refers to the VAX Performance Advisor software.
DECps	DECps refers to the DEC Performance Solution software.
OpenVMS	OpenVMS refers to the OpenVMS VAX and OpenVMS Alpha operating systems.
DECwindows	Software interface for video displays

7. OpenVMS Documentation

The full VSI OpenVMS documentation set can be found on the VMS Software Documentation webpage at <https://docs.vmssoftware.com>.

8. VSI Encourages Your Comments

You may send comments or suggestions regarding this manual or any VSI document by sending electronic mail to the following Internet address: <docinfo@vmssoftware.com>. Users who have VSI OpenVMS support contracts through VSI can contact <support@vmssoftware.com> for help with this product.

9. Conventions

The following conventions are used in this manual:

Convention	Meaning
{ <i>parameter</i> }	In command formats, braces indicate required parameters. Do not enter the braces when you enter the parameter in a command.
[<i>parameter</i>]	In command formats, square brackets indicate optional parameters. Do not enter the brackets when you enter the optional parameters.
[]	In interactive procedures, square brackets denote default responses.
<i>italic text</i>	In examples and messages, italic text represents input that is unique for each system.
boldface text	Boldface text is used for emphasis.
Ctrl/ <i>x</i>	Control key sequence indicates that you hold down the Ctrl key while you press another character key (for example, Ctrl/W or Ctrl/C).
.	In examples, a vertical ellipsis represents the omission of data that the system displays in response to a command or data that a user enters.
file-spec[, ...]	In command formats, a horizontal ellipsis indicates that the preceding items can be repeated one or more times.

Chapter 1. Overview of the Disk File Optimizer

1.1. Introduction

This chapter introduces the Disk File Optimizer (DFO) software. It contains the following major sections:

- A brief introduction to the Disk File Optimizer software, including a description of the two variants and the available interfaces
- A discussion of basic file system concepts and file fragmentation
- A brief description of suggested methods for determining file and volume fragmentation

1.2. Product Description

The Disk File Optimizer software is an OpenVMS utility designed to reduce file fragmentation on a storage device while that device remains on line to other users. The utility does not prevent users from accessing their files while it is running. If a user tries to access a file while it is being defragmented, the defragmentation process releases the file to the user and moves on to other files that require defragmentation. The file defragmentation utility is compatible with any Files-11 structure level 2 (ODS-2) formatted device including single volumes, bound volume sets, stripe sets, and shadow sets. The file defragmentation utility is also compatible with ODS-5 formatted device on OpenVMS Alpha 7.3 and above.

Two variants of the Disk File Optimizer software are available:

- Defrag provides basic capabilities to defragment files and volumes.
- Defrag PLUS provides all the capabilities of Defrag, plus scheduling and database options.

1.2.1. Operational Goals

The Disk File Optimizer has three fundamental operational goals. These goals in order of importance are to:

1. Maintain absolute data integrity
2. Avoid negative impact on normal user operations
3. Reduce file fragmentation on the target volume

1.2.2. Features

Table 1.1, "Features" lists the features of the Disk File Optimizer.

Table 1.1. Features

Feature	Description
Safe defragmentation	Built-in safety measures to protect user data.

Feature	Description
Works on line	Defragmentation can occur at the same time a device is user accessible. Open files are not defragmented, and defragmentation operations are transparent to users.
Unattended operation	Defragmentation can occur without operator intervention.
VMSccluster compatibility	Works with any supported VMSccluster configuration.
Automatic scheduling	A full range of scheduling options are available in Defrag PLUS, including daily, weekly, and continuous operating modes. Defrag does not provide scheduling capabilities but can be used with DEC scheduler, if desired.
Defragmentation levels	A full range of defragmentation levels is available, from a volume analysis to a full defragmentation operation.

1.2.3. User Interface

Users interact with the file defragmentation utility using the DCL commands listed in *Table 1.2, "Disk File Optimizer Command Summary"* or the graphic user interface.

For details on using the Defrag variant of the software, see *Chapter 3, "Using Defrag"*. For details on using the Defrag PLUS variant of the software, see *Chapter 4, "Using Defrag PLUS"*. For details on using the DECwindows interface, see *Chapter 5, "The DECwindows Interface for Defrag"*.

The Disk File Optimizer interfaces allow users to perform the following operations:

- Start defragmentation operations
- Stop defragmentation operations
- Schedule defragmentation operations (Defrag PLUS only)
- Monitor defragmentation operations
- Control CPU and I/O resources consumed by the defragmentation process
- Show database contents (Defrag PLUS only)
- Show volume fragmentation

Once the operator sets a defragmentation schedule in Defrag PLUS, no further operator intervention is required. Schedules can be set for any periodic interval; some volumes or files may be defragmented weekly, others biweekly, and so forth.

Table 1.2. Disk File Optimizer Command Summary

Command	Variant	Description
DEFRAGMENT ABORT {script-name[,...] volume-name[,...]} [/QUALIFIERS...]	Both	Aborts the defragmentation process associated with the specified volume (Defrag) or script (Defrag PLUS). Before exiting, the defragmentation process performs an orderly shutdown of all outstanding activities.

Command	Variant	Description
DEFRAGMENT ATTACH {process_name} [/QUALIFIERS...]	Both	Transfers control from the current process (which goes into hibernation) to the specified process.
DEFRAGMENT CANCEL {script-name[,...]} [/QUALIFIERS...]	Defrag PLUS	Cancels the next scheduled execution of the specified script. Normal scheduling resumes thereafter.
DEFRAGMENT DISABLE {script-name[,...]} [/QUALIFIERS...]	Defrag PLUS	Disables the specified script. A script does not execute while it is disabled. The script remains disabled until it is enabled with the ENABLE command.
DEFRAGMENT ENABLE {script-name[,...]} [/QUALIFIERS...]	Defrag PLUS	Enables a previously disabled script. Once enabled, a script operates normally (as scheduled).
DEFRAGMENT EXIT	Both	Returns the user from the Disk File Optimizer utility to the OpenVMS format.
DEFRAGMENT FILE {file_name[,...]} [/QUALIFIERS...]	Both	Allows the user to defragment a selected file (or files). Different qualifiers are available for Defrag and Defrag PLUS. If you have installed Defrag, the DEFRAGMENT FILE command creates a detached process that defragments the target file or files. This process executes immediately if files are available. If you have installed Defrag PLUS, the DEFRAGMENT FILE command creates and enters a script into the scheduling database. This script defragments the file or files specified on the command line. By default, the script is immediately eligible for execution, and once complete, is removed from the scheduling database.
DEFRAGMENT HELP {command}	Both	Invokes the standard VMS HELP utility for the appropriate version of the Disk File

Command	Variant	Description
		Optimizer software on your system.
DEFRAGMENT MODIFY {script-name} [/QUALIFIERS...]	Defrag PLUS	Allows the user to modify an existing script. This command supersedes use of the /INHERIT qualifier for modifying scripts.
DEFRAGMENT MONITOR {script-name volume-name} [/QUALIFIERS...]	Both	Invokes the defragmentation monitor, which monitors an ongoing defragmentation process. For Defrag, the input parameter must be a volume name. For Defrag PLUS, the input parameter can be either a script or a volume name, depending on the setting of the /VOLUME qualifier.
DEFRAGMENT OFFLINE_VOLUME {volume-name[,...]} [/QUALIFIERS...]	Both	Defragments INDEXF.SYS on the specified volume according to the selected command line qualifiers.
DEFRAGMENT REMOVE {script-name[,...]} [/QUALIFIERS...]	Defrag PLUS	Removes the specified script from the scheduling and parameter database.
DEFRAGMENT SET {(parameter=value[,...])} [/NODE=(node[,...])]	Defrag PLUS	Sets the specified parameters (CPU_LIMIT, IO_LIMIT, PROCESS_LIMIT) to the values specified.
DEFRAGMENT SHOW [{script-name[,...]} {volume-name[,...]}] [/QUALIFIERS...]	Both	Shows the contents of the scheduling and parameter database (Defrag PLUS) or the fragmentation state of the supplied volumes.
DEFRAGMENT SPAWN {command}	Both	Creates a subprocess and either transfers control to the subprocess or executes a DCL command from within the subprocess.
DEFRAGMENT START {script-name[,...]} [/QUALIFIERS]	Defrag PLUS	Starts the specified script at the time specified by the /AFTER qualifier. By default, the script starts immediately.
DEFRAGMENT VOLUME {volume_name[,...]} [/QUALIFIERS...]	Both	Allows the user to defragment a selected volume (or volumes). Different qualifiers are available for Defrag and Defrag PLUS.

Command	Variant	Description
		If you have installed Defrag, the <code>DEFRAGMENT VOLUME</code> command creates a detached process that defragments the target volume. This process executes immediately. If you have installed Defrag PLUS, the <code>DEFRAGMENT VOLUME</code> command creates and enters a script into the scheduling database. This script defragments the volume associated with the volume parameter. By default, the script is immediately eligible for execution, and once complete, is removed from the scheduling database.

1.2.4. Programming Interface

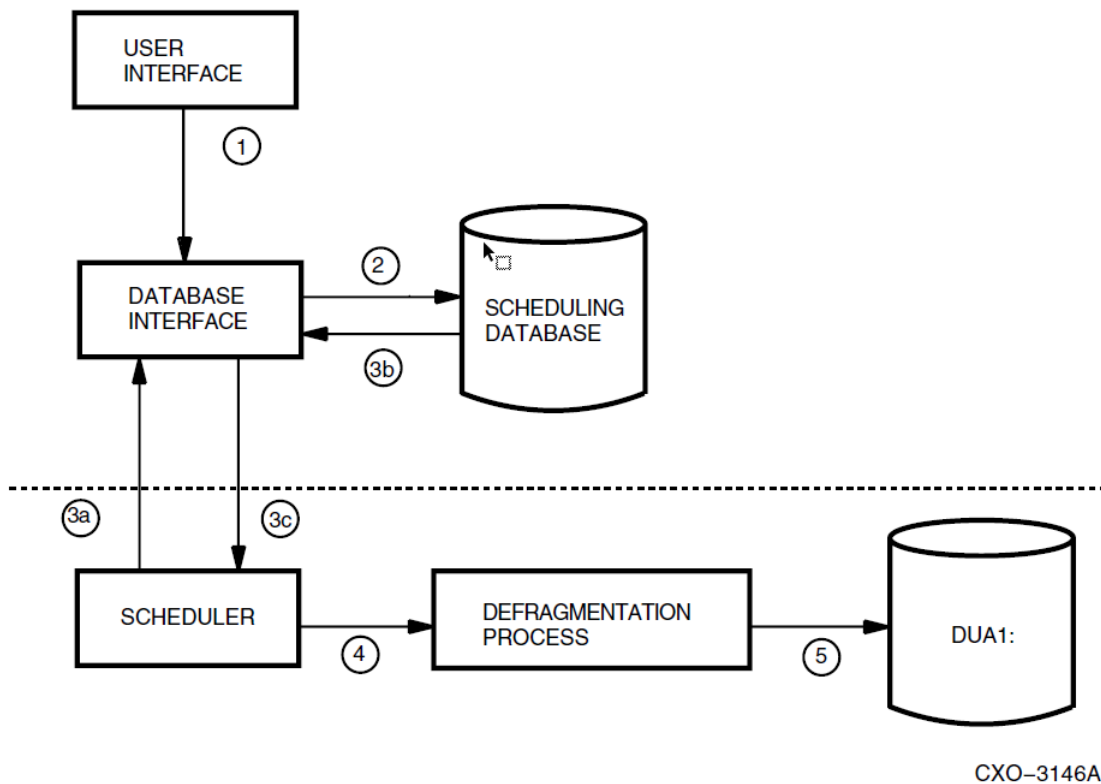
The Disk File Optimizer software also provides the opportunity for programmers to call for defragmentation operations to occur directly from their application programs. For detailed information about the callable interface, see *Chapter 7, "Callable Interface"*.

1.2.5. Operational Overview

This section presents an operational overview of the file defragmentation process implemented by the Disk File Optimizer software.

1.2.5.1. Defragmentation Process

Figure 1.1, "Command Operation" illustrates the general organization of the Disk File Optimizer software.

Figure 1.1. Command Operation

The numbers on the illustration correspond to the numbers in the following list:

1. The user enters the DEFRAGMENT FILE or DEFRAGMENT VOLUME command. At this point, if you are running the Defrag variant of the software, go directly to Step 5.
2. The database interface stores the command information in the database, notifies the schedulers if appropriate, and returns the user to the operating system.
3. A separate, autonomous scheduling process reads the stored information using the database interface and determines when the volume or file should be defragmented.
4. At the appropriate time, the scheduler starts a defragmentation process that is dedicated to defragmenting the target volume or file.
5. The defragmentation process defragments the target volume or file. For the example shown in *Figure 1.1, "Command Operation"*, the target volume is DUA1.

1.2.5.2. Prologue and Epilogue Files

Under some circumstances, you want to perform some special processing before and after files undergo the defragmentation process. Using the /PROLOGUE and /EPILOGUE qualifiers with DCL command files, you can:

- Check the status of the disk to ensure defragmentation is needed
- Take a snapshot of files or disks before and after the defragmentation process to monitor results
- Shut down a process that opens a file before defragmentation and start it up after defragmentation

1.2.5.3. Operational Summary

The Defrag process provides a very flexible combination of scheduling and operational characteristics. For example, defragmentation operations can occur one time only, can execute on a periodic schedule, can be limited to certain nodes in a cluster, or can execute on any node in the cluster. You can control the CPU and I/O resources the defragmentation processes use on a node-by-node basis. The dashed line in *Figure 1.1, "Command Operation"* indicates a potential node boundary. The actions above the line and those below the line can occur on different nodes in a cluster.¹

1.2.6. Scheduling Defrag PLUS

This section explains how the scheduling and database functions for the Defrag PLUS variant of the software work.

1.2.6.1. Scheduling Database and Database Interface

The scheduling database stores DCL command lines. These stored command lines are called *scripts*. Almost every script is available for execution at any time.²

Along with the script information, the database stores scheduling information in a time-ordered queue. The scheduler uses this information to determine when to start a given script. Starting the script initiates the defragmentation process associated with that script.

1.2.6.2. Scheduler

The scheduler runs as a detached process. A separate scheduler process is required for each node that executes the defragmentation process. Thus, if all defragmentation is done from NODE1, only one scheduler is required and it must execute on NODE1. Conversely, if the defragmentation work is spread across multiple nodes in a cluster, multiple scheduler processes are required: one on each node that executes the defragmentation process.

The scheduler process spends most of its time in the LEF state. It comes out of the LEF state if any of the following events occur:

- A command is entered that requires scheduler functions
- An existing defragmentation process completes
- A scheduled event occurs

If a script is scheduled to execute, and it may execute on the same node as the scheduler, the scheduler initiates the defragmentation process.

1.2.6.3. Defragmentation Process

The defragmentation process is an independent, detached process started by the scheduler. It is dedicated to defragmenting the target volume or files. The defragmentation process:

- Provides the interface to the monitor

¹The database and database interface are actually shared among the nodes in a cluster.

²Scripts that are not named(/SCRIPT) and that are not part of a linked set of scripts are removed from the database when they complete.

- Aborts itself
- Throttles itself
- Executes preprocessing and post processing files
- Time limits itself
- Defragments files
- Consolidates free space
- Arranges files according to access patterns

After initialization, the defragmentation process performs its tasks in the following phases:

1. File evaluation—Determines which files should be defragmented and which files are dormant, frequent, excluded, and so on
2. File defragmentation phase one — defragment those files that can be easily defragmented
3. File defragmentation phase two — defragment files using a best fit algorithm (For example: defragment a file with 30 extents to 5 extents.)
4. File defragmentation phase three — defragment those files that could not be defragmented in phase one or phase two
5. Frequent file placement — arrange those files in the frequent list and hot file database together between INDEXF.SYS and LBN 0
6. Dormant file placement — arrange those files in the dormant list together between the high LBN and INDEXF.SYS
7. Free space consolidation—Consolidates free space

If all files have not been defragmented in this first pass because free space was not available, the phases are repeated again as needed.

1.3. Basic File and File System Concepts

A file is a body of data. The file system uses some of the body of data strictly to manipulate the file (file metadata). The rest of the data comprises the information the user stores in the file. In the OpenVMS File Control Program (FCP), the file metadata is stored in the file header, directory entries, and in-memory data structures. The user data area is composed of virtual blocks of 512 bytes.

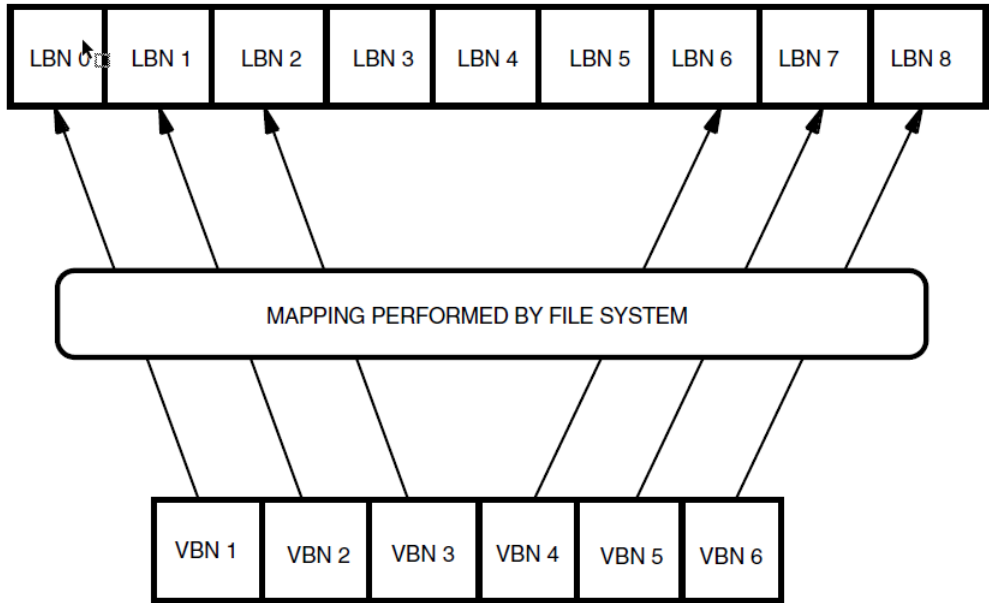
The virtual blocks that make up the user data area of a file are always contiguous and are numbered from 1 to n , where n is the number of virtual blocks in the file. These numbers are called virtual block numbers (VBNs). One of the primary functions of the FCP is to map these virtual blocks to their corresponding logical blocks on a volume. Like virtual blocks, logical blocks are 512 bytes. Unlike virtual blocks, logical blocks are numbered from 0 to $n-1$ and the numbers are called logical block numbers (LBNs). The logical blocks that make up a file do not have to be contiguous.

As more and more files are manipulated on a volume, there are fewer available contiguous logical blocks. Free space may be fragmented into many small areas. Eventually, the FCP creates or extends a file so

that portions of it reside on different parts of the disk. This is file fragmentation, a simple example of which is shown in *Figure 1.2, "File Fragmentation Example"*. In general, the effects of file fragmentation are not appreciable until the allocated space on a volume reaches 70 to 80 percent of the available space.

Figure 1.2. File Fragmentation Example

PORTION OF DEVICE LBN RANGE



FILE VBN RANGE

CXO-3147A

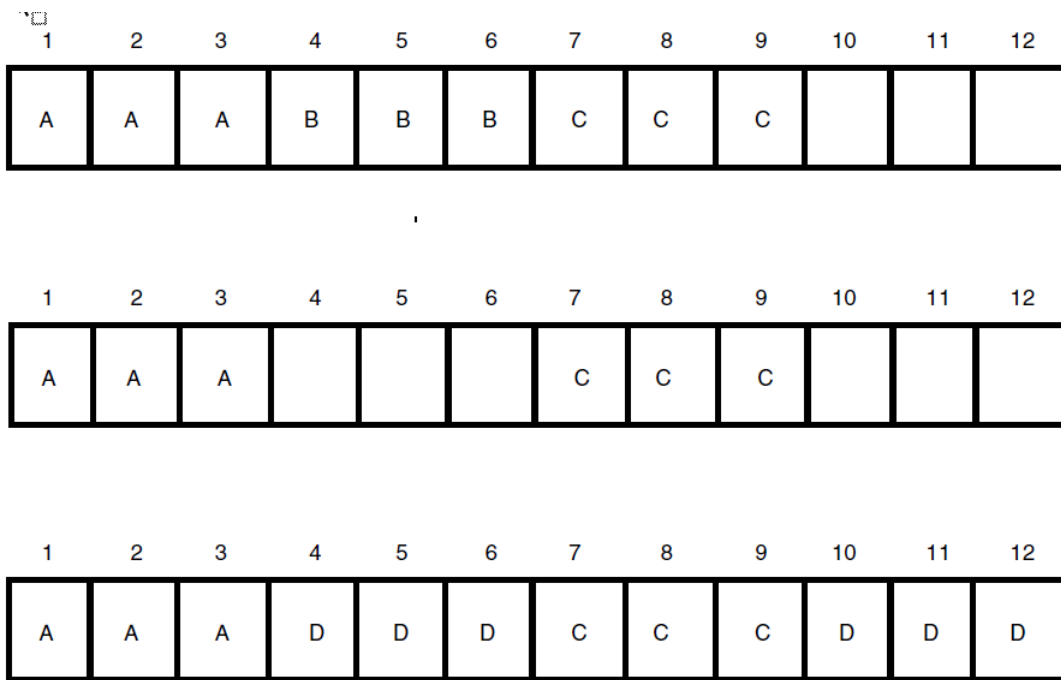
The FCP allocates and extends files in units called *clusters*. Each cluster contains one or more logical blocks.³ The number of logical blocks in a cluster is called the volume cluster factor.

The FCP allocates disk blocks by sequentially searching the extent cache for available free space and using the free space as it is available. If the extent cache dwindles, it is refilled from the bit map stored on each disk. In addition, blocks that are returned because of file deletion or file truncation are returned to the extent cache.

Consider a newly initialized disk that has 12 contiguous clusters available for allocation. Users A, B, and C each allocates and writes a three-cluster file. User A allocates clusters 1–3 by default, because that is what resides in the extent cache. User B allocates clusters 4–6 and user C allocates clusters 7–9. This is shown in the first part of *Figure 1.3, "Allocating Fragmented Files"*. User B then deletes the file and the clusters are returned to the extent cache. Thus, the extent cache now indicates that clusters 1–3 and 7–9 are allocated as shown in the second part of *Figure 1.3, "Allocating Fragmented Files"*.

Now suppose another user, user D, allocates a six-cluster file. Because the default file allocation algorithm is to sequentially search the extent cache and use free space as it is available, user D receives clusters 4–6 and 10–12. Blocks 7–9 are already allocated. This is shown in the last part of *Figure 1.3, "Allocating Fragmented Files"*.

³The default number of blocks in a cluster is three for volumes of 50,000 blocks or more, or one for volumes having less than 50,000 blocks.

Figure 1.3. Allocating Fragmented Files

CXO-3148A

1.3.1. File Header Information

A file header, which is physically separate from the file it describes, describes each file on a Files-11 volume. All file headers are contained in the INDEXF.SYS file located in the Master File Directory (MFD). The file header contains all the information necessary to access the file. The FCP, RMS, the Dump utility, and the Backup utility use the file header. The defragmentation process uses the file header to determine the state of the file before and after defragmentation. The file header contains the following areas:

- Header area—Header validation information, file ID, ownership and protection codes, and offsets to the other areas.
- Ident area—Primarily accounting information.
- Map area—List of retrieval pointers used to perform the VBN to LBN mapping.
- Access control list area—Primarily information about who can access this file.
- Reserved area—Not used by the FCP. It is available for use by user programs.
- Checksum—Helps determine whether the header data is valid.

Since the size of the file header is fixed, files with a great deal of mapping or access control information require more than one header. A file in this condition is called a *multiheader file* and is described by multiple file headers that are chained together. Multiheader files are typically large and fragmented and they may span multiple volumes on a volume set.

Of the various areas in the file header, the map area is especially important from a file fragmentation standpoint. This area shows how many pieces compose a file. It also shows the size of the pieces. The pieces themselves are contiguous and are called *extents*. A retrieval pointer, which in general contains a

starting LBN and a size field, describes each extent. The Dump utility can display the extents of a file, as shown in *Example 1.1, "Viewing Retrieval Pointers Using the Dump Utility"*.

Example 1.1. Viewing Retrieval Pointers Using the Dump Utility

```
$ DUMP/HEADER/BLOCKS=COUNT=0 FILE.EXT
:
Map area
  Retrieval pointers
    Count:      42      LBN:    1009767
    Count:    1347      LBN:    1009851
    Count:     30      LBN:    1011201
    Count:     33      LBN:    1011237
:
```

This example shows that FILE.EXT is fragmented into four extents, each described by a separate retrieval pointer. The first extent begins on LBN 1009767 and ends on LBN 1009808, a total of 42 LBNs. Thus, the location of the file's contents can be determined from the retrieval pointers found in the file header.

1.3.2. Extent Mapping Windows

Whether multiple extents cause a performance problem depends upon a number of variables, one of which is the extent mapping window for the given file. The extent mapping window is an in-memory structure that contains the same information as the on-disk retrieval pointers for a file. The size of the window determines the number of retrieval pointers that can be mapped concurrently. The default window size is seven, meaning seven extents can be mapped at the same time. You can use the following methods to set the window size:

- The DCL command INITIALIZE device-name /WINDOWS= *n*
- The DCL command MOUNT device-name /WINDOWS= *n*
- The FAB\$B_RTV field at file open time
- The FDL attributes FILE WINDOW_SIZE
- The system parameter ACP_WINDOW (if the disk is mounted /SYSTEM)

If all the retrieval pointers to a file cannot fit into the mapping window, the FCP may need to read additional retrieval pointers from disk to map a different portion of the file. This operation is called a *window turn*. A window turn requires one extra I/O and can, therefore, affect performance.

By default, the four extents of FILE.EXT shown in *Example 1.1, "Viewing Retrieval Pointers Using the Dump Utility"* can be mapped into memory at the same time. No additional I/O operation is required to find the location of any of the four extents.

1.3.3. Cathedral Window

In extreme cases of fragmentation, the file system can generate a special type of mapping window called a *cathedral window*. Basically, if the number of extents exceeds the window size, AND the user requests a file be mapped completely on OPEN, then the file system generates a cathedral window. The cathedral window maps the extents of an entire file regardless of the size, number of headers, number of mapping pointers, or size of the ACL for the file. It does not generate a window turn.

1.4. Determining File Fragmentation

The following methods can help you determine whether a particular volume or file is fragmented:

- Enter the following command:

```
DEFRAGMENT SHOW {volume-name}/VOLUME
```

- Enter a DCL DUMP command
- Use the VMS Monitor utility
- Use VAX Performance Advisor (VPA) software
- Use DEC Performance Solution (DECps) software

1.4.1. DEFRAGMENT SHOW {volume}/VOLUME Command

The DEFRAGMENT SHOW {volume}/VOLUME command can be used to display the fragmentation state of a given volume. Several options provide different levels of detail. By default, the SHOW subcommand provides a summary number called a fragmentation index that gives a rough idea of the fragmentation level for the volume. Along with the fragmentation index is a brief list of statistics about file and free space fragmentation that support the fragmentation index value. For additional information about using the SHOW subcommand, see *Chapter 3, "Using Defrag"* or *Chapter 4, "Using Defrag PLUS"*.

1.4.2. DUMP Utility

The Dump utility shows how many extents compose a file and the starting LBN for those extents. An example of using the Dump utility is given in *Example 1.1, "Viewing Retrieval Pointers Using the Dump Utility"*. For more information, see the *VMS DCL Dictionary*.

1.4.3. Monitor Utility

The Monitor utility displays various classes of information that show device I/O rates, file system cache statistics, and FCP statistics. The FCP class shows the window turn rate for the system. The window turn rate is not broken down by device, but a high window turn rate may indicate fragmentation. Use information from the VPA or DECps software to reinforce this indication. In addition, use the Monitor utility to see statistics about the file system caches on your system. For more information, see the *VMS Monitor Utility Manual*.

1.4.4. Performance Products

DECps is a follow-on to the VPA product. Either product can help you analyze the performance of your system. Using the DECps command ADVISE PERFORMANCE REPORT (or the VPA equivalent VPA ADVISE/REPORT), you can obtain information about disk split I/O rates which can indicate disk fragmentation.

You can use the /USE_HOTFILES_DATABASE qualifier on the DEFRAGMENT FILE or DEFRAGMENT VOLUME command to direct the defragmentation process to query the appropriate database (VPA or DECps) for a list of hotfiles (frequently accessed files). This list is automatically added

to the /FREQUENT files list. For more information about using VPA or DECps software to augment the frequent files list, see *Chapter 3, "Using Defrag"* or *Chapter 4, "Using Defrag PLUS"*.

1.5. File Defragmentation and VMS Backup

In the past, BACKUP/RESTORE was the only VSI-approved method to defragment a volume. Now, the Disk File Optimizer allows you to defragment a volume while that volume remains on line and accessible to other users—without operator intervention.

Note

The Disk File Optimizer does not replace the need to use the Backup utility to protect your data.

A fully defragmented disk significantly improves performance in the same way that using BACKUP and then RESTORE improves performance. Contiguous files require less time to access. Since the free space on the volume has also been consolidated, the time required to create new files is also decreased.

Chapter 2. Getting Started

2.1. Introduction

This chapter is for the new user of Disk File Optimizer(DFO) software. It includes:

1. Minimizing file fragmentation when initializing volumes
2. Improving I/O performance
3. Licensing DFO
4. Creating scripts
5. Scheduling defragmentation jobs
6. Creating scheduling scripts

2.2. Organizing Volumes Efficiently

You can keep file fragmentation to a minimum by using certain qualifiers with the volume INITIALIZE command. To maximize performance by keeping head movement on a volume to a minimum, the following files must be clustered around the center of the disk:

- INDEXF.SYS (controlled by volume INITIALIZE qualifiers)
- Frequent or hot files (/FREQUENT and /HOTFILE_DATABASE)
- 000000.dir (INITIALIZE qualifiers)

The following sections describes these qualifiers and their effect.

2.2.1. /CLUSTER_SIZE=n

Retaining log files on a volume initialized with a large (100 or so) cluster size can keep fragmentation to a minimum. System error and other log files are always extending and creating large, fragmented files. Also, keeping these normally open logfiles off the system volume allows the Disk File Optimizer software to consolidate free space more efficiently. The trade-off is more unused space in the last cluster of files.

2.2.2. /DIRECTORIES=n

The /DIRECTORIES=n qualifier is important if you are placing frequent files when defragmenting the volume. Frequent files are placed around the center of the volume. The 000000.DIR is also placed around the center of the volume by specifying /INDEX=MIDDLE when initializing the volume. However, if more than 16 files and top level directories are created in the [000000] directory, the directory will extend and possibly be allocated on another part of the volume. You can keep the 000000.DIR from being moved by specifying the approximate number of files that will be in this directory.

2.2.3. /HEADERS=n

The `/HEADERS=n` qualifier is important for I/O performance. `INDEXF.SYS` contains the file headers for all files on the volume. By default, when the volume is initialized it contains space for 16 file headers. When more than 16 files are created on the volume, it extends creating another extent. When the new extent fills up, it extends again. This extending fragments `INDEXF.SYS` and can cause poor I/O performance. By preallocating enough space for `INDEXF.SYS`, you can keep `INDEXF.SYS` from extending. One way to estimate the number of headers is by looking at existing disks that are the same type.

If you are placing frequent or hot files, using the `/HEADERS=n` qualifier could affect volume access. Frequent or hotfile placement is optimized by having `INDEXF.SYS` in the middle of the volume. By default or by specifying `/INDEX=MIDDLE`, the fourth extent of `INDEXF.SYS` is placed in the middle of the volume. If enough space is not preallocated for all of the file headers on the volume, `INDEXF.SYS` is extended and the new extent could be placed anywhere on the volume.

2.2.4. `/INDEX=MIDDLE`

The `/INDEX=MIDDLE` qualifier is important if you are placing frequent files when defragmenting the volume. Frequent file placement is optimized by having `INDEXF.SYS` in the middle of the volume.

2.3. Improving I/O Performance

You may be able to improve I/O performance by moving some files and setting the characteristics of your volumes. The following sections describe how you might improve I/O performance:

2.3.1. Move Log Files to One Volume

You can improve I/O performance on your system volume by moving log files to a dedicated volume. Usually the system volume has a small cluster size which causes the log files to have many small fragments. By initializing the dedicated volume with a large cluster size, you can keep the number of extents down. You are also taking some of the writes to the volume off the system volume.

By moving these large, open, fragmented files to another volume, you will see an improvement in DFO. It will not have to work around all of these large, open, fragmented files when consolidating free space.

2.3.2. `SET VOLUME/NOHIGHWATER_MARKING`

High-water marking guarantees that users cannot read data that was not written by them. Enabling high-water marking causes extra I/Os when accessing the volume. If you do not need this level of security, you can disable this feature. File high-water marking is enabled by default.

To check if a volume has high-water marking enabled, use the following command:

```
$ SHOW DEVICE/FULL volume_name:
```

The footing of the display will have the words: file high-watermarking. You can disable the high-water marking using the following command:

```
$ SET VOLUME/NOHIGHWATER_MARKING volume_name:
```

2.3.3. `SET VOLUME/ DATA_CHECK=(NOREAD,NOWRITE)`

The data check operation causes a read-after-read, are ad-after-write, or both. Enabling data check causes extra I/Os when accessing a file. The default is not to have data check enable. You can check it by using the following command:

```
$ SHOW DEVICE/FULL volume_name:
```

The header of the display will have the words: data check on reads and/or data check on writes. You can disable the data check using the following command:

```
$ SET VOLUME/DATA_CHECK=(NOREAD,NOWRITE) volume_name:
```

2.3.4. SET VOLUME/NOERASE_ON_DELETE

Erase on delete writes a system specific pattern over the file when the file is deleted. Enabling erase on delete causes extra I/Os when deleting a file. The default is not to have erase on delete enabled. You can check it by using the following command:

```
$ SHOW DEVICE/FULL volume_name:
```

The footing of the display will have the words: erase on delete. You can disable the erase on delete using the following command:

```
$ SET VOLUME/NOERASE_ON_DELETE volume_name:
```

2.4. Disk File Optimizer Licensing Considerations

Disk File Optimizer is furnished only under a license and it supports the OpenVMS License Management Facility. License units for the software are granted for use on one specified system, based on the processor system class. The system classes are Workgroup, Departmental, and Enterprise. For more licensing information, please refer to the Software Product Description, 55.85.xx.

When using DFO on a standalone system, the system must be licensed according to the class of CPU of that system. When using DFO in a cluster, only one node in the cluster must be licensed to run DFO. That one node can then defragment all the disks that are accessible within that cluster. For performance reasons, many customers have DFO licensed on multiple nodes within the cluster. This will speed up the defragment process for all the disks and spread the load among the processors.

2.5. VMSccluster System Considerations

When creating scheduling scripts, you may want to consider the impact of the following on your VMSccluster system:

- Processes per node
- Common scheduling database
- Locally mounted volumes

The following sections describe these considerations.

2.5.1. Processes Per Node

When setting up the scheduling database, you may want to limit the number of defragmentation processes on a node. Unless specified, a node can start an unlimited number of defragmentation processes.

To limit the number of defragmentation processes on a node, use the DEFAGMENT SET command:

```
$ DEFAGMENT SET PROCESS_LIMIT=2/NODE=node_name
```

You can specify different process limits for each node by specifying the /NODE qualifier. If you do not specify the /NODE qualifier, all nodes in the database are affected by the command.

To check the process limit parameter for the nodes in the database, use the DEFAGMENT SHOW command:

```
$ DEFAGMENT SHOW /PARAMETERS
```

2.5.2. Common Scheduling Database

Be sure to install the scheduling database on a volume common to all DFO licensed nodes in the cluster. A common volume must be mounted with the /CLUSTER qualifier. The default is to place the scheduling database on the system volume. This will work only if this volume is accessible to all nodes.

If you have already installed DFO and need to move the scheduling database to a more appropriate volume, use the following steps:

1. Stop the schedulers on all nodes in the cluster using the following command on each node:

```
$ @SYS$STARTUP:DFG$STOP_SCHEDULER
```

2. Create a directory on an appropriate volume.
3. Copy the files from the old directory to the new directory and delete the old files.
4. Edit SYS\$STARTUP:DFG\$STARTUP.COM and change the DFG\$DATABASE and DFG\$MAIL_ADDRESS logicals to point to the new directory:

```
$ CALL DEFINE_LOGICAL DFG$DATABASE new_volume:[new_directory]
$ CALL DEFINE_LOGICAL DFG$MAIL_ADDRESS new_volume:[new_directory]DFG
$MAIL.DIS
```

5. Restart the schedulers using the following command on each node of the cluster:

```
$ @SYS$STARTUP:DFG$START_SCHEDULER
```

2.5.3. Locally Mounted Volumes

For locally mounted volumes, you must consider license requirements and scheduling script creation. To defragment volumes that are locally mounted, you must have a DFO license for that node. Also, when creating a scheduling script for that volume, you must specify the node that the volume is mounted on. The following is an example of specifying the node:

```
$ DEFAGMENT VOLUME $1$DKA100:/NODE=LUNTIC/SCRIPT=DEFRAG_$1$DKA100
```

In the above example, the scheduler starts the defragmentation process on node LUNTIC.

2.6. Creating Scheduling Scripts

To schedule defragmentation jobs, you have to create scheduling scripts for Defrag PLUS or command procedures for Defrag. By creating scheduling scripts or command procedures, you define which volumes to defragment and how often to defragment them. The following examples deal with creating scheduling scripts for Defrag PLUS. You can still use the same commands to create command procedures for Defrag.

2.6.1. Gathering Initial Information

Before creating your scheduling scripts, you may want to record the following information:

- The fragmentation index for each volume. Use the following command for each volume:

```
$ DEFRAGMENT SHOW volume_name:/VOLUME
```

Appendix C, "DEFRAGMENT SHOW Command Report" describes how to read the report. You can use this information as a starting point to monitor how often you may want to defragment a particular volume.

- Which volumes you want to defragment on which nodes.

2.6.2. Creating Scheduling Scripts for Defrag PLUS

This section gives you examples of how to create scheduling scripts. *Chapter 4, "Using Defrag PLUS"* contains a complete description of each command.

The following shows the command to create a script for a locally mounted volume that is defragmented every night at 21:00.

```
$ DEFRAGMENT VOLUME $2$DKA100: -           ❶  
  /SCRIPT=DEFRAG_$2$DKA100 -                 ❷  
  /INTERVAL=1-00:00 -                         ❸  
  /AFTER=21:00 -                             ❹  
  /FULL -                                    ❺  
  /LOG=DFG$DATABASE:DEFRAG_$2$DKA100 -       ❻  
  /NODE=LUNTIC -                             ❼  
  /MAIL/NOOPCOM_NOTIFICATION                 ❽
```

The following describes each line in the above example:

- ❶ The defragment volume command with the volume to defragment, \$2\$DKA100:
- ❷ The script name, DEFRAG_\$2\$DKA100
- ❸ The interval between the start of script, EVERY DAY
- ❹ The time to start the script, 21:00
- ❺ The type of log to generate, FULL
- ❻ The name of the log file, DFG\$DATABASE:DEFRAG_\$2\$DKA100
- ❼ The name of the node to start the script on, LUNTIC
- ❽ The type of notification, MAIL and not OPCOM

The following shows the command to create a script for a volume that is defragmented every third night on any available node.

```
$ DEFRAGMENT VOLUME $1$DUA2: -  
  /SCRIPT=DEFRAG_$1$DUA2 -  
  /INTERVAL=3-00:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA2
```

The following shows the command to create a script to defragment INDEXF.SYS. The script does not have a schedule. You start it with the DEFRAGMENT START command after you have dismounted the volume.

```
$ DEFRAGMENT OFFLINE_VOLUME $1$DUA2: -  
  /SCRIPT=DEFRAG_OFFLINE_$1$DUA3 -  
  /NOAFTER
```

The easiest way to create scheduling scripts is with a DCL command procedure. This way, if anything changes, you can edit the command procedure and reload the scheduling database. Also, if something should happen to the database, you can create a new database and reload it.

The following example shows the above interactive examples put in one command procedure, **LOAD_DB.COM**:

```
$ DEFRAGMENT VOLUME $2$DKA100: -  
  /SCRIPT=DEFRAG_$2$DKA100 -  
  /INTERVAL=1-00:00 -  
  /AFTER=21:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$2$DKA100 -  
  /NODE=LUNTIC -  
  /MAIL/NOOPCOM_NOTIFICATION  
$ DEFRAGMENT VOLUME $1$DUA2: -  
  /SCRIPT=DEFRAG_$1$DUA2 -  
  /INTERVAL=3-00:00 -  
  /AFTER=17:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA2  
$ DEFRAGMENT OFFLINE_VOLUME $1$DUA2: -  
  /SCRIPT=DEFRAG_OFFLINE_$1$DUA3  
  /NOAFTER  
$ EXIT
```

After creating the **LOAD_DB.COM** command procedure, you can load the scheduling database with the following command:

```
$ @LOAD_DB
```

If you need to change the procedure, edit **LOAD_DB.COM** and make the changes, then remove the old scripts and reload them using the following commands:

```
$ DEFRAGMENT REMOVE *  
$ @LOAD_DB
```

You will have to wait 30 seconds or so between the above commands to allow DFO to delete the scripts from the database.

2.6.3. Modifying Scheduling Scripts for DEFRAG Plus

This sections shows you how to modify a script using the DEFRAGMENT MODIFY command. *Chapter 4, "Using Defrag PLUS"* contains complete descriptions of each command.

Use the following command to modify the DEFRAG_\$2\$DKA100 script to start at 17:00, disable mail, and also run on any of three nodes instead of one node:

```
$ DEFRAGMENT MODIFY DEFRAG_$2$DKA100 -  
  /AFTER=17:00 -  
  /NOMAIL -  
  /NODE=(LUNTIC,PRNOID,SENILE)
```

2.6.4. Creating Scheduling Scripts for Defrag

If you have installed the Defrag variant, you can create command procedures as scripts and schedule the procedure. You can create a command procedure for each volume or a command procedure for each node. The following is an example of command procedure that would defragment \$2\$DKA100:

```
$ DEFRAGMENT VOLUME $2$DKA100: -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$2$DKA100 -  
  /NOOPCOM_NOTIFICATION  
$ MAIL/NOSELF/SUB="VOLUME $2$DKA100 DEFRAGMENTED" NL: SYSTEM  
$ exit
```

2.6.5. Checking for the Right Script Interval

Now that you have created the scripts, you need to monitor the script interval. You may:

- Periodically issue the DEFRAGMENT SHOW command for each volume to ensure the fragmentation index is at an acceptable level.
- Create a program that gathers fragmentation information and creates a graph of the fragmentation of each volume.

To check the fragmentation index of the volume, use the following command:

```
$ DEFRAGMENT SHOW $1$DKB0:/VOLUME  
Disk File Optimizer for OpenVMS DFG V2.9  
2005, VSI OpenVMS  
F r a g m e n t a t i o n   R e p o r t  
DISK$ALPHA071                                05-JAN-2005  
15:22:58.11  
The fragmentation index is 14.2  
  1 - 20.9 is excellent  
 21 - 40.9 is good  
 41 - 60.9 is fair  
 61 - 80.9 is poor  
 81 - 100 indicates a badly fragmented disk  
Approximately 8.8 (out of 80.0 possible) is due to file fragmentation  
Approximately 5.4 (out of 20.0 possible) is due to freespace fragmentation  
Freespace Summary:  
  Total free space:          146848 blocks  
  Percentage free:           7 (rounded)  
  Total free extents:        27  
  Maximum free extent:      30312 blocks, LBN: 1799744
```

```
Minimum free extent:      4 blocks, LBN: 125948
Average free extent:      5438 blocks
Median free extent:       492 blocks
File Fragmentation Summary:
  Number of files (with some allocation):  8363
  Total file extents on the disk:          8887
  Average number of file extents per file: 1.062657
  Median number of file extents per file:  1
Most Fragmented File:
  [SYSTEM_FILES.SPIELN]ACCOUNTNG.DAT;7 (85 extents)
```

Refer to *Appendix C, "DEFRAGMENT SHOW Command Report"* for information about the above report.

2.6.6. Modifying Scheduling Scripts

You can modify scripts by two methods. If you have the LOAD_DB.COM command procedure, you can, for example, change the defragmentation interval on \$1\$DUA2 as follows:

```
$ DEFRAGMENT VOLUME $2$DKA100: -
  /SCRIPT=DEFRAG_$2$DKA100 -
  /INTERVAL=1-00:00 -
  /AFTER=21:00 -
  /FULL -
  /LOG=DFG$DATABASE:DEFRAG_$2$DKA100 -
  /NODE=LUNTIC -
  /MAIL/NOOPCOM_NOTIFICATION
$ DEFRAGMENT VOLUME $1$DUA2: -
  /SCRIPT=DEFRAG_$1$DUA2 -
  /INTERVAL=2-00:00 -           !<- changed 3 to 2
  /AFTER=17:00 -
  /FULL -
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA2
$ DEFRAGMENT OFFLINE_VOLUME $1$DUA2: -
  /SCRIPT=DEFRAG_OFFLINE_$1$DUA3
  /NOAFTER
$ EXIT
```

After changing to the command procedure, use the following commands to make the change in the scheduling database:

```
$ DEFRAGMENT REMOVE *
$ @LOAD_DB
```

You must wait 30 seconds or so between these commands to allow DFO to delete the scripts from the database.

You may also effect this interval change to \$1\$DUA2 using the following command:

```
$ DEFRAGMENT MODIFY DEFRAG_$1$DUA2/INTERVAL=2-00:00
```

2.7. Example Scheduling Scripts for a VMSccluster

The first example creates scheduling scripts for a VMSccluster consisting of 4 CPUs and 20 volumes. The CPUs are an Alpha system and 3 VAX systems. The Alpha system has its own system volume, 2 of the

VAX systems share one system volume, and the other VAX system has its own system volume. The 24 volumes consist of the following volumes:

System volumes (one shadowed)

Volumes where layered products are installed

A volume for log files

Volumes for user files (one is a RAID set from StorageWorks RAID Software)

A locally mounted volume

Volumes for code management system (CMS) files and software builds

The volumes are shown below:

Device Name	Device Status	Error Count	Volume Label
DSA0:	Mounted	0	VAXVMS062
DPA0: (LUNTIC)	Online	0	(remote access)
DPA1: (LUNTIC)	Mounted	0	GROUP
\$1\$DKA130: (LUNTIC)	Mounted	0	PSYCHO
\$1\$DUA0: (BRNDMG)	ShadowSetMember	0	(member of DSA0:)
\$1\$DUA1: (BRNDMG)	ShadowSetMember	0	(member of DSA0:)
\$1\$DUA2: (BRNDMG)	Mounted	0	CMS
\$1\$DUA3: (BRNDMG)	Mounted	0	BUILD
\$1\$DUA11: (BRNDMG)	Mounted	0	USER1
\$1\$DUA12: (BRNDMG)	Mounted	0	USER2
\$1\$DUA13: (BRNDMG)	Mounted	0	USER3
\$1\$DUA14: (BRNDMG)	Mounted	0	USER4
\$1\$DUA52: (BRNDMG)	Mounted	0	GROUP0000001
\$1\$DUA53: (BRNDMG)	Mounted	0	GROUP0000002
\$1\$DUA54: (BRNDMG)	Mounted	0	VAX_PRODUCTS
\$1\$DUA55: (BRNDMG)	Mounted	61	LOGFILES
\$1\$DUA58: (BRNDMG)	Mounted	0	PRNOID_PAGE
\$1\$DUA59: (BRNDMG)	Mounted	0	LUNTIC_PAGE
\$1\$DUA60: (BRNDMG)	Mounted	0	SENILE_PAGE
\$1\$DUA63: (BRNDMG)	Mounted	0	PHYSICAL
\$1\$DUA110: (DMNTIA)	Mounted	0	ALPVMS062
\$1\$DUA140: (DMNTIA)	Mounted	0	ALP_PRODUCTS
\$4\$DIA1: (NROSIS1)	Mounted	0	OPENVMS062

Table 2.1, "Script Scheduling Policies" shows the policy and reason for creating scripts for different volume attributes.

Table 2.1. Script Scheduling Policies

Volume Attributes	Policy	Reason
Volume receives Minimal writes	Defragment every 7 days	Fewer writes mean less defragmentation
Volume use is minimal	Defragment every 7 days	Fewer writes mean less defragmentation
Volume receives constant writes	Defragment every day	Constant writes and deletes cause more fragmentation
Volume is very busy	Defragment during periods of low use	Many users mean many files will not be defragmented
Volume is mounted to all nodes	Do not restrict which node can defragment the volume	No restrictions mean more efficient processing

Volume Attributes	Policy	Reason
Volume is mounted locally	Restrict defragmentation process to the node to which it is mounted	Only that node can see the volume

If defragmentation cannot be scheduled when few users are on the system, the priority of the process may be changed to take advantage of system resources as they are available.

For example, the scheduling scripts for the system volumes and layered software products volumes. Both of these types of volumes have a minimum amount of writes to them.

The following shows which nodes boot off of which volume:

- LUNTIC boots off of \$1\$DUA110:
- PRNOID and SENILE boot off of DSA0: (shadow set)
- NROSIS boots off of \$4\$DIA1:

The following shows part of LOAD_DB.COM that would create scripts for these volumes:

```
$ DEFRAGMENT VOLUME DSA0: -
  /SCRIPT=DEFRAG_DSA0 -
  /INTERVAL=7-00:00 -
  /AFTER=17:00 -
  /FULL -
  /LOG=DFG$DATABASE:DEFRAG_DSA0 -
  /NODE=(PRNOID, SENILE) -
  /MAIL/NOOPCOM_NOTIFICATION
$ DEFRAGMENT VOLUME $1$DUA110: -
  /SCRIPT=DEFRAG_$1$DUA110 -
  /INTERVAL=7-00:00 -
  /AFTER=17:00 -
  /FULL -
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA110 -
  /NODE=(LUNTIC) -
  /MAIL/NOOPCOM_NOTIFICATION
$ DEFRAGMENT VOLUME $4$DIA1: -
  /SCRIPT=DEFRAG_$4$DIA1 -
  /INTERVAL=7-00:00 -
  /AFTER=17:00 -
  /FULL -
  /LOG=DFG$DATABASE:DEFRAG_$4$DIA1 -
  /NODE=(NROSIS) -
  /MAIL/NOOPCOM_NOTIFICATION
$ DEFRAGMENT VOLUME $1$DUA140: -
  /SCRIPT=DEFRAG_$1$DUA140 -
  /INTERVAL=7-00:00 -
  /AFTER=17:00 -
  /FULL -
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA140 -
  /NODE=(LUNTIC) -
  /MAIL/NOOPCOM_NOTIFICATION
$ DEFRAGMENT VOLUME $1$DUA54: -
  /SCRIPT=DEFRAG_$1$DUA54 -
  /INTERVAL=7-00:00 -
  /AFTER=17:00 -
```

```
/FULL -  
/LOG=DFG$DATABASE:DEFRAG_$1$DUA54 -  
/NODE=(NROSIS,SENILE,PRNOID) -  
/MAIL/NOOPCOM_NOTIFICATION
```

The next group is the user volumes. One of the user volumes is a RAID set. These volumes are mounted on all nodes in the cluster so defragmentation is not restricted. The following shows part of `LOAD_DB.COM` that would create scripts for these volumes:

```
$ DEFRAGMENT VOLUME DPA1: -  
  /SCRIPT=DEFRAG_DPA1 -  
  /INTERVAL=1-00:00 -  
  /AFTER=17:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_DPA1 -  
  /MAIL/NOOPCOM_NOTIFICATION  
$ DEFRAGMENT VOLUME $1$DUA11: -  
  /SCRIPT=DEFRAG_$1$DUA11 -  
  /INTERVAL=1-00:00 -  
  /AFTER=17:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA11 -  
  /MAIL/NOOPCOM_NOTIFICATION  
$ DEFRAGMENT VOLUME $1$DUA12: -  
  /SCRIPT=DEFRAG_$1$DUA12 -  
  /INTERVAL=1-00:00 -  
  /AFTER=17:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA12 -  
  /MAIL/NOOPCOM_NOTIFICATION  
$ DEFRAGMENT VOLUME $1$DUA13: -  
  /SCRIPT=DEFRAG_$1$DUA13 -  
  /INTERVAL=1-00:00 -  
  /AFTER=17:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA13 -  
  /MAIL/NOOPCOM_NOTIFICATION  
$ DEFRAGMENT VOLUME $1$DUA14: -  
  /SCRIPT=DEFRAG_$1$DUA14 -  
  /INTERVAL=1-00:00 -  
  /AFTER=17:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA14 -  
  /MAIL/NOOPCOM_NOTIFICATION  
$ DEFRAGMENT VOLUME $1$DUA63: -  
  /SCRIPT=DEFRAG_$1$DUA63 -  
  /INTERVAL=1-00:00 -  
  /AFTER=17:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DUA63 -  
  /MAIL/NOOPCOM_NOTIFICATION
```

The volume that is locally mounted on LUNTIC is `$1$DKA130`. The disk is not used that much, so it is scheduled once a week. The following shows part of `LOAD_DB.COM` that would create a script for this volume:

```
$ DEFRAGMENT VOLUME $1$DKA130: -  
  /SCRIPT=DEFRAG_$1$DKA130 -
```

```
/INTERVAL=7-00:00 -  
/AFTER=17:00 -  
/FULL -  
/LOG=DFG$DATABASE:DEFRAG_$1$DKA130 -  
/NODE= (LUNTIC) -  
/MAIL/NOOPCOM_NOTIFICATION
```

The last group of volumes are the disks used for the software builds each night, \$1\$DUA2: and \$1\$DUA3:. The goals are to:

- Start the scripts at 6 a.m. to avoid the nightly software builds.
- Lower the priority of the defragmentation process to minimize user impact or optimize system resources.
- Run the defragmentation process on the Alpha system and the fastest VAX system.

The following shows part of LOAD_DB.COM that would create scripts for these volumes:

```
$ DEFRAGMENT VOLUME $1$DUA2: -  
/SCRIPT=DEFRAG_$1$DUA2 -  
/INTERVAL=1-00:00 -  
/AFTER=06:00 -  
/FULL -  
/PRIORITY=DEFAULT=1, MINIMUM=0) -  
/LOG=DFG$DATABASE:DEFRAG_$1$DUA2 -  
/NODE= (PRNOID, LUNTIC) -  
/MAIL/NOOPCOM_NOTIFICATION  
$ DEFRAGMENT VOLUME $1$DUA3: -  
/SCRIPT=DEFRAG_$1$DUA3 -  
/INTERVAL=1-00:00 -  
/AFTER=06:00 -  
/FULL -  
/PRIORITY=DEFAULT=1, MINIMUM=0) -  
/LOG=DFG$DATABASE:DEFRAG_$1$DUA3 -  
/NODE= (LUNTIC, PRNOID) -  
/MAIL/NOOPCOM_NOTIFICATION
```

Note

Scripts were not created for the following volumes because:

- The paging volumes (\$1\$DUA58:, \$1\$DUA59:, and \$1\$DUA60:) have only one file, the paging file. The paging file is always open so DFO will not defragment any files on the volumes.
- The individual RAID set volumes (\$1\$DUA52: and \$1\$DUA53:), have one large file which is not fragmented. We created the script to defragment the RAID set, DPA1:.
- The individual shadow set members (\$1\$DUA0: and \$1\$DUA1:) are defragmented with the script for DSA0:.

2.8. Creating Scheduling Scripts for a Standalone System

In this example, the standalone system consists of an Alpha system and three volumes. The three volumes are shown below:

Device Name		Device Status	Error Count	Volume Label
\$1\$DKA0:	(SPIELN)	Mounted	0	ALPHA071
\$1\$DKA100:	(SPIELN)	Mounted	0	USER1
\$1\$DKB0:	(SPIELN)	Online	0	BUILD

Because no one is using the system at night, schedule the scripts to run at 9 p.m. The following shows the LOAD_DB.COM for this system:

```
$ DEFRAGMENT VOLUME $1$DKA0: -  
  /SCRIPT=DEFRAG_$1$DKA0 -  
  /INTERVAL=1-00:00 -  
  /AFTER=21:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DKA0 -  
  /MAIL/NOOPCOM_NOTIFICATION  
$ DEFRAGMENT VOLUME $1$DKA100: -  
  /SCRIPT=DEFRAG_$1$DKA100 -  
  /INTERVAL=1-00:00 -  
  /AFTER=21:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DKA100 -  
  /MAIL/NOOPCOM_NOTIFICATION  
$ DEFRAGMENT VOLUME $1$DKB0: -  
  /SCRIPT=DEFRAG_$1$DKB0 -  
  /INTERVAL=1-00:00 -  
  /AFTER=21:00 -  
  /FULL -  
  /LOG=DFG$DATABASE:DEFRAG_$1$DKB0 -  
  /MAIL/NOOPCOM_NOTIFICATION
```


Chapter 3. Using Defrag

3.1. Introduction

This chapter explains how to use the Defrag variant of the Disk File Optimizer to defragment files and disks. It contains the following sections:

- General guidelines for using the Defrag software
- Examples illustrating how to use the Defrag software
- The Defrag command set
- Defrag command reference material in a format similar to that of the *VSI OpenVMS DCL Dictionary*

3.2. Defrag Capabilities

The Defrag variant of the Disk File Optimizer software provides capabilities to defragment files and volumes, but not for scheduling and storing information about the defragmentation operations. To schedule defragmentation operations, you can create a batch file to define defragmentation operations, or you can use a scheduling product. The Defrag PLUS variant of the software, as described in *Chapter 4, "Using Defrag PLUS"*, provides scheduling options and allows you to store defragmentation commands in a database.

3.2.1. Interface

Disk File Optimizer commands are available from the DCL interface or from within the Disk File Optimizer. When you enter a command from within the Disk File Optimizer, you do not enter the word “DEFRAGMENT” first, except for the DEFRAGMENT FILE and DEFRAGMENT VOLUME commands.

Examples of defragmentation DCL commands:

```
$ DEFRAGMENT FILE *.C /FULL/LOG=defrag_c.log
$ DEFRAGMENT MONITOR DUA1:
$ DEFRAGMENT SHOW DUA1:
```

The same commands entered from within the DFO utility would look like this:

```
$ DEFRAGMENT
DFO> DEFRAGMENT FILE *.C /FULL/LOG=defrag_c.log
DFO> MONITOR DUA1:
DFO> SHOW DUA1:
```

3.2.2. Graphic User Interface

The Disk File Optimizer graphic user interface GUI is available by entering the following command at the DCL prompt:

```
$ DEFRAGMENT/INTERFACE=DECWINDOWS
```

You can explicitly select the character-cell interface by entering this command at the DCL prompt:

```
$ DEFRAGMENT / INTERFACE=CHARACTER_CELL
```

3.2.3. Online Help

If you need a reminder of the Disk File Optimizer command syntax while working on the system, you can access online help at the DCL prompt or from within the Disk File Optimizer utility. To access online help, enter the following commands at the DCL prompt:

```
$ HELP DEFRAGMENT
```

or

```
$ DEFRAGMENT HELP
```

To access online help from within the Disk File Optimizer software, enter the following command:

```
DFO> HELP
```

3.2.4. Speed and Effectiveness of Defragmentation Operations

How quickly and effectively the defragmentation process defragments a volume depends on a number of variables, such as:

- The amount of free space on the volume.

If all other variables are constant, a volume with little free space takes longer to defragment than a volume with a lot of free space.

- The number of users accessing files on the volume.

Because the defragmentation process does not impact a user's ability to read and write files, it cannot defragment a file that is being accessed. Thus a large number of user file accesses on the target volume can negatively impact the effectiveness of the defragmentation process.

- User-set CPU or I/O limits.

In general, the more restrictive the limits, the longer it takes to defragment a given volume.

3.2.5. Scheduling Defragmentation Operations

Although Defrag does not provide actual scheduling capabilities, you still need to consider the best time to run defragmentation operations on your system. Use VAX Performance Analyzer (VPA) or DEC Performance Solution (DECps) software to find many of the parameters, such as periods of low activity, that will help determine the best times to schedule the defragmentation process.

3.2.5.1. Scheduling Guidelines

Use the following scheduling guidelines:

- Schedule defragmentation during periods of low user activity. This step ensures the process can defragment all candidate files. If a user opens a file while the defragmentation process is running, the fragmented file cannot be defragmented.

- Schedule the defragmentation process periodically to maintain volumes in a defragmented state. This also makes the defragmentation process more effective.
- Schedule defragmentation before and after user activity that may significantly contribute to fragmentation, such as end-of-month processing.

3.2.5.2. Using DECscheduler

You can use the DECscheduler product with Defrag to schedule defragmentation operations. There are no special requirements for doing this. For information about DECscheduler, see *DECscheduler document*.

3.2.6. Using Hot-files Information

The Disk File Optimizer allows you to automatically use hotfile records from the VPA or DECps database to identify frequently accessed files. The defragmentation process automatically determines which, if either, of these products is installed on your system and queries the appropriate database. To augment the frequent files list using the hotfiles records from one of these products, use the /HOTFILE_DATABASE qualifier with the DEFRAGMENT FILE or DEFRAGMENT VOLUME command.

If you use the /HOTFILE_DATABASE qualifier, the defragmentation process queries the appropriate database for hotfile records relating to the target volume. The files listed in the hotfile records are automatically added to the /FREQUENT file list.

The standard defragmentation process samples hotfile data using the previous 24 hour period (that is, 24 hours before you enter the DEFRAGMENT FILE or DEFRAGMENT VOLUME command) as a sample time.

Both VPA and DECps allow you to modify the hotfile collection process. If you have VPA installed, use the ADVISE /COLLECT command and /[NO]HOTFILE_QUEUE={ value } qualifier. If you have DECps installed, use the ADVISE COLLECT SET command and the / [NO]HOTFILE_QUEUE={ value } qualifier. See the VPA or DECps documentation for additional details.

3.2.7. Defining Quotas for the Defragmentation Process

The process quotas are values applied to the defragmentation process and then disappear when the process is complete. For more information on process quota values, refer to the *VMS Authorize Utility Manual*.

You can define the process quotas for defragmentation processes. When the defragmentation process starts up it selects the maximum value from the following three sources:

- Hard coded values
- SYSGEN process quotas (PQL_Mxxx)
- DFO logical process quotas

By changing the SYSGEN process quotas or DFO logical process quotas you can define the process quotas for the defragmentation processes. The following sections describes these values and describes how to change them.

3.2.7.1. Hard coded values

You cannot change the hard coded values for the defragmentation processes but the following shows the hard coded values:

- ASTLIM - 100
- BIOLM - 18
- BYTLM - 61440
- DIOLM - 18
- ENQLM - 500
- FILLM - 50
- JTQUOTA - (uses system default)
- PGFLQUOTA - 50000
- PRCLM - 8
- TQELM - 10
- WSDEFAULT - 512
- WSEXTENT - 16000
- WSQUOTA - 2048

3.2.7.2. SYSGEN Process Quotas

The SYSGEN PQL_Mxxx process quotas are defined in SYSGEN. You can change these dynamic PQL_Mxxx parameters in SYSGEN. For more information on SYSGEN process quota values, refer to the SYSGEN Generation Utility.

This list shows the PQL_Mxxx parameters that affect the defragmentation processes:

- PQL_MASTLM
- PQL_MBIOLM
- PQL_MBYTLM
- PQL_MDIOLM
- PQL_MENQLM
- PQL_MFILLM
- PQL_MJTQUOTA
- PQL_MPGFLQUOTA

- PQL_MPRCLM
- PQL_MTQELM
- PQL_MWSDEFAULT
- PQL_MWSEXTENT
- PQL_MWSQUOTA

3.2.7.3. DFO Logical Process Quotas

DFO provides logicals that allow you to define the process quotas for the defragmentation process. These logicals are defined in SYS\$STARTUP:DFG\$STARTUP.COM. When you install DFO these logicals are commented out. By uncommenting the affected line and executing DFG\$STARTUP.COM you can affect the quotas for the defragmentation processes.

For example, if a disk containing a large database routinely encounters the insufficient virtual memory error (INSVIRMEM) during defragmentation, increase the page file quota (PGFLQUOTA) to a value larger than the DFO default of 50,000.

```
$ DEFINE/SYSTEM/EXECUTIVE_MODE DFG$PGFLQUOTA 120000
```

These rules apply to setting process quota values:

- You cannot decrease a quota value below the system PQL minimum as defined by SYSGEN.
- You may decrease a DFO quota value to the associated PQL_Mxxx minimum value, but this is not recommended.

The right way to change these logicals is to edit the file SYS\$STARTUP:DFG\$STARTUP.COM. An example of changing the page file quota is shown below:

```
$! To use these logicals, increase the associated value, uncomment the
$! desired line, and execute this file.
$!
$! CALL DEFINE_LOGICAL DFG$ASTLM      100
$! CALL DEFINE_LOGICAL DFG$BIOLM      18
$! CALL DEFINE_LOGICAL DFG$BYTLM      61440
$! CALL DEFINE_LOGICAL DFG$DIOLM      18
$! CALL DEFINE_LOGICAL DFG$ENQLM      500
$! CALL DEFINE_LOGICAL DFG$FILLM      50
$! CALL DEFINE_LOGICAL DFG$JTQUOTA    {uses system default}
$ CALL DEFINE_LOGICAL DFG$PGFLQUOTA   120000
$! CALL DEFINE_LOGICAL DFG$PRCLM      8
$! CALL DEFINE_LOGICAL DFG$TQELM      10
$! CALL DEFINE_LOGICAL DFG$WSDEFAULT  512
$! CALL DEFINE_LOGICAL DFG$WSEXTENT   16000
$! CALL DEFINE_LOGICAL DFG$WSQUOTA    2048
```

3.2.7.4. Defining the Transfer Size Logical

The logical, DFG\$TRANSFERSIZE, helps you tune the Disk File Optimizer. The definition of this logical allows you to change the amount of data that is transferred at onetime. On a volume that has files with large extents, you could wait up to 20 minutes while the system transfers these large extents before you could access the file. By defining this logical, you could access this file in less than a minute.

This logical changes the impact that the Disk File Optimizer has on applications that are simultaneously accessing the same files.

This logical is set to 10,000 blocks by default. You can change it by editing SYS\$STARTUP:DFG \$STARTUP.COM and then executing the command procedure.

3.3. Wildcard Character Use

The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*.*. Specifying *.TMP;* selects all *.TMP;* files on the entire volume.

3.4. Examples Using Defrag

This section provides some general examples of what you can do with the Defrag software. Additional examples are shown for each command in *Section 3.5, "Defrag Command Set"*.

3.4.1. Creating Log Files

Log files are created using the /LOG qualifier. There are two types of log files:

- Brief
- Full

Brief log files provide an abbreviated description of events that occur during a defragmentation operation. Brief log files are the default log file type.

Full log files provide an extended version of events that occur during defragmentation operations. Full log files are obtained by specifying /FULL.

Examples of brief and full log files can be found in *Appendix D, "Brief Log File Example"* and *Appendix E, "Full Log File Example"*, respectively. The command shown in the following example creates a full log file:

```
$ DEFRAGMENT VOLUME DISK$ONE/LEVEL=2/LOG=DISK$ONE_ANA.LOG/FULL
```

In log files, the symbols preceding a file name indicate the following:

Blank	- file has been preprocessed for defragmentation
d	- file has been defragmented
*	- file has been partly defragmented
D	- file has been placed for dormant access
Dd	- file has been defragmented and placed dormant
D*	- file has been placed dormant and been partly defragmented
F	- file has been placed for frequent access
Fd	- file has been defragmented and placed frequent
F*	- file has been placed frequent and been partly
defragmented	
Moved	- file has been moved (number of extents unchanged)
EMPTY	- file was empty; no action taken
FAIL	- file's fragmentation cannot be reduced
IN USE	- new disk space for file was in use; no action taken
OPEN	- file was open; no action taken
NOMOVE	- file may not be moved; no action taken
UNAVL	- file was unavailable; no action taken

3.4.2. Showing the State of a Disk

The defragmentation software can show the current fragmentation state of a volume. The **SHOW** subcommand provides a variety of report options that show the level of fragmentation on a given volume.

For example, to view the default information in the volume fragmentation report, enter the following command:

```
$ DEFRAGMENT SHOW DISK$ONE/VOLUME
```

This command generates a fragmentation index, free space summary, and file fragmentation summary. The fragmentation index is an estimate of the level of fragmentation on the specified volume. Its value is derived from the number of files on the volume, their sizes, the number of extents in those files, and the state of the free space on the volume. The file and free space fragmentation summary provides more detail on the fragmentation index. Refer to *Appendix C, "DEFRAGMENT SHOW Command Report"* for more information on this report.

Other options available show file and free space histograms, list all free space holes, list all files with more than a given number of extents, and provide locations for specified files.

3.5. Defrag Command Set

The Defrag command set allows the system manager to perform file and volume defragmentation tasks. *Table 3.1, "Defrag Command Summary"* describes the commands available for Defrag.

Table 3.1. Defrag Command Summary

Command	Description
DEFRAGMENT ABORT {volume-name[,...]} [/QUALIFIERS...]	Aborts the defragmentation process associated with the specified volume. Before exiting, the defragmentation process performs an orderly shutdown of all outstanding activities.
DEFRAGMENT ATTACH {process_name} [/QUALIFIERS...]	Transfers control from the current process (which goes into hibernation) to the specified process.
DEFRAGMENT EXIT	Exits the user from the Disk File Optimizer utility and returns to the system prompt.
DEFRAGMENT FILE {file_name[,...]} [/QUALIFIERS...]	Defragments a selected file (or files).
DEFRAGMENT HELP {command}	Invokes VMS HELP utility for the Disk File Optimizer software.
DEFRAGMENT MONITOR {volume-name} [/QUALIFIERS...]	Invokes the defragmentation monitor, which monitors an ongoing defragmentation process. The input parameter must be a volume name.
DEFRAGMENT OFFLINE_VOLUME {volume-name[,...]} [/QUALIFIERS...]	Defragments INDEXF.SYS on the specified volume according to the selected command line qualifiers.
DEFRAGMENT SHOW [{volume-name[,...]}] [/QUALIFIERS...]	Shows the fragmentation state of the specified volumes.

Command	Description
DEFRAGMENT SPAWN {command}	Creates a subprocess and either transfers control to the subprocess or executes a DCL command from within the subprocess.
DEFRAGMENT VOLUME {volume_name[,...]} [/QUALIFIERS...]	Defragments a selected volume (or volumes). The DEFRAGMENT VOLUME command creates a detached process that defragments the target volume. This process executes immediately.

3.5.1. The Defrag Commands

This section provides detailed information about each command for the Defrag variant of the Disk File Optimizer software.

DEFRAGMENT ABORT

DEFRAGMENT ABORT — The ABORT subcommand terminates the defragmentation process on the volume specified in the command line.

Format

DEFRAGMENT ABORT [{ volume-name[,...] } { /ALL }]

Parameters

volume-name

The volume-name parameter identifies the volume or volumes on which the defragmentation process is to be aborted. The volume-name specification accepts the standard wildcard characters. If you use an asterisk (*) for the volume-name specification, Defrag aborts all defragmentation processes on the VAXcluster system. If you use a percent sign (%) to match a particular character in the volume-name specification, Defrag aborts the defragmentation processes on the volumes in the VAXcluster system with names meeting the specification. For example, to abort defragmentation processes on all DUA drives in the VAXcluster system, use DUA% for the volume-name specification.

Description

The ABORT subcommand terminates execution of the defragmentation process(es) associated with the volumes specified on the command line. The process is terminated in an orderly fashion. The defragmentation process completes current functions, executes housekeeping functions, and then exits, leaving the volume fully usable and accessible.

Restrictions

The volume-name parameter is not allowed if the /ALL qualifier is used.

Qualifiers

/ALL

/NOALL

The /ALL qualifier aborts all defragmentation processes on the system. This is equivalent to using an asterisk (*) for the volume-name specification.

Examples

The following are examples of the ABORT command:

1. `$ DEFRAGMENT ABORT DISK$USER1`

This command aborts the defragmentation process currently executing on volume DISK\$USER1.

2. `$ DEFRAGMENT ABORT /ALL`

This command aborts all defragmentation processes on all volumes in the VAXcluster system.

DEFRAGMENT ATTACH

DEFRAGMENT ATTACH — The ATTACH subcommand transfers control from the current process (which then hibernates) to the specified process.

Format

DEFRAGMENT ATTACH [{ process-name | /IDENTIFICATION=pid }]

Parameters

process-name

The process-name specifies the name of a process to which control transfers. The process must already exist, be part of your current job, and share the same input stream as your current process. However, the process cannot be your current process or a subprocess created with the /NOWAIT qualifier.

Process names can contain from 1 to 15 alphanumeric characters. If a connection to the specified process cannot be made, an error message is displayed.

Description

The ATTACH command allows you to connect your input stream to another process. You can use the ATTACH command to change control from one subprocess to another subprocess or to the parent process.

When you enter the ATTACH command, the parent or “source” process is put into hibernation and your input stream is connected to the specified destination process. You can use the ATTACH command to connect to a subprocess that is part of a current job left hibernating as a result of the DCL SPAWN/WAIT command or another ATTACH command, as long as the connection is valid. (No connection can be made to the current process, to a process that is not part of the current job, or to a process that does not exist. If any of these connections are attempted, an error message is displayed.)

You can also use the ATTACH command with the DCL SPAWN /WAIT command to return to a parent process without terminating the subprocess running the Disk File Optimizer.

Restrictions

- The ATTACH command cannot be used if your terminal has an associated mailbox.
- You cannot use the process-name parameter with the /IDENTIFICATION qualifier.

- You must supply either a process name or a PID for this command.

Qualifiers

/IDENTIFICATION=pid

/IDENTIFICATION lets you specify the process identification (PID) of the process to which terminal control will be transferred. Leading zeros can be omitted.

Examples

The following are examples of the ATTACH command:

1. `$ DEFRAGMENT ATTACH JONES_2`

This ATTACH command transfers the terminal's control to the subprocess named JONES_2.

2. `$ DEFRAGMENT ATTACH /IDENTIFICATION=2A401A1B`

This ATTACH command switches control from the current process to the process whose PID is 2A401A1B.

DEFRAGMENT EXIT

DEFRAGMENT EXIT — The EXIT subcommand exits the Disk File Optimizer utility and returns the system prompt to the user.

Format

DEFRAGMENT EXIT

Parameters

None.

Description

The EXIT subcommand exits the user from the Disk File Optimizer utility. EXIT shuts everything down in an orderly sequence.

Restrictions

None.

Qualifiers

None.

Examples

The following is an example of the EXIT command:

```
DFO> EXIT
```

\$

This command exits the user from the Disk File Optimizer utility and returns the user to the DCL prompt.

DEFRAGMENT FILE

DEFRAGMENT FILE — The FILE command defragments the files specified on the command line according to the DEFRAGMENT command line. Use this command to defragment a small number of files. To defragment a large number of files, use the DEFRAGMENT VOLUME command.

Format

DEFRAGMENT FILE [file-spec[,...]] [{ /BRIEF | /CONSOLIDATE_FREESPACE | /CPU_LIMIT | /DORMANT | /EPILOGUE | /EXCLUDE | /FREQUENT | /FULL | /HOTFILE_DATABASE | /IGNORE | /IO_LIMIT | /LEVEL | /LOG | /OPCOM_NOTIFICATION | /OPTIONS | /PRIORITY | /PROLOGUE | /RVN | /TIME_LIMIT | /WAIT | /WRITE_CHECK }]

Parameters

file-spec[,...]

The file-spec parameter identifies the files to be defragmented. You may use wildcards for the file specification.

Description

The DEFRAGMENT FILE command defragments the specified files according to the selected command line qualifiers. When you specify a file name without the device and/or directory, the defragmentation software operates on files in the current directory on the current device. You must explicitly specify the device and directory name for files not in the default directory.

The defragmentation process begins immediately.

Restrictions

- The /LOG qualifier is required with /LEVEL=5.
- /BRIEF and /FULL cannot be used concurrently.

Qualifiers

/BRIEF (default)

/NOBRIEF

The /BRIEF qualifier can be used only when the /LOG qualifier is used. The /BRIEF qualifier requests a brief log of all actions taken by the defragmentation process. This log includes beginning and ending file status information. The /BRIEF and /FULL qualifiers are mutually exclusive, and /NOBRIEF produces the same result as /FULL. The /BRIEF qualifier is the default.

/CONSOLIDATE_FREESPACE

/NOCONSOLIDATE_FREESPACE (default)

The /CONSOLIDATE_FREESPACE qualifier tells the Disk File Optimizer to use its free space consolidation algorithm.

/CPU_LIMIT=n
/NOCPU_LIMIT

The /CPU_LIMIT qualifier specifies the percent of CPU resources that may be used for anyone defragmentation process. If this percentage is exceeded, the priority for the process drops to the minimum as specified by the /PRIORITY qualifier, or defaults to three.

/DORMANT=(file-spec[,...])
/NODORMANT (default)

The /DORMANT qualifier indicates files to be considered dormant when final file placement decisions are made. In order for the file to be placed dormant, it must be the file specified in the command line. Dormant files are placed at the high end of the volume.

If a file is indicated in both the DORMANT and FREQUENT lists, it is considered dormant. If a file is indicated in both the EXCLUDE and DORMANT lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

/EPILOGUE={file-name}
/NOEPILOGUE (default)

The /EPILOGUE qualifier identifies a DCL command file which DFO executes upon completion of the DEFRAGMENT operation.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/EXCLUDE=(file-spec[,...])
/NOEXCLUDE (default)

The /EXCLUDE qualifier specifies files to be excluded from processing. The defragmentation process ignores files matching the listed file specifications. Any file indicated in the EXCLUDE list and in either the FREQUENT or DORMANT lists is excluded from processing.

Note

The defragmentation process always excludes system files, open files, files excluded in an options file (/OPTIONS qualifier), and files excluded using the /IGNORE qualifier.

/FREQUENT=(file-spec[,...])
/NOFREQUENT (default)

The /FREQUENT qualifier indicates files to be considered frequently accessed when final file placement decisions are made. In order for the file to be placed frequent, it must be the file specified in the command line. Frequent files are placed toward the center of the volume.

If a file is indicated in both the FREQUENT and DORMANT lists, it is considered dormant. If a file is indicated in both the FREQUENT and EXCLUDE lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

/FULL

/NOFULL (default)

The /FULL qualifier requests a full log of all actions taken by the defragmentation process. The /FULL qualifier must be used with the /LOG qualifier. The /FULL and /BRIEF qualifiers are mutually exclusive, and /NOFULL provides the same functionality as /BRIEF.

/HOTFILE_DATABASE

/NOHOTFILE_DATABASE (default)

The /HOTFILE_DATABASE qualifier causes the defragmenter to query the appropriate hotfile database and append the list of hotfiles to the defragmenter's FREQUENT file list.

/IGNORE=(keyword[,...])

/IGNORE=(PLACED)(default)

/NOIGNORE

The /IGNORE qualifier requests that the defragmentation process ignore certain types of files as specified by the following keywords:

- INDEXED—Ignores all RMS indexed files
- NOINDEXED—May defragment RMS indexed files
- PLACED—Ignores all files with explicit placement flags set in their headers
- NOPLACED—May defragment files with explicit placement flags

Files indicated by the /IGNORE qualifier are added to the EXCLUDE list.

The default value (/IGNORE=PLACED) indicates that files with explicit placement control should be ignored, and that RMS indexed files should be defragmented and placed in appropriate locations on the volume.

/IO_LIMIT=n

/NOIO_LIMIT

The /IO_LIMIT qualifier specifies the maximum direct I/O rate (per second) of any one defragmentation process. If the I/O rate is exceeded, the priority for the process drops to the minimum priority as specified by the /PRIORITY qualifier or defaults to three.

/LEVEL={n}

/LEVEL=2 (default)

The defragmentation process selects files for defragmentation (from among those not excluded) based on the /LEVEL qualifier. Once a file is selected as a candidate, the defragmentation process attempts to fully defragment that file (make it 100 percent contiguous). Specify the level number as an integer ranging from one to five:

- **/LEVEL=1**

Any file that is not 100 percent contiguous, or that can be selected using levels two and three, is a candidate for defragmentation.

- **/LEVEL=2 (default)**

Any file whose extents cannot be completely mapped by the mapping window, or that can be selected using level three, is a candidate for defragmentation. The default window value is seven and can be set for a given volume using:

```
$ SET VOLUME {device-spec}/WINDOWS=[n]
```

- **/LEVEL=3**

Any file with retrieval pointers in more than one file header is a candidate for defragmentation.

- **/LEVEL=4**

The defragmentation process for level four processing performs only free space consolidation. This level is used with the **/CONSOLIDATE_FREESPACE** qualifier.

- **/LEVEL=5**

This level requests a volume analysis only; no defragmentation occurs. The statistical results are available in the log file, which is required when **/LEVEL=5** is selected (see the **/LOG** qualifier). If level 5 is selected, the **/LEVEL=2** candidate file spec

/LOG={file-spec}**/NOLOG (default)**

The **/LOG** qualifier causes the defragmentation process to create a file containing a list of all its operations. The information in the log file varies with the presence of the **/FULL** or **/BRIEF** qualifiers. The **/FULL** qualifier lists all files analyzed by the defragmentation process and reports their status during each phase of defragmentation. The **/BRIEF** qualifier lists only the statistics of each phase, such as number of files analyzed and number of files defragmented.

/OPCOM_NOTIFICATION (default)**/NOOPCOM_NOTIFICATION**

The **/OPCOM_NOTIFICATION** qualifier sends brief status and termination messages regarding defragmentation processes to OPCOM.

/OPTIONS={file-spec}**/NOOPTIONS (default)**

The **/OPTIONS** qualifier provides an alternate method for associating files with the **/DORMANT**, **/EXCLUDE**, or **/FREQUENT** qualifiers. If the list of files for any of these qualifiers is long, you can create a separate file that contains this information. Each record in the options file consists of a file name followed by one of the following keywords:

- **/EXCLUDE**—File to be excluded
- **/FREQUENT**—File to be considered frequently accessed
- **/DORMANT**—File to be considered dormant

The following example shows the contents of an options file called MYOPTFILE.OPT. This file specifies that all files on the target device called FILE1.EXE are frequent files, all files in the [ARCHIVE] directory are to be excluded, and all files in the [OLDSTUFF] directory and its subdirectories are dormant.

```
FILE1.EXE/FREQUENT
[ARCHIVE]*.*;*/EXCLUDE
[OLDSTUFF...]*.*;*/DORMANT
```

Files indicated by the /EXCLUDE qualifier and by either the /DORMANT or /FREQUENT qualifier are excluded. Files indicated by the /DORMANT and /FREQUENT qualifiers are considered dormant. This rule is applied equally across all files specified in the /OPTIONS file, as well as those associated with these qualifiers on the command line.

/PRIORITY=(DEFAULT=n, MINIMUM=n)
/NOPRIORITY (default)

The /PRIORITY qualifier specifies priorities for the defragment process. The minimum priority setting is used when the defragmentation process exceeds the value specified by either /CPU_LIMIT or /IO_LIMIT.

/PROLOGUE={file-name}
/NOPROLOGUE (default)

The /PROLOGUE qualifier identifies a DCL command file which DFO executes just before the Defragment operation starts.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/RVN=n
/RVN=0 (default)

When you have a bound volume set, use the /RVN qualifier to identify the relative volume number (RVN) of a specific disk (from the set) to be defragmented. The default of zero tells the Disk File Optimizer to defragment all disks in the bound volume set.

/TIME_LIMIT={delta-time}
/NOTIME_LIMIT (default)

The /TIME_LIMIT qualifier specifies the maximum elapsed time allowed for a given defragmentation operation. If the time limit is exceeded, the defragmentation process is aborted.

Specify time as a delta time. Refer to the *VSI OpenVMS User's Manual* for more information on specifying times.

If the time limit is set unreasonably low, the defragmentation process may not accomplish useful defragmentation.

/WAIT (default)
/NOWAIT

The /WAIT qualifier causes the input device to wait until the defragmentation process finishes before returning control to the user.

/WRITE_CHECK**/NOWRITE_CHECK (default)**

The **/WRITE_CHECK** qualifier requests that a read-after-write check be performed on all defragmented or placed files. This qualifier significantly extends the execution time of a defragmentation process.

Examples

The following examples show different ways to defragment a file or group of files and illustrate how to use the common qualifiers.

1. `$ DEFRAGMENT FILE *.C;*`

This command defragments all files with a `.C` extension in the current default directory. To determine the device and directory in which the files are located, issue the **SHOW DEFAULT** command.

2. `DFO> DEFRAGMENT FILE *.C;*`

This command defragments all files with a `.C` extension on the current default device and directory. The only difference between this command and the previous example is that this command is entered from within the Disk File Optimizer utility instead of from the DCL prompt.

3. `$ DEFRAGMENT FILE SYS$LOGIN:LOGIN.COM;2/EXCLUDE=*.OLD;*`

This command defragments the `LOGIN.COM` file and excludes all files with a `.OLD` extension.

DEFRAGMENT HELP

DEFRAGMENT HELP — The **HELP** subcommand invokes VMS Help for the Disk File Optimizer utility.

Format

DEFRAGMENT HELP

Parameters

None.

Restrictions

None.

Examples

The following examples show the various ways you can access **HELP** for the Disk File Optimizer utility.

1. `$ DEFRAGMENT HELP`

This command invokes **HELP** for the Disk File Optimizer utility directly from the DCL prompt.

2. `$ HELP DEFRAGMENT`

This command invokes **HELP** for the Disk File Optimizer utility directly from the DCL prompt.

3. DFO> HELP

This command invokes HELP from within the Disk File Optimizer utility.

DEFRAGMENT MONITOR

DEFRAGMENT MONITOR — The MONITOR subcommand displays the progress of ongoing defragmentation processes.

Format

DEFRAGMENT MONITOR [{ volume-name } { /CONTINUOUS }]

Parameters

volume-name

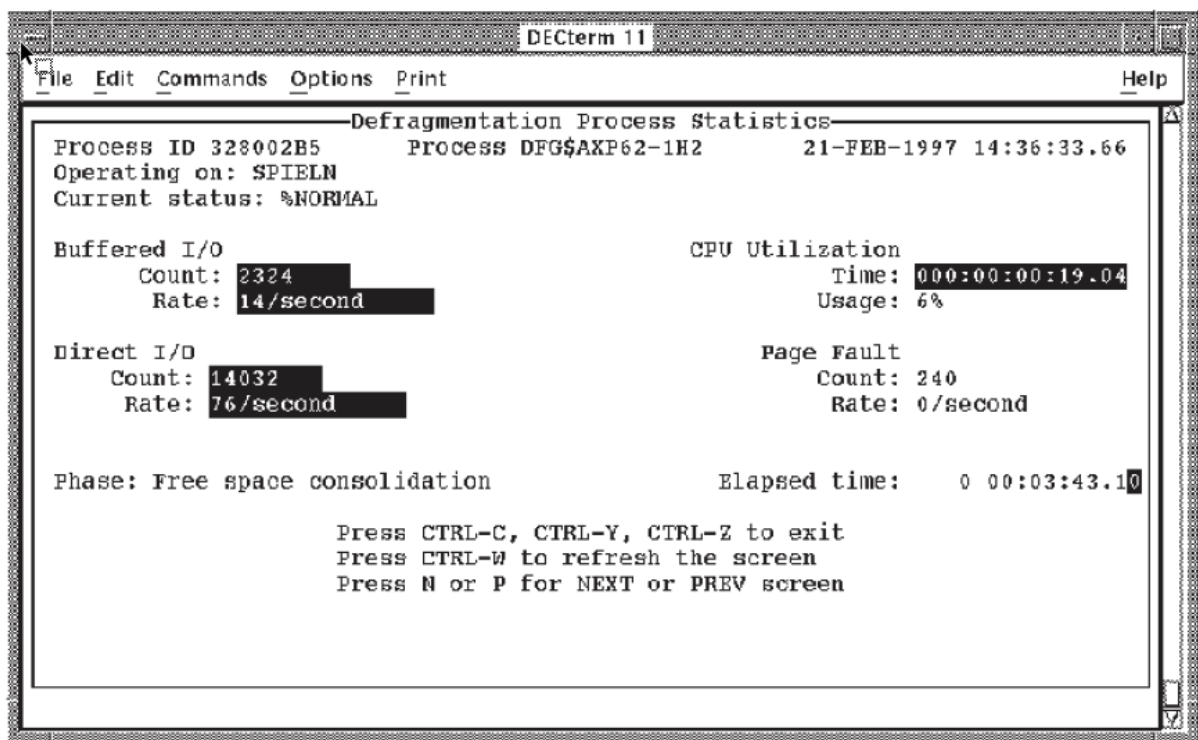
The volume-name parameter identifies the volume whose defragmentation process you want monitored.

Description

The MONITOR subcommand provides current process statistics for the defragmentation process and information about the volume being defragmented. This information is shown in three display windows—the process statistics display, the process activities display, and the free space display.

The process statistics display shows the following defragmentation process information (*Figure 3.1, "Process Statistics Display"*):

Figure 3.1. Process Statistics Display



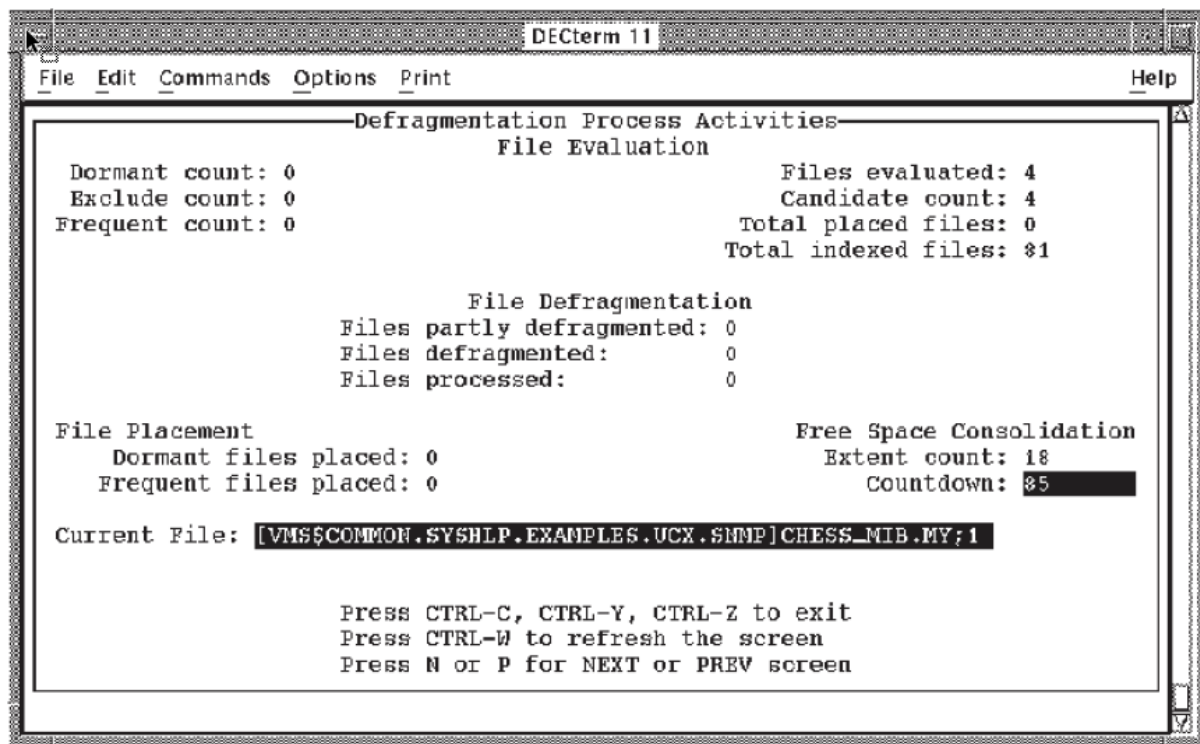
- The defragmentation process ID
- The defragmentation process name

- The current date and time
- The node on which the defragmentation process is operating
- The status of the defragmentation process*
- The buffered I/O count and rate of the defragmentation process*
- The CPU utilization (time and usage) of the defragmentation process*
- The direct I/O count and rate of the defragmentation process*
- The page fault count and rate of the defragmentation process*
- The current defragmentation process phase*
- The elapsed time since the start of the defragmentation process

The process information marked with an asterisk (*) in the preceding list appears highlighted on the process display whenever it changes.

To access the process activities display, enter N or press the Next Screen key. The process activities display shows the following process-related information (*Figure 3.2, "Process Activities Display"*):

Figure 3.2. Process Activities Display

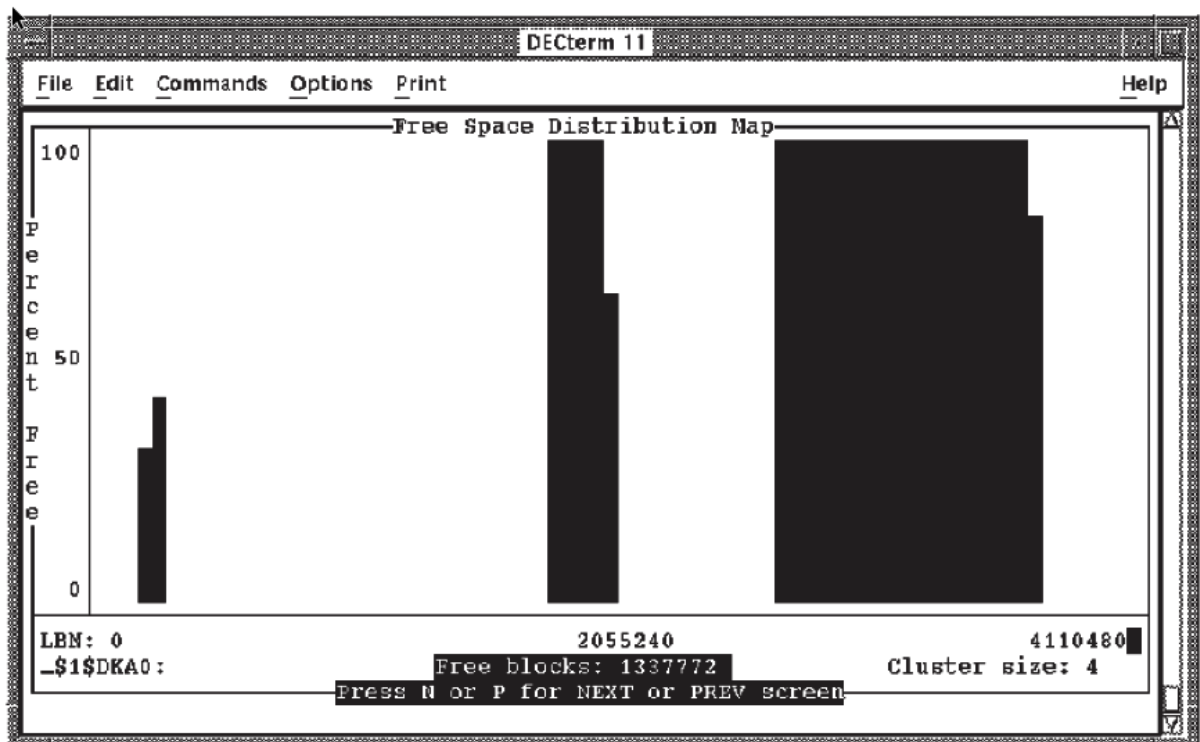


- Name of current defragmentation phase
- Number of files evaluated
- Number of candidate files for defragmentation
- Number of dormant files

- Total number of placed files
- Number of excluded files
- Total number of indexed files
- Number of frequent files
- For the file defragmentation phase:
 - Number of files partly defragmented
 - Number of files defragmented
 - Number of files processed
- For the file placement phase:
 - Number of dormant files placed
 - Number of frequent files placed
- For the free space consolidation phase:
 - Number of extents to be consolidated into free space
 - Number of extents left to be consolidated into free space
- Current file being evaluated

To access the free space display, enter N or press the Next Screen key. The free space display shows the following device-related information (*Figure 3.3, "Free Space Display"*):

Figure 3.3. Free Space Display



- The cluster size of the device
- The maximum LBN on the device

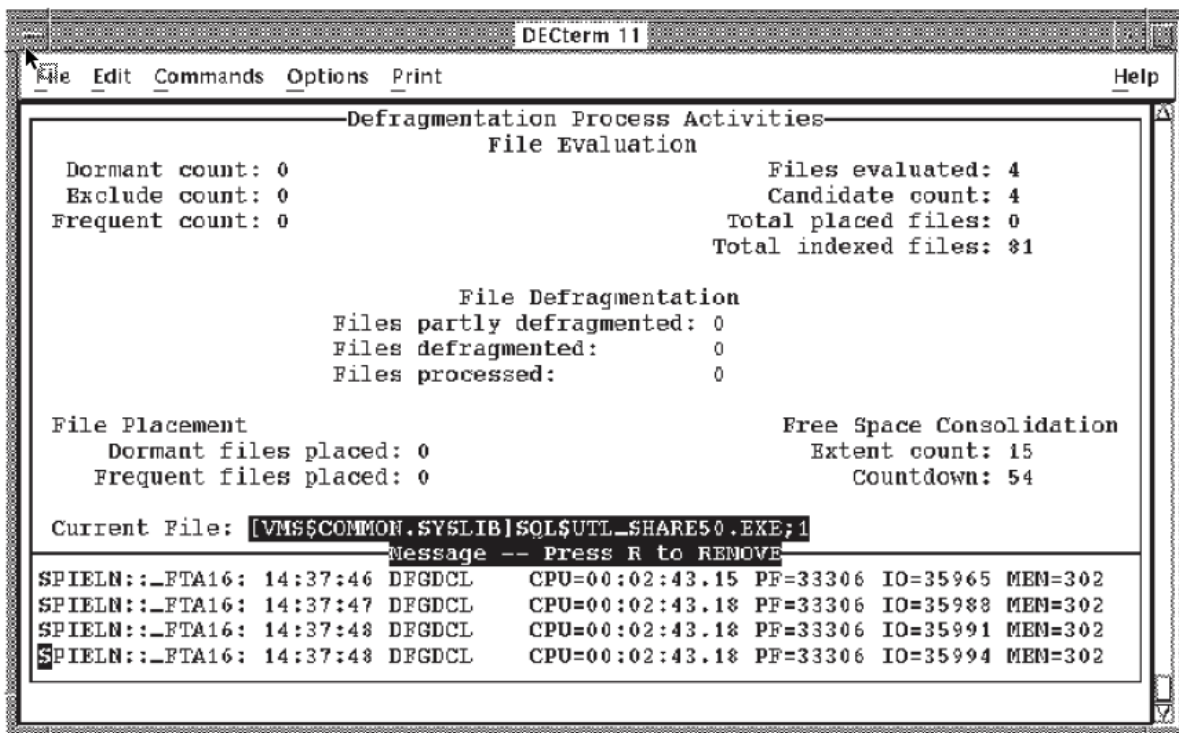
The following information, also on the device display, is highlighted whenever it changes.

- An approximate free space distribution map
- The name of the device being defragmented
- The number of free blocks on the device

The free space map represents how free space is organized on the volume being defragmented. The Y axis indicates the percentage of free space in a given LBN range; the X axis indicates the LBN range.

A fourth display of information—the message display—appears on the bottom of your screen whenever a broadcast message is received by the process executing the monitor. The message display holds the last four broadcast messages (such as mail notification) received by the process. To erase this display, enter Ctrl/W; to remove this display entirely, enter R or press the Remove key. To scroll through the broadcast messages, enter Ctrl/T. *Figure 3.4, "Process Statistics Display with Message Display"* shows the process statistics display with the additional message display.

Figure 3.4. Process Statistics Display with Message Display



The following list summarizes the keystrokes that manipulate the monitor displays:

- Ctrl/C, Ctrl/Y, or Ctrl/Z exit the monitor.
- Ctrl/W refreshes the screen and erases the message display.
- N or Next Screen switches to the next display.
- P or Previous Screen switches to previous display.

- R or Remove removes the message display.
- Ctrl/T scrolls through the broadcast messages.

Restrictions

None.

Qualifiers

/CONTINUOUS (default)

/NOCONTINUOUS

The /CONTINUOUS qualifier provides a continuous monitor display.

The /NOCONTINUOUS provides a snapshot display and then exits to the DCL prompt.

Examples

The following example shows how to invoke the monitor.

```
$ DEFRAGMENT MONITOR DISK$ONE/NOCONTINUOUS
```

In this example, the defragmentation process executing against DISK\$ONE is monitored. Since the /NOCONTINUOUS qualifier was specified, the monitor function samples the defragmentation process statistics and the free space on DISK\$ONE one time before exiting to the DCL prompt.

DEFRAGMENT OFFLINE_VOLUME

DEFRAGMENT OFFLINE_VOLUME — The OFFLINE_VOLUME subcommand defragments the INDEXF.SYS file on the specified offline volume according to the defragment command line qualifiers.

Format

DEFRAGMENT OFFLINE_VOLUME [{ volume-name } { /EPILOGUE | /LOG, brief only | /OPCOM_NOTIFICATION | /PROLOGUE | /WAIT }]

Parameters

volume-name

The volume-name is the name of the volume containing the INDEXF.SYS file to be defragmented.

Description

The OFFLINE_VOLUME subcommand defragments INDEXF.SYS on the specified volume according to the selected command line qualifiers. By definition INDEXF.SYS has a minimum of four extents. The extents consist of the following:

1. Bootstrap block and home block
2. Backup home block
3. Backup index file header
4. Index file bitmap and file headers

The minimum number of extents INDEXF.SYS can have after defragmentation is five.

Before defragmenting the INDEXF.SYS file, do the following:

1. Defragment and consolidate free space on the disk volume. Consolidating free space on the disk volume gives DFO a place to create a new defragmented INDEXF.SYS file.
2. Check the free space summary to ensure there is enough free space on the disk volume to create a defragmented INDEXF.SYS. Use the following command to check the free space summary:

```
$ DEFRAGMENT SHOW disk_volume:/VOL
```

An example of the free space summary report is shown below:

```
Freespace Summary:
  Total free space:      211072 blocks
  Percentage free:       10 (rounded)
  Total free extents:    23
  Maximum free extent:   53316 blocks, LBN: 671852
  Minimum free extent:   4 blocks, LBN: 730360
  Average free extent:   9177 blocks
  Median free extent:    584 blocks
```

If you defragment the INDEXF.SYS file without adequate free extent, the defragmentation of INDEXF.SYS could fail.

3. Analyze the disk volume to make sure it has no problems:

```
$ ANALYZE/DISK disk_name
```

If the disk needs to be repaired, use the following command to repair it:

```
$ ANALYZE/DISK/REPAIR disk_name
```

After defragmenting the INDEXF.SYS file, perform another ANALYZE/DISK/REPAIR.

Restrictions

- This command is only valid on a disk device that is not mounted. If the device is mounted, an error occurs and the process will abort. This command cannot work on the system disk because it is always mounted.
- For a bound volume set, this utility only works on the "ROOT" volume.
- For disk volumes in a shadow set (single member or otherwise), the following restrictions apply:
 - Do not specify the virtual unit name to defragment (for example: DSAnnnn:).
 - Specify only one physical device name (for example: \$1\$DUAnnnn:).
 - Use the following procedure to defragment INDEXF.SYS in a shadow set:
 1. Dismount both members of the shadow set.
 2. Mount one of the physical disk volumes with /OVERRIDE=SHADOW and then dismount it before doing the defragmentation. If you do not do this, you will receive the following error:

```
%SYSTEM-F-WRITLCK, write lock error
```

3. Defragment the INDEXF.SYS on the disk volume.
 4. Mount the disk volume into the shadow set.
 5. Add the disk volume that was not defragmented into the shadow set. Mounting this disk volume causes a full copy operation.
- If you experience a power failure during the defragmentation of INDEXF.SYS, you may need to do the following:
 - Repair alternate index file header and index file bitmap by doing a ANALYZE/DISK/REPAIR.
 - Delete disk:[000000]DFG\$DEFRAG_INDEXF.DAT.

Qualifiers

/EPILOGUE={file-name}
/NOEPILOGUE (default)

The /EPILOGUE qualifier identifies a DCL command file which DFO executes upon completion of the DEFRAGMENT operation.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/LOG={file-spec}
/NOLOG (default)

The /LOG qualifier causes the defragmentation process to create a file containing a list of all its operations.

/MAIL_NOTIFICATION
/NOMAIL_NOTIFICATION (default)

The /MAIL_NOTIFICATION qualifier requests that brief status and termination messages regarding defragmentation processes be mailed to the mail distribution list provided in the file DFG \$MAIL_ADDRESS.

/OPCOM_NOTIFICATION (default)
/NOOPCOM_NOTIFICATION

The /OPCOM_NOTIFICATION qualifier requests that brief status and termination messages regarding defragmentation processes be sent to OPCOM.

/PROLOGUE={file-name}
/NOPROLOGUE (default)

The /PROLOGUE qualifier identifies a DCL command file which DFO executes just before the Defragment operation starts.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/WAIT (default)**/NOWAIT**

The /WAIT qualifier causes the input device to wait until the defragmentation process finishes before returning control to the user.

Examples

The following example illustrates how to use the OFFLINE_VOLUME subcommand and its qualifiers.

```
$ DISMOUNT/CLUSTER $1$DKB100:
$ MOUNT/OVER=ID $1$DKB100:
$ ANALYZE/DISK_STRUCTURE $1$DKB100:
$ DISMOUNT $1$DKB100:
$ DEFRAGMENT OFFLINE_VOLUME $1$DKB100:/LOG=OFFLINE_DKB100.LOG
$ MOUNT/OVER=ID $1$DKB100:
$ ANALYZE/DISK_STRUCTURE $1$DKB100:
$ DISMOUNT $1$DKB100:
$ MOUNT/CLUSTER $1$DKB100: USER1
```

In this example, the following is performed:

1. Dismount volume from the cluster
2. Mount volume privately to analyze disk structure
3. Check volume for readability and validity of Files-11 structure information
4. Dismount volume to defragment INDEXF.SYS
5. Defragment INDEXF.SYS on volume
6. Mount volume privately to analyze disk structure
7. Check volume for readability and validity of Files-11 structure information
8. Dismount volume
9. Mount volume into the cluster

DEFRAGMENT SHOW

DEFRAGMENT SHOW — The SHOW subcommand displays information about the fragmentation state of the given volume.

Format

DEFRAGMENT SHOW [{ volume-name } { /FILE_LIST | /FREE_LIST | /HISTOGRAM | /LOCATION | /OUTPUT | /STATISTICS }]

Parameters

volume-name

The parameter for the SHOW subcommand is a volume name. The SHOW command displays information about the fragmentation state of the specified volume.

Description

Information displayed as a result of showing the volume fragmentation state includes:

- A list of files with a user-selected minimum level of fragmentation
- A histogram of the file fragmentation and free space fragmentation state of the volume
- A list of all free space extents on the volume
- Summary statistics on both file fragmentation and free space fragmentation
- The location of user selected files

All information in the volume fragmentation analysis is optional. By default, only the summary statistics are provided. If the volume is mounted with the /NOWRITE qualifier, you cannot use the DEFRAGMENT SHOW command.

Restrictions

None.

Qualifiers

/FILE_LIST=(min_extents)

/FILE_LIST=(VOL_MAP_WINDOW + 1) (default)

The /FILE_LIST qualifier shows the fragmentation state of a volume. It lists all files on the volume that have *min_extents* or more extents (fragments). By default, the number of file window mapping pointers set for the volume (INITIALIZE{volume}/WINDOW=n) plus one is used. This value lists all files that require at least one window turn to completely map.

/FREE_LIST

/NOFREE_LIST (default)

The /FREE_LIST qualifier shows the fragmentation state of the volume. It lists the location and size of all of the free space extents on the volume.

/HISTOGRAM

/NOHISTOGRAM (default)

The /HISTOGRAM qualifier shows the fragmentation state of a volume. It generates two histograms. The first plots the number of files against the number of extents per file and provides information about how many files are fragmented and how badly they are fragmented. The second histogram plots the number of free space extents (holes) against the size of these free space extents. It provides information about how many free space extents are on the volume and their sizes.

/LOCATION=(file-spec[,...])

/NOLOCATION

The /LOCATION qualifier shows the fragmentation state of the volume. It prints the extents (fragments) associated with all specified files. If no files are specified, the /LOCATION qualifier

appends location information to all files listed with the /FILE_LIST qualifier. The supplied file-spec may contain wildcards in place of the directory, file name, extension, or version fields.

/OUTPUT={file-spec}
/OUTPUT=SYS\$OUTPUT (default)

The /OUTPUT qualifier may be used when showing the volume fragmentation state. It directs the output of the SHOW subcommand to a file. By default, output is directed to SYS\$OUTPUT. If output is directed to a filename without extension, the file is given a default extension of .lis.

/STATISTICS (default)
/NOSTATISTICS

The /STATISTICS qualifier shows the volume fragmentation state. It provides a summary of the file and free space fragmentation for the volume. Refer to *Appendix C, "DEFRAGMENT SHOW Command Report"* for more information on this report.

Examples

The following examples illustrate the SHOW subcommand and its qualifiers.

1. `$ DEFRAGMENT SHOW MY_VOLUME/OUTPUT=OUTPUT.TXT`

In this example, information associated with MY_VOLUME is displayed. Output is directed to a file called OUTPUT.TXT.

Note

If output is directed to a filename without extension, the file is given a default extension of .lis.

2. `$ DEFRAGMENT SHOW DISK$ONE/HISTOGRAM`

In this example, the volume fragmentation report includes the summary statistics (/STATISTICS is default) and histograms for both file fragmentation and free space fragmentation. The report is sent to the SYS\$OUTPUT device.

DEFRAGMENT SPAWN

DEFRAGMENT SPAWN — The SPAWN subcommand creates a subprocess and executes a DCL command in the subprocess, or transfers terminal control to the subprocess if no command is supplied.

Format

DEFRAGMENT SPAWN [{ command }]

Parameters

command

This is a DCL command string of less than 132 characters that is to be executed in the context of the created subprocess. When the command completes, the subprocess terminates and control returns to the parent process. A command is optional. If no command is supplied, control passes to DCL.

Description

The SPAWN subcommand creates a subprocess of your current process.

You should use the DCL LOGOUT command to terminate the subprocess and return to the parent process. You can also use the DCL ATTACH command to transfer control of the terminal to another process in the subprocess tree, including the parent process, (the Defragment Utility).

If the DFO SPAWN command is invoked with a DCL command specified as the *command* parameter, that command is executed in the subprocess, and control returns immediately to the parent process.

If the *command* parameter is not specified, then control is passed to the created DCL subprocess until it is deleted or until an ATTACH command is issued to transfer control to another process.

Restrictions

None.

Examples

The following examples show how to SPAWN processes from within the Disk File Optimizer utility.

1.

```
$ DEFRAGMENTDFO> SPAWN
$
```

The SPAWN command creates a subprocess and transfers terminal control to the created subprocess. Since no particular command was entered in this example, SPAWN sends control to DCL.

2.

```
DFO> SPAWN SHOW DEFAULT    DISK$1:[JONES]
DFO>
```

The SPAWN command creates a subprocess and executes the DCL command SHOW DEFAULT in the created subprocess. Control then returns to the Disk File Optimizer parent process.

DEFRAGMENT VOLUME

DEFRAGMENT VOLUME — The VOLUME subcommand defragments the volume given as a parameter on the command line according to the defragment command line qualifiers.

Format

```
DEFRAGMENT VOLUME [{ volume-name } { /BRIEF | /CONSOLIDATE_FREESPACE
| /CPU_LIMIT | /DORMANT | /EPILOGUE | /EXCLUDE | /FREQUENT | /FULL | /
HOTFILE_DATABASE | /IGNORE | /IO_LIMIT | /LEVEL | /LOG | /OPCOM_NOTIFICATION | /
OPTIONS | /PRIORITY | /PROLOGUE | /RVN | /TIME_LIMIT | /WAIT | /WRITE_CHECK }]
```

Parameters

volume-name

The volume-name parameter is the name of the volume to be defragmented.

Description

The VOLUME subcommand defragments the specified volume according to the selected command line qualifiers. The defragmentation process begins executing immediately and executes only once.

Restrictions

- The /LOG qualifier is required if /LEVEL=5 is selected.
- /BRIEF and /FULL cannot be used concurrently.

Qualifiers

/BRIEF (default)

/NOBRIEF

The /BRIEF qualifier has meaning only when the /LOG qualifier is also present. The /BRIEF qualifier requests a brief log of all actions taken by the defragmentation process. This log includes beginning and ending volume status information. The /BRIEF and /FULL qualifiers are mutually exclusive, and /NOBRIEF produces the same result as /FULL. The /BRIEF qualifier is the default.

/CONSOLIDATE_FREESPACE (default)

/NOCONSOLIDATE_FREESPACE

The /CONSOLIDATE_FREESPACE qualifier tells the Disk File Optimizer to use its free space consolidation algorithm.

/CPU_LIMIT=n

/NOCPU_LIMIT

The /CPU_LIMIT qualifier specifies the percent of CPU resources to be used for any one defragmentation process. If this percentage is exceeded, the priority for the process drops to the minimum, as specified by the /PRIORITY qualifier, or defaults to three.

/DORMANT=(file-spec[,...])

/NODORMANT (default)

The /DORMANT qualifier indicates files to be considered dormant when final file placement decisions are made. Dormant files are placed at the high end of the volume.

If a file is indicated in both the DORMANT and FREQUENT lists, it is considered dormant. If a file is indicated in both the EXCLUDE and DORMANT lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

/EPILOGUE={file-name}

/NOEPILOGUE (default)

The /EPILOGUE qualifier identifies a DCL command file which DFO executes upon completion of the DEFRAGMENT operation.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/EXCLUDE=(file-spec[,...])

/NOEXCLUDE (default)

The /EXCLUDE qualifier specifies files to be excluded from processing. The defragmentation process ignores files matching the listed file specifications. Any file indicated in the EXCLUDE list and in either the FREQUENT or DORMANT lists is excluded from processing.

Note

The defragmentation process always excludes system files, open files, files excluded in an options file (/OPTIONS qualifier), and files excluded using the /IGNORE qualifier.

/FREQUENT=(file-spec[,...])

/NOFREQUENT (default)

The /FREQUENT qualifier indicates files to be considered frequently accessed when final file placement decisions are made. Frequent files are placed toward the center of the volume.

If a file is indicated in both the FREQUENT and DORMANT lists, it is considered dormant. If a file is indicated in both the FREQUENT and EXCLUDE lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

/FULL

/NOFULL (default)

The /FULL qualifier requests a full log of all actions taken by the defragmentation process. The /FULL qualifier must be used with the /LOG qualifier. The /FULL and /BRIEF qualifiers are mutually exclusive, and /NOFULL provides the same functionality as /BRIEF.

/HOTFILE_DATABASE

/NOHOTFILE_DATABASE (default)

The /HOTFILE_DATABASE qualifier causes the defragmenter to query the appropriate hotfile database and append the list of hotfiles to the defragmenter's /FREQUENT file list.

/IGNORE=(keyword[,...])

/IGNORE=(PLACED)(default)

/NOIGNORE

The /IGNORE qualifier requests that the defragmentation process ignore certain types of files as specified by the following keywords:

- INDEXED—Ignores all RMS indexed files
- NOINDEXED—May defragment RMS indexed files
- PLACED—Ignores all files with explicit placement flags set in their headers
- NOPLACED—May defragment files with explicit placement flags

Files indicated by the `/IGNORE` qualifier are added to the `EXCLUDE` list.

The default value (`/IGNORE=PLACED`) indicates that files with explicit placement control should be ignored, and that RMS indexed files should be defragmented and placed in appropriate locations on the volume.

`/IO_LIMIT=n`

`/NOIO_LIMIT`

The `/IO_LIMIT` qualifier specifies the maximum direct I/O rate (per second) of any one defragmentation process. If the I/O rate is exceeded, the priority for the process drops to the minimum priority as specified by the `/PRIORITY` qualifier or defaults to three.

`/LEVEL={n}`

`/LEVEL=2 (default)`

The defragmentation process selects files for defragmentation (from among those not excluded) based on the `/LEVEL` qualifier. Once a file is selected as a candidate, the defragmentation process attempts to fully defragment that file (make it 100 percent contiguous). Specify the level number as an integer ranging from one to five:

- **`/LEVEL=1`**

Any file that is not 100 percent contiguous, or that can be selected using levels two and three, is a candidate for defragmentation.

- **`/LEVEL=2 (default)`**

Any file whose extents cannot be completely mapped by the mapping window, or that can be selected using level three, is a candidate for defragmentation. The default window value is seven and can be set for a given volume using:

```
$ SET VOLUME {device-spec}/WINDOWS=[n]
```

- **`/LEVEL=3`**

Any file with retrieval pointers in more than one file header is a candidate for defragmentation.

- **`/LEVEL=4`**

The defragmentation process for level four processing performs only free space consolidation. This level is used with the `/CONSOLIDATE_FREESPACE` qualifier.

- **`/LEVEL=5`**

This level requests a volume analysis only; no defragmentation occurs. The statistical results are available in the log file, which is required when `/LEVEL=5` is selected (see the `/LOG` qualifier). If level 5 is selected, the `/LEVEL=2` candidate file specifications are listed in the full log file.

`/LOG={file-spec}`

`/NOLOG (default)`

The `/LOG` qualifier causes the defragmentation process to create a file containing a list of all its operations. The information in the log file varies with the presence of the `/FULL` or `/BRIEF` qualifiers. The `/FULL` qualifier lists all files analyzed by the defragmentation process and reports their status during each phase of defragmentation. The `/BRIEF` qualifier lists only the statistics of each phase, such as number of files analyzed, and number of files defragmented.

/OPCOM_NOTIFICATION (default)
/NOOPCOM_NOTIFICATION

The /OPCOM_NOTIFICATION qualifier requests that brief status and termination messages regarding defragmentation processes be sent to OPCOM.

/OPTIONS={file-spec}
/NOOPTIONS (default)

The /OPTIONS qualifier provides an alternate method for associating files with the /DORMANT, /EXCLUDE, or /FREQUENT qualifiers. If the list of files for any of these qualifiers is long, you can create a separate file that contains this information. Each record in the options file consists of a file name followed by one of the following keywords:

- /EXCLUDE—File to be excluded
- /FREQUENT—File to be considered frequently accessed
- /DORMANT—File to be considered dormant

The following example shows the contents of an options file called MYOPTFILE.OPT. This file specifies that all files on the target device called FILE1.EXE are frequent files, all files in the [ARCHIVE] directory are to be excluded, and all files in the [OLDSTUFF] directory and its subdirectories are dormant.

```
FILE1.EXE/FREQUENT
[ARCHIVE]*.*;*/EXCLUDE
[OLDSTUFF...]*.*;*/DORMANT
```

Files indicated by the /EXCLUDE qualifier and by either the /DORMANT or /FREQUENT qualifier are excluded. Files indicated by the /DORMANT and /FREQUENT qualifiers are considered dormant. This rule is applied equally across all files specified in the /OPTIONS file on the command line.

/PRIORITY=(DEFAULT=n, MINIMUM=n)
/NOPRIORITY (default)

The /PRIORITY qualifier specifies priorities for the defragment process. The minimum priority setting is used when the defragmentation process exceeds the value specified by either /CPU_LIMIT or /IO_LIMIT.

/PROLOGUE={file-name}
/NOPROLOGUE (default)

The /PROLOGUE qualifier identifies a DCL command file which DFO executes just before the Defragment operation starts.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/RVN=n
/RVN=0 (default)

When you have a bound volume set, use this qualifier to identify a disk from the volume set or all disks in the volume set. The default of zero tells the Disk File Optimizer to defragment all disks in

the bound volume set. Specify a relative volume number (RVN) to defragment a specific disk that belongs to a bound volume set.

/TIME_LIMIT={delta-time}

/NOTIME_LIMIT (default)

The /TIME_LIMIT qualifier specifies the maximum elapsed time allowed for a given defragmentation operation. If the time limit is exceeded, the defragmentation process is aborted.

Specify time as a delta time. Refer to the *VSI OpenVMS User's Manual* for more information on specifying times.

If the time limit is set unreasonably low, the defragmentation process may not accomplish useful defragmentation.

/WAIT (default)

/NOWAIT

The /WAIT qualifier causes the input device to wait until the defragmentation process finishes before returning control to the user.

/WRITE_CHECK

/NOWRITE_CHECK (default)

The /WRITE_CHECK qualifier requests that a read-after-write check be performed on all defragmented or placed files. This qualifier significantly extends the execution time of a defragmentation process.

Examples

The following example shows how to defragment a volume.

```
$ DEFRAGMENT VOLUME DISK$USER_1
```

This command defragments DISK\$USER_1 using default qualifiers. The defragmentation process executes once.

Chapter 4. Using Defrag PLUS

4.1. Introduction

This chapter explains how to use the Defrag PLUS variant of the Disk File Optimizer software. It contains these major sections:

- General guidelines for using the Defrag PLUS software
- Information about storing defragmentation operation information in database scripts
- Information about scheduling defragmentation operations
- Defrag PLUS command reference material in a format similar to that of the *VSI OpenVMS DCL Dictionary*

4.2. General Guidelines

Defrag PLUS provides all the functions of Defrag, plus the ability to:

- Store sets of defragmentation operations
- Schedule defragmentation operations

4.2.1. DCL Interface

Disk File Optimizer commands are available from the DCL prompt or from within the Disk File Optimizer utility. When you enter a command from within the Disk File Optimizer, you do not enter the word “DEFRAGMENT” first, except for the DEFRAGMENT FILE and DEFRAGMENT VOLUME commands.

Examples of some defragmentation commands entered at the DCL prompt are:

```
$ DEFRAGMENT FILE *.C /FULL/LOG=defrag_c.log/SCRIPT=c/AFTER=22:00
$ DEFRAGMENT MODIFY c /NOAFTER/FILES=*.OBJ
$ DEFRAGMENT SHOW c
$ DEFRAGMENT REMOVE c
```

The same commands entered from within the Disk File Optimizer utility would look like this:

```
$ DEFRAGMENT
DFO> DEFRAGMENT FILE *.C /FULL/LOG=defrag_c.log/SCRIPT=c/AFTER=22:00
DFO> MODIFY c /NOAFTER/FILES=*.OBJ
DFO> SHOW c
DFO> REMOVE c
```

4.2.2. Wildcard Use

The file-spec field allows wildcard characters. You may specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*.*. Specifying *.TMP;*selects all *.TMP;* files on the entire volume.)

4.2.3. Graphic User Interface

The Disk File Optimizer graphic user interface GUI is available by entering the following command at the DCL prompt:

```
$ DEFRAGMENT / INTERFACE=DECWINDOWS
```

You can explicitly select the character-cell interface by entering this command at the DCL prompt:

```
$ DEFRAGMENT / INTERFACE=CHARACTER_CELL
```

4.2.4. Online Help

If you need a reminder of the Disk File Optimizer command syntax while working on the system, you can access online help at the DCL prompt or from within the Disk File Optimizer utility. To access online help, enter the following at the DCL (\$) prompt:

```
$ HELP DEFRAGMENT
```

or

```
$ DEFRAGMENT HELP
```

To access online help from within the Disk File Optimizer utility, enter the following command:

```
DFO> HELP
```

4.2.5. Quickness and Effectiveness of Defragmentation

How quickly and effectively the defragmentation process defragments a volume depends on a number of variables, such as:

- The amount of free space on the volume.

If all other variables are constant, a volume with little free space takes longer to defragment than a volume with a lot of free space.

- The number of users accessing files on the volume.

Because the defragmentation process does not impact a user's ability to read and write files, it cannot defragment a file that is being accessed. Thus a large number of user file accesses on the target volume can negatively impact the effectiveness of the defragmentation process.

- User-set CPU or I/O limits.

In general, the more restrictive the limits, the longer it takes to defragment a given volume.

4.2.6. Cluster Management

By default, all DEFRAGMENT subcommands have systemwide or clusterwide scope. For example, an ABORT subcommand entered on one node of a cluster, effects all nodes. Where appropriate, you may limit the scope of a command by using the /NODE qualifier.

By specifying the node name, you limit defragmentation process creation to the specified node. Thus, the actual defragmentation work occurs on the dedicated node, leaving the other nodes in the cluster available for their usual workload. With this scheme, only one scheduler process is required. Scheduler

processes are only required on the nodes that execute defragmentation processes. This scheme does not inhibit command processing in any way. DEFRAGMENT commands can be entered and processed from any node in the cluster.

Another option is to use all nodes in the cluster. This is the default for Disk File Optimizer software. By using the DEFRAGMENT SET command to set the PROCESS_LIMIT parameter, you can limit the number of concurrent defragmentation processes on a node-by-node basis. Thus, some nodes may execute any number of concurrent defragmentation processes while others may execute a limited number of concurrent defragmentation processes.

If the target volume for defragmentation is not available clusterwide, use the /NODE qualifier. Specify the name of the node from which the volume is accessible.

Note

If you enter a DEFRAGMENT command for a device that is currently unavailable, or unavailable from the issuing node, the Disk File Optimizer software issues a warning message indicating the device is unsuitable for defragmentation and asks if you wish to continue to input the command. If this command will not take effect until after the device is available, enter Y in response to this message and the command is entered into the database. If you enter N in response to this message, the command is not entered into the database. When the scheduler starts the script, the device must be mounted or the command will fail.

4.3. Scripts

Defragmentation commands are stored in the scheduling database as *scripts*. When a script is entered into the scheduling database, a permanent record of that script is created. This provides a great deal of flexibility and allows commands to be issued and executed from any node within a cluster. You can create scripts that:

- Run immediately, after a specified time, or at defined intervals
- Remain in the database for future use
- Delete after running once

4.3.1. Creating and Using Scripts

Creating permanent scripts that run periodically eliminates the need for operator intervention and prevents the build-up of fragmented files on the disk. Usually, once you set up a script to run on a routine schedule, no further intervention is required. (Refer to the DEFRAGMENT START command in Section 4.6, "Disk File Optimizer Command Set" to restart scripts after a system or node failure.)

To create a script and schedule defragmentation operations, use the DEFRAGMENT FILE or DEFRAGMENT VOLUME command. If you need to defragment a file or a small group of files, use the DEFRAGMENTFILE command. If you need to defragment a large group of files, a volume, or several volumes, use the DEFRAGMENT VOLUME command, for example:

```
$ DEFRAGMENT VOLUME volume-name[/QUALIFIERS]
```

Where:

volume-name is the name of the target device.

/QUALIFIERS defines the desired script characteristics.

The scheduling and defragmentation qualifiers determine when and how a script operates. For more information about the specific qualifiers for these commands, see *Section 4.6, "Disk File Optimizer Command Set"*, the *DEFRAGMENT SHOW* or *DEFRAGMENT DISABLE* commands.

4.3.2. Script Naming Conventions

When you create a script, use the */SCRIPT* qualifier to assign a script name of up to 64 characters that is meaningful to you. Observe the following rules when naming a script:

- You may use any combination of uppercase and lowercase letters (a through Z).
- You may use any number (0 through 9).
- You may use the underscore (_).
- You can include the dollar sign (\$) within the script name, but not as the first character. (VSI reserves the dollar sign for special use.)
- You may use a hyphen (-) in the script name, but do not end a script name with a hyphen because the hyphen is the DCL continuation character.

4.3.3. Examples of Creating and Using Scripts

This section includes examples of analyzing volumes, creating log files, and using the scheduling and defragmentation qualifiers to defragment volumes.

4.3.3.1. Creating Log Files

Log files are created using the */LOG* qualifier. There are two types of log files:

- Brief
- Full

Brief log files provide an abbreviated description of events that occur during a defragmentation operation. Brief log files are the default log file type.

Full log files provide an extended version of events that occur during defragmentation operations. Full log files are obtained by specifying */FULL*.

Examples of brief and full log files can be found in *Appendix D, "Brief Log File Example"* and *Appendix E, "Full Log File Example"*, respectively. The command shown in the following example creates a full log file:

```
$ DEFRAGMENT VOLUME DISK$ONE/LEVEL=2/LOG=DISK$ONE_ANA.LOG/FULL
```

In log files, the symbols preceding a file name indicate the following:

Blank	- file has been preprocessed for defragmentation
d	- file has been defragmented
*	- file has been partly defragmented
D	- file has been placed for dormant access
Dd	- file has been defragmented and placed dormant

D*	- file has been placed dormant and been partly defragmented
F	- file has been placed for frequent access
Fd	- file has been defragmented and placed frequent
F*	- file has been placed frequent and been partly defragmented
Moved	- file has been moved (number of extents unchanged)
EMPTY	- file was empty; no action taken
FAIL	- file's fragmentation cannot be reduced
IN USE	- new disk space for file was in use; no action taken
OPEN	- file was open; no action taken
NOMOVE	- file may not be moved; no action taken
UNAVL	- file was unavailable; no action taken

4.3.3.2. Showing the State of a Disk

The defragmentation software can show the current fragmentation state of a volume. The SHOW subcommand provides a variety of report options which show the level of fragmentation on a given volume.

For example, to view the default information in the volume fragmentation report, enter the following command:

```
$ DEFRAGMENT SHOW DISK$ONE/VOLUME
```

This command generates a fragmentation index, free space summary, and file fragmentation summary. The fragmentation index is an estimate of the level of fragmentation on the specified volume. Its value is derived from the number of files on the volume, their sizes, the number of extents in those files, and the state of the free space on the volume. The file and free space fragmentation summary provide more detail on the fragmentation index. Refer to *Appendix C, "DEFRAGMENT SHOW Command Report"* for more information on this report.

Other options available show file and free space histograms, list all free space holes, list all files with more than a given number of extents, and provide locations for specified files.

4.3.3.3. Creating a Named, Periodic Script

The following command is an example of a defragmentation operation. Qualifier values can be changed to meet your needs.

```
$ DEFRAGMENT VOLUME DISK$ONE/SCRIPT=DEFRAG_1/LEVEL=2-  
_$/LOG=DEFRAG_1.LOG/INTERVAL=7-0/NODE=NODE_1
```

This command assigns a name of DEFRAG_1 to this script. Candidate files are selected on the basis of whether they meet LEVEL 2 file qualifications. A log file will be written to DEFRAG_1.LOG.

Additionally, because no start time was indicated, script DEFRAG_1 begins executing immediately and will execute at the same time every seventh day (/INTERVAL=7-0) thereafter.

DEFRAG_1 is restricted to execute on NODE_1 in the VAXcluster system. Finally, because no time limit was given in this command, the defragmentation process will execute until it completes.

4.3.3.4. Copying Scripts

The /INHERIT qualifier is used to copy a set of script characteristics. Using the /INHERIT qualifier, you can create new scripts, based on existing scripts.

When using the /INHERIT qualifier, a script's characteristics, such as start time (/AFTER), excluded files (/EXCLUDE), and so forth, are derived from a combination of the script given as a value to the /INHERIT qualifier and any additional command line qualifiers. To prohibit a script from inheriting a given characteristic, negate the associated command line qualifier by providing a new value. For example, to prohibit a script from inheriting the starting time of the inherited script, specify /NOAFTER or /AFTER={time-spec}.

The following example shows how to create a new script (DEFRAG_2) based on an existing script (DEFRAG_1):

```
$ DEFRAGMENT VOLUME DISK$TWO/SCRIPT=DEFRAG_2-  
_$_ /INHERIT=DEFRAG_1/LOG=DEFRAG_2.LOG
```

Note

The Disk File Optimizer software will not allow two scripts to defragment the same volume at the same time.

The DEFRAG_2 script will inherit all DEFRAG_1 characteristics except the target volume, script name, and log file.

4.3.3.5. Modifying Scripts

Once you create a script and store it in the database, you can use it over and over again. You may want to create a script that defines the important defragmentation commands and run it periodically against a different set of files. Defrag PLUS provides a MODIFY command that lets you modify an existing script. The following example shows how to modify a script:

```
$ DEFRAGMENT MODIFY DEFRAG_2 /FREQUENT=(*.EXE,*.C)/NOLOG
```

In this example, DEFRAG_2 is modified so that it has a list of frequent files and no longer produces a log file.

For additional examples of modifying scripts, see *Section 4.3.3.6, "Linking and Modifying Linked Scripts"*.

4.3.3.6. Linking and Modifying Linked Scripts

Use the /PREDECESSOR qualifier to link two or more scripts so they execute one after another. A script identified as a predecessor script must first complete executing before its successor script can begin executing. The following is an example of a linked script:

```
$ DEFRAGMENT VOLUME DISK$TWO/SCRIPT=DEFRAG_2-  
_$_ /PREDECESSOR=DEFRAG_1/LOG=DEFRAG_2.LOG
```

In this case, DEFRAG_1 must complete execution before DEFRAG_2 can begin executing. If you entered another command and specified /SCRIPT=DEFRAG_3/PREDECESSOR=DEFRAG_2, then the scripts would execute in order: DEFRAG_1, DEFRAG_2, DEFRAG_3.

Modifying scripts that are part of a linked set of scripts sometimes involves modifying the linked set. For example, assume two sets of linked scripts: (A,B,C) and (D,E,F). The following examples and descriptions illustrate how these links can be modified.

```
$ DEFRAGMENT MODIFY B /PREDECESSOR=D
```

When the predecessor field of a script that is part of a linked set of scripts is modified, that script is first removed from any linked set of scripts. After it is removed, the script is placed in the new linked set as indicated by the /PREDECESSOR qualifier. Thus, B is removed from its linked set of (A,B,C) and made the successor of D. Two linked sets of scripts are now present (A,C) and (D,B,E,F).

To remove a script from a linked set of scripts, specify the /NOPREDECESSOR qualifier, as shown in this example:

```
$ DEFRAGMENT MODIFY B /NOPREDECESSOR
```

This command removes B from its linked set of scripts. From the previous example, the linked scripts are now (A,C) and (D,E,F). Script B is no longer part of a linked set of scripts and operates according to its own scheduling characteristics.

The following example places script B back in its original position:

```
$ DEFRAGMENT MODIFY B /PREDECESSOR=A
```

After executing this command, there are again two sets of linked scripts (A,B,C) and (D,E,F).

4.3.3.7. Saving Database Contents

The entire (or selected) contents of the database may be saved in a command file. The following command saves the entire database in a file called MY_DB_CONTENTS.COM:

```
$ DEFRAGMENT SHOW/ALL/PARAMETERS/CMDLINE/OUTPUT=MY_DB_CONTENTS.COM
```

MY_DB_CONTENTS.COM now contains a list of valid defragment command lines that describes all the scripts and I/O, process, and CPU limits found in the database.

4.3.3.8. Re-creating the Database

Section 4.3.3.7, "Saving Database Contents" describes how to save the contents of the database in a file called MY_DB_CONTENTS.COM. With this file, you can re-create the database using the following commands:

```
$ RUN SYS$SYSTEM:DFG$CREATE_DATABASE
$ RUN SYS$SYSTEM:DFG$INIT_DATABASE$ @MY_DB_CONTENTS
```

If you have a database and want to erase its contents, enter the following command:

```
$ RUN SYS$SYSTEM:DFG$INIT_DATABASE
$ @MY_DB_CONTENTS
```

Caution

Stop all scheduler processes before you initialize the database by executing SYS\$MANAGER:DFG\$STOP_SCHEDULER.COM on all nodes that have running schedulers. If you fail to stop the scheduler before running SYS\$SYSTEM:DFG\$INIT_DATABASE.EXE, the scheduler will fail and report an inconsistent database message.

4.3.4. Using Hotfiles Information

The Disk File Optimizer allows you to automatically use hotfile records from the VPA or DECps database to identify frequently accessed files. The defragmentation process automatically determines

which of these products is installed on your system and queries the appropriate database. To augment the frequent files list using the hotfiles records from one of these products, use the /HOTFILE_DATABASE qualifier with the DEFRAGMENT FILE or DEFRAGMENT VOLUME command.

If you use the /HOTFILE_DATABASE qualifier, the defragmentation process first queries the appropriate database for hotfile records relating to the target volume. The files listed in the hotfile records are automatically added to the /FREQUENT file list.

The time over which the hotfile data is sampled depends on the scheduling information on the DEFRAGMENT command line, as described in *Table 4.1, "Sample Time Interval"*.

Table 4.1. Sample Time Interval

Selected Scheduling	Result
None selected	Sample data over the previous 24 hours.
/INTERVAL=X	Sample data over the previous X time period.
/PREDECESSOR=X	Search through the list of predecessors. If a predecessor is found with an /INTERVAL time, use that time. If no /INTERVAL is found, use the previous 24 hours.

Both VPA and DECps allow you to modify the hotfile collection process. If you have VPA installed, use the ADVISE /COLLECT command and /[NO]HOTFILE_QUEUE={ value } qualifier. If you have DECps installed, use the ADVISE COLLECT SET command and the / [NO]HOTFILE_QUEUE={ value } qualifier. See the VPA or DECps documentation for additional details.

4.3.5. Defining Quotas for the Defragmentation Process

The process quotas are values applied to the defragmentation process and then disappear when the process is complete. For more information on process quota values, refer to the *VMS Authorize Utility Manual*.

You can define the process quotas for defragmentation processes. When the defragmentation process starts up it selects the maximum value from the following three sources:

- Hard coded values
- SYSGEN process quotas (PQL_Mxxx)
- DFO logical process quotas

By changing the SYSGEN process quotas or DFO logical process quotas you can define the process quotas for the defragmentation processes. The following sections describes these values and describes how to change them.

4.3.5.1. Hard coded values

You cannot change the hard coded values for the defragmentation processes but the following shows the hard coded values:

- ASTLIM - 100

- BIOLM - 18
- BYTLM - 61440
- DIOLM - 18
- ENQLM - 500
- FILLM - 50
- JTQUOTA - (uses system default)
- PGFLQUOTA - 50000
- PRCLM - 8
- TQELM - 10
- WSDEFAULT - 512
- WSEXTENT - 16000
- WSQUOTA - 2048

4.3.5.2. SYSGEN Process Quotas

The SYSGEN PQL_Mxxx process quotas are defined in SYSGEN. You can change these dynamic PQL_Mxxx parameters in SYSGEN. For more information on SYSGEN process quota values, refer to the SYSGEN Generation Utility.

This list shows the PQL_Mxxx parameters that affect the defragmentation processes:

- PQL_MASTLM
- PQL_MBIOLM
- PQL_MBYTLM
- PQL_MDIOLM
- PQL_MENQLM
- PQL_MFILLM
- PQL_MJTQUOTA
- PQL_MPGFLQUOTA
- PQL_MPRCLM
- PQL_MTQELM
- PQL_MWSDEFAULT

- PQL_MWSEXTENT
- PQL_MWSQUOTA

4.3.5.3. DFO Logical Process Quotas

DFO provides logicals that allow you to define the process quotas for the defragmentation process. These logicals are defined in SYS\$STARTUP:DFG\$STARTUP.COM. When you install DFO these logicals are commented out. By uncommenting the affected line and executing DFG\$STARTUP.COM you can affect the quotas for the defragmentation processes.

For example, if a disk containing a large database routinely encounters the insufficient virtual memory error (INSVIRMEM) during defragmentation, increase the page file quota (PGFLQUOTA) to a value larger than the DFO default of 50,000.

```
$ DEFINE/SYSTEM/EXECUTIVE_MODE DFG$PGFLQUOTA 120000
```

These rules apply to setting process quota values:

- You cannot decrease a quota value below the system PQL minimum as defined by SYSGEN.
- You may decrease a DFO quota value to the associated PQL_Mxxx minimum value, but this is not recommended.

The right way to change these logicals is to edit the file SYS\$STARTUP:DFG\$STARTUP.COM. An example of changing the page file quota is shown below:

```
$! To use these logicals, increase the associated value, uncomment the
$! desired line, and execute this file.
$!
$! CALL DEFINE_LOGICAL DFG$ASTLM      100
$! CALL DEFINE_LOGICAL DFG$BIOLM      18
$! CALL DEFINE_LOGICAL DFG$BYTLM      61440
$! CALL DEFINE_LOGICAL DFG$DIOLM      18
$! CALL DEFINE_LOGICAL DFG$ENQLM      500
$! CALL DEFINE_LOGICAL DFG$FILLM      50
$! CALL DEFINE_LOGICAL DFG$JTQUOTA    {uses system default}
$ CALL DEFINE_LOGICAL DFG$PGFLQUOTA  120000
$! CALL DEFINE_LOGICAL DFG$PRCLM      8
$! CALL DEFINE_LOGICAL DFG$TQELM      10
$! CALL DEFINE_LOGICAL DFG$WSDEFAULT  512
$! CALL DEFINE_LOGICAL DFG$WSEXTENT   16000
$! CALL DEFINE_LOGICAL DFG$WSQUOTA    2048
```

4.3.5.4. Defining the Transfer Size Logical

The logical, DFG\$TRANSFERSIZE, helps you tune the Disk File Optimizer. The definition of this logical allows you to change the amount of data that is transferred at onetime. On a volume that has files with large extents, you could wait up to 20 minutes while the system transfers these large extents before you could access the file. By defining this logical, you could access this file in less than a minute. This logical changes the impact that the Disk File Optimizer has on applications that are simultaneously accessing the same files.

This logical is set to 10,000 blocks by default. You can change it by editing SYS\$STARTUP:DFG\$STARTUP.COM and then executing the command procedure.

4.4. Scheduling Defragmentation Operations

Defrag PLUS allows you to execute named scripts once, every day at a specified time, on a particular date, or according to any periodic interval. You cannot, however, define a single script to run Monday at 4:00, Tuesday at 7:00, and Friday at 9:00. To do this, you would either have to modify the script daily, or create three scripts.

Consider the following suggestions when determining the best time to run the defragmentation process.

- Schedule defragmentation during periods of low user activity. This step ensures the defragmentation process can defragment all candidate files. If a user opens a file while the defragmentation process is running, the process stops. Use VPA or DECps software to find periods of low use activity.
- Schedule the defragmentation process periodically to maintain volumes in a defragmented state. This also makes the defragmentation process more effective.
- Schedule defragmentation before and after user activity that may significantly contribute to fragmentation, such as end-of-month processing.

4.5. Error Recovery Procedures

Appendix F, "Defragmentation Messages" lists all the messages that the Disk File Optimizer generates. The messages are in alphabetical order and contain a description and suggested user action. However, if a system failure occurs while defragmentation is in process the following procedure ensures any lost extension file headers are cleared up:

1. Enter the MOUNT /REBUILD command—This is the default action after a system failure.
2. Enter the ANALYZE/DISK_STRUCTURE command—If you receive the LOSTEXTHDR message, use the /REPAIR qualifier to eliminate the problem.

The LOSTEXTHDR error indicates that an extension file header is not in the extension file header chain of any file. The file header is rewritten with a deleted header, and the blocks allocated to the file header are marked free.

Note

The LOSTEXTHDR error does not affect file access.

If a linked script is executing when a system failure occurs, you must start the successors to that script manually using the DEFRAGMENT START command. Alternately, if the linked set of scripts executes on a periodic schedule, you can wait for the next period when all the scripts in the linked set execute automatically.

4.6. Disk File Optimizer Command Set

The Defrag PLUS command set provides the system manager with a full range of scheduling and execution options for file and volume defragmentation. Scripts can execute once or on a periodic schedule, depending on the command qualifiers selected. *Table 4.2, "Defrag PLUS Command Summary"* briefly describes the Defrag PLUS commands.

Table 4.2. Defrag PLUS Command Summary

Command	Description
DEFRAGMENT ABORT {script-name[,...]} volume-name[,...]} [/QUALIFIERS...]	Aborts the defragmentation process associated with the specified script. Before exiting, the defragmentation process performs an orderly shutdown of all outstanding activities.
DEFRAGMENT ATTACH {process_name} [/QUALIFIERS...]	Transfers control from the current process (which goes into hibernation) to the specified process.
DEFRAGMENT CANCEL {script-name[,...]} [/QUALIFIERS...]	Cancels the next scheduled execution of the specified script. Normal scheduling resumes after the next scheduled execution.
DEFRAGMENT DISABLE {script-name[,...]} [/QUALIFIERS...]	Disables the specified script. A script does not execute while it is disabled. The script remains disabled until it is enabled with the ENABLE subcommand.
DEFRAGMENT ENABLE {script-name[,...]} [/QUALIFIERS...]	Enables a previously disabled script. Once enabled, a script operates normally.
DEFRAGMENT EXIT	Exits the user from the Disk File Optimizer utility and returns to the system prompt.
DEFRAGMENT FILE {file_name[,...]} [/QUALIFIERS...]	Defragments a selected file (or files).
DEFRAGMENT HELP {command}	Invokes VMS HELP utility for the Disk File Optimizer software.
DEFRAGMENT MODIFY {script-name} [/QUALIFIERS...]	Modifies an existing script.
DEFRAGMENT MONITOR {script-name volume-name} [/QUALIFIERS...]	Invokes the defragmentation monitor, which monitors an ongoing defragmentation process. The input parameter can be either a script or a volume name, depending on the setting of the /VOLUME qualifier.
DEFRAGMENT OFFLINE_VOLUME {volume-name[,...]} [/QUALIFIERS...]	Defragments INDEXF.SYS on the specified volume according to the selected command line qualifiers.
DEFRAGMENT REMOVE {script-name[,...]} node-name[,...]} [/QUALIFIERS...]	Removes the specified script or node name from the scheduling and parameter database.
DEFRAGMENT SET {(parameter=value[,...])} [/NODE=(node[,...])]	Sets the specified parameters (CPU_LIMIT, IO_LIMIT, PROCESS_LIMIT) to the values specified.
DEFRAGMENT SHOW [{script-name[,...]} {volume-name[,...]}] [/QUALIFIERS...]	Shows the contents of the scheduling and parameter database or the fragmentation state of the supplied volumes.
DEFRAGMENT SPAWN {command}	Creates a subprocess and either transfers control to the subprocess or executes a DCL command from within the subprocess.

Command	Description
DEFRAGMENT START { script-name[,...]} [/QUALIFIERS]	Starts the specified script at the time specified by the /AFTER qualifier. By default, the script starts immediately.
DEFRAGMENT VOLUME { volume_name[,...]} [/QUALIFIERS...]	Defragments a selected volumes. The DEFRAGMENT VOLUME command creates and enters a script into the scheduling database. This script defragments the specified volume. By default, the script is immediately eligible for execution.

4.6.1. Defrag PLUS Commands

This section provides detailed information about each command for the Defrag PLUS variant of the Disk File Optimizer software.

DEFRAGMENT ABORT

DEFRAGMENT ABORT — The ABORT subcommand terminates the execution of the script or scripts specified in the command line.

Format

DEFRAGMENT ABORT [{ script-name[,...] } { /AFTER | /ALL }]

Parameters

[script-name]

The script-name parameter is the name of the script to be aborted. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

Description

The ABORT subcommand terminates execution of the defragmentation process(es) associated with the scripts specified on the command line. The process is terminated in an orderly fashion. The defragmentation process completes current functions, executes housekeeping functions, and then exits, leaving the volume fully usable and accessible. If the specified script is not executing, a message indicating that the script is not executing is returned to the user.

Restrictions

You cannot use the script-name parameter with the /ALL qualifier.

Qualifiers

/AFTER={time}

/NOAFTER (default)

The /AFTER qualifier aborts the script after the specified time.

You can specify time as either an absolute time or as a combination of absolute and delta times. For complete information on specifying time values, see the *VSI OpenVMS DCL Dictionary*.

/ALL

/NOALL (default)

The /ALL qualifier aborts all currently executing scripts across the system. This qualifier is equivalent to using an asterisk (*) for the script-name parameter.

Example

The following are examples of the ABORT command:

1. `$ DEFRAGMENT ABORT MY_SCRIPT/AFTER=16:30`

This command aborts MY_SCRIPT after 16:30 on the day this command is submitted.

2. `$ DEFRAGMENT ABORT/ALL`

This command immediately aborts all currently executing scripts in the entire system.

DEFRAGMENT ATTACH

DEFRAGMENT ATTACH — The ATTACH subcommand transfers control from the current process (which then hibernates) to the specified process.

Format

DEFRAGMENT ATTACH [{ process-name } { /IDENTIFICATION=pid }]

Parameters

[process-name]

The process-name specifies the name of a process to which control transfers. The process must already exist, be part of your current job, and share the same input stream as your current process. However, the process cannot be your current process or a subprocess created with the /NOWAIT qualifier.

Process names can contain from 1 to 15 alphanumeric characters. If a connection to the specified process cannot be made, an error message is displayed.

Description

The ATTACH command allows you to connect your input stream to another process. You may use the ATTACH command to change control from one subprocess to another subprocess or to the parent process.

When you enter the ATTACH command, the parent or “source” process is put into hibernation, and your input stream is connected to the specified destination process. You may use the ATTACH command to connect to a subprocess that is part of a current job left hibernating as a result of the DCLSPAWN / WAIT command or another ATTACH command as long as the connection is valid. (No connection can be made to the current process, to a process that is not part of the current job, or to a process that does not exist. If any of these connections are attempted, an error message is displayed.)

You can also use the `ATTACH` command with the `DCL SPAWN /WAIT` command to return to a parent process without terminating the subprocess running the Disk File Optimizer software.

Restrictions

The `ATTACH` command cannot be used if your terminal has an associated mailbox. You cannot use the *process-name* parameter with the `/IDENTIFICATION` qualifier. You must supply either a process name or a PID for this command.

Qualifiers

`/IDENTIFICATION=pid`

`/IDENTIFICATION` lets you specify the process identification (PID) of the process to which terminal control will be transferred. Leading zeros can be omitted.

Examples

The following are examples of the `ATTACH` command:

1. `$ DEFRAGMENT ATTACH JONES_2`

This `ATTACH` command transfers the terminal's control to the subprocess named `JONES_2`.

2. `$ DEFRAGMENT ATTACH /IDENTIFICATION=2A401A1B`

This `ATTACH` command switches control from the current process to the process whose PID is `2A401A1B`.

DEFRAGMENT CANCEL

DEFRAGMENT CANCEL — The `CANCEL` subcommand skips the next scheduled execution of the scripts specified in the command line.

Format

DEFRAGMENT CANCEL [{ script-name } { /ALL }]

Parameters

[script-name]

The `script-name` parameter is the name of the scripts to be canceled. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

Description

The `CANCEL` subcommand causes the scheduler to skip the next scheduled execution of the script specified in the command line. Normal scheduling resumes afterward.

Restrictions

The `script-name` parameter is not allowed if the `/ALL` qualifier is used.

Qualifiers

/ALL

/NOALL (default)

The /ALL qualifier cancels the next scheduled execution of all scripts in the scheduling and parameter database.

Examples

The following is an example of the CANCEL command:

```
$ DEFRAGMENT CANCEL MY_SCRIPT, YOUR_SCRIPT
```

In this example, the next scheduled execution of MY_SCRIPT and YOUR_SCRIPT are canceled. Afterwards, both scripts resume their normal schedule.

DEFRAGMENT DISABLE

DEFRAGMENT DISABLE — The DISABLE subcommand makes the specified script ineligible for execution. Currently executing scripts are not affected. Use the ABORT subcommand to stop currently executing scripts.

Format

DEFRAGMENT DISABLE [{ script-name } { /ALL }]

Parameters

[script-name]

The script-name parameter is the name of the scripts to be canceled. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

Description

The DISABLE subcommand makes the specified script ineligible for execution. The script remains disabled until reactivated using the ENABLE subcommand.

Disabling a predecessor script automatically disables any scripts that are successors.

Disabling an already disabled script has no effect; however, an informational message is issued.

Restrictions

The script-name parameter is not allowed if the /ALL qualifier is used.

Qualifiers

/ALL

/NOALL (default)

The /ALL qualifier disables all scripts in the scheduling and parameter database.

Examples

The following example shows how to disable a script:

```
$ DEFRAGMENT DISABLE MY_SCRIPT
```

In this example, MY_SCRIPT is disabled.

DEFRAGMENT ENABLE

DEFRAGMENT ENABLE — The ENABLE subcommand reactivates scripts that have been disabled.

Format

```
DEFRAGMENT ENABLE [ { script-name } { /ALL } ]
```

Parameters

[script-name]

The script-name parameter is the name of the scripts to be canceled. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

Description

The ENABLE subcommand makes the specified script eligible for execution. The specified script remains enabled until disabled by the DISABLE subcommand.

Enabling an already enabled script has no effect; however, an informational message is issued.

The ENABLE subcommand reactivates disabled scripts including any successor/predecessor dependencies within a linked list of scripts.

Restrictions

The script-name parameter is not allowed if the /ALL qualifier is used.

Qualifiers

/ALL

/NOALL (default)

The /ALL qualifier enables all disabled scripts in the scheduling and parameter database.

Examples

The following are examples of the ENABLE command:

```
1. $ DEFRAGMENT ENABLE/ALL
```

In this example, all previously deactivated scripts are enabled. The enabled script executes according to its previously set schedule.

Enabled predecessor or successor scripts resume executing according to their link dependencies.

2. \$ DEFRAGMENT ENABLE MY_SCRIPT

In this example, MY_SCRIPT is enabled. MY_SCRIPT starts executing according to the parameters and qualifiers assigned to it.

DEFRAGMENT EXIT

DEFRAGMENT EXIT — The EXIT subcommand exits from the Disk File Optimizer utility and returns the user to the system prompt.

Format

DEFRAGMENT EXIT

Parameters

None.

Description

The EXIT subcommand exits the user from the Disk File Optimizer utility. EXIT shuts everything down in an orderly sequence.

Restrictions

None.

Qualifiers

None.

Examples

The following is an example of the EXIT command:

```
DFO> EXIT
$
```

This command exits from the Disk File Optimizer utility and returns the user to the DCL prompt.

DEFRAGMENT FILE

DEFRAGMENT FILE — The DEFRAGMENT FILE command defragments the files specified on the command line according to the DEFRAGMENT command line qualifiers. Use this command to defragment a small number of files. To defragment a large number of files, use the DEFRAGMENT VOLUME command.

Format

DEFRAGMENT FILE [{ file-spec[...] } { /AFTER | /BRIEF | /CONSOLIDATE_FREESPACE | /DORMANT | /EPILOGUE | /EXCLUDE | /FREQUENT | /FULL | /HOTFILE_DATABASE | /IGNORE | /INHERIT | /INTERVAL | /LEVEL | /LOG | /MAIL_NOTIFICATION | /NODE | /

OPCOM_NOTIFICATION | /OPTIONS | /PREDECESSOR | /PRIORITY | /PROLOGUE | /RVN | /
SCRIPT | /TIME_LIMIT | /WRITE_CHECK }}

Parameters

[file-spec[,...]]

The file-spec parameter identifies the file or files to be defragmented. You may use wildcards for the file specification.

Description

The DEFRAGMENT FILE command defragments the specified file or files according to the selected command line qualifiers. When you specify a file name without the device and/or directory, the defragmentation software operates on files in the current directory on the current device. You must explicitly specify the device and directory name for files not in the default directory where the command is issued.

This command may be permanently added to the scheduling and parameter database as a script. If no script name is specified using the /SCRIPT qualifier, the defragmentation software generates a unique script name of the form DFG\$SCRIPTX[X...], where X is a hexadecimal number. This number is chosen to keep all script names unique. Names beginning with DFG\$ are reserved for use by the defragmentation software.

Once the defragmentation command line is entered in the database, it is eligible for execution according to its scheduling parameters. If no scheduling information is provided, the script begins executing immediately and executes only once.

By default, script definitions remain in the scheduling database until their associated defragmentation process completes. However, a script is permanently added to the scheduling database if it is named as a predecessor by another script or if any of the following qualifiers are present on the DEFRAGMENTFILE command line:

- /SCRIPT={script-name}
- /INTERVAL={delta-time}
- /PREDECESSOR={script-name}

Restrictions

A script that has the /CONTINUOUS qualifier may not be named as the predecessor of another script. If you specify the /INTERVAL qualifier, the /PREDECESSOR qualifier is not allowed. If /LEVEL=5 is selected, the /LOG qualifier is required. Do not use /BRIEF and /FULL concurrently.

Qualifiers

/AFTER={time}

/AFTER={current_time}(default)

/NOAFTER

The /AFTER qualifier keeps the script from executing until after the specified time.

You can specify time as either an absolute time or as a combination of absolute and delta times. For complete information on specifying time values, see the *VSI OpenVMS DCL Dictionary*.

The /AFTER qualifier has three states. It can be specified with a valid time, it can be omitted from the command line, or it can be negated (/NOAFTER). If /AFTER is specified with a valid time, the script is scheduled to execute after that time. If the /AFTER qualifier is omitted, the defragmentation software processes the command as if the /AFTER qualifier were specified with the current time. If /NOAFTER is specified, no starting time is associated with the script being defined. The /NOAFTER qualifier allows you to define a script without scheduling it to execute. To subsequently start the script, use the START subcommand.

/BRIEF (default)**/NOBRIEF**

The /BRIEF qualifier is used only with the /LOG qualifier. The /BRIEF qualifier requests a brief log of all actions taken by the defragmentation process. This log includes beginning and ending file status information. The /BRIEF and /FULL qualifiers are mutually exclusive, and /NOBRIEF produces the same result as /FULL. The /BRIEF qualifier is the default.

/CONSOLIDATE_FREESPACE**/NOCONSOLIDATE_FREESPACE (default)**

The /CONSOLIDATE_FREESPACE qualifier tells the Disk File Optimizer to use its free space consolidation algorithm.

/DORMANT=(file-spec[,...])**/NODORMANT (default)**

The /DORMANT qualifier indicates files to be considered dormant when final file placement decisions are made. In order for the file to be placed dormant, it must be the file specified in the command line. Dormant files are placed at the high end of the volume.

If a file is indicated in both the DORMANT and FREQUENT lists, it is considered dormant. If a file is indicated in both the EXCLUDE and DORMANT lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

/EPILOGUE={file-name}**/NOEPILOGUE (default)**

The /EPILOGUE qualifier identifies a DCL command file which DFO executes upon completion of the DEFRAGMENT operation.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/EXCLUDE=(file-spec[,...])**/NOEXCLUDE (default)**

The /EXCLUDE qualifier specifies files to be excluded from processing. The defragmentation process ignores files matching the listed file specifications. Any file indicated in the EXCLUDE list and in either the FREQUENT or DORMANT lists is excluded from processing.

Note

The defragmentation process always excludes system files, open files, files excluded in an options file (/OPTIONS qualifier), and files excluded using the /IGNORE qualifier.

/FREQUENT=(file-spec[,...])

/NOFREQUENT (default)

The /FREQUENT qualifier indicates files to be considered frequently accessed when final file placement decisions are made. In order for the file to be placed frequent, it must be the file specified in the command line. Frequent files are placed toward the center of the volume.

If a file is indicated in both the FREQUENT and DORMANT lists, it is considered dormant. If a file is indicated in both the FREQUENT and EXCLUDE lists, it is excluded.

Note

The defragmentation process always excludes system files, open files, files excluded in an options file (/OPTIONS qualifier), and files excluded using the /IGNORE qualifier.

See *Section 4.3, "Scripts"* for more information.

/FULL

/NOFULL (default)

The /FULL qualifier requests a full log of all actions taken by the defragmentation process. The /FULL qualifier has no meaning if the /LOG qualifier is absent. The /FULL and /BRIEF qualifiers are mutually exclusive, and /NOFULL provides the same functionality as /BRIEF.

/HOTFILE_DATABASE

/NOHOTFILE_DATABASE (default)

The /HOTFILE_DATABASE qualifier causes the defragmenter to query the appropriate hotfile database and append the list of hotfiles to the /FREQUENT file list.

/IGNORE=(keyword[,...])

/IGNORE=(PLACED)(default)

/NOIGNORE

The /IGNORE qualifier requests that the defragmentation process ignore certain types of files as specified by the following keywords:

- INDEXED—Ignores all RMS indexed files
- NOINDEXED—May defragment RMS indexed files
- PLACED—Ignores all files with explicit placement flags set in their headers
- NOPLACED—May defragment files with explicit placement flags

Files indicated by the /IGNORE qualifier are added to the EXCLUDE list.

The default value (/IGNORE=PLACED) indicates that files with explicit placement control should be ignored, and that RMS indexed files should be defragmented and placed in appropriate locations on the volume.

/INHERIT={script-name}
/NOINHERIT (default)

The purpose of the /INHERIT qualifier is to make script creation easier. By inheriting most of the command data from a previously entered script and modifying only a few items (such as volume name), a completely new script can be created. The /INHERIT qualifier creates a database entry for a new script in the following way:

- Copies all fields from the specified script to the new script.
- Overwrites any qualifier or parameter values in the new script with those specified on the command line.
- Names the new script as appropriate. The user may specify a script name using the /SCRIPT={script-name} qualifier. Otherwise, the defragmentation software creates a unique script name. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

/INTERVAL={delta-time}
/NOINTERVAL (default)

The /INTERVAL qualifier executes a script at regularly scheduled intervals. The /INTERVAL qualifier specifies the minimum time between two consecutive executions of a script.

Specify time as a delta time. Refer to the *VSI OpenVMS DCL Dictionary* for more information on specifying times. If you do not specify a time with the /INTERVAL qualifier, a default value of 24 hours is used. This value causes the associated script to run every 24 hours.

If you specify the /AFTER qualifier with the /INTERVAL qualifier, the first defragmentation operation occurs at or after the time specified by the /AFTER qualifier; all subsequent operations execute at intervals set according to the /INTERVAL qualifier.

/LEVEL={n}
/LEVEL=1 (default)

The defragmentation process selects files for defragmentation (from among those not excluded) based on the /LEVEL qualifier. Once a file is selected as a candidate, the defragmentation process attempts to fully defragment that file (make it 100 percent contiguous). Specify the level number as an integer ranging from one to five:

- /LEVEL=1

Any file that is not 100 percent contiguous, or that can be selected using levels two and three, is a candidate for defragmentation.

- /LEVEL=2 (default)

Any file whose extents cannot be completely mapped by the mapping window, or that can be selected using level three, is a candidate for defragmentation. The default window value is seven and can be set for a given volume using:

```
$ SET VOLUME {device-spec}/WINDOWS=[n]
```

- /LEVEL=3

Any file with retrieval pointers in more than one file header is a candidate for defragmentation.

- **/LEVEL=4**

The defragmentation process for level four processing performs only free space consolidation. This level is used with the **/CONSOLIDATE_FREESPACE** qualifier.

- **/LEVEL=5**

This level requests a volume analysis only; no defragmentation occurs. The statistical results are available in the log file, which is required when **/LEVEL=5** is selected (see the **/LOG** qualifier). If level 5 is selected, the **/LEVEL=2** candidate file spec

/LOG={file-spec}

/NOLOG (default)

The **/LOG** qualifier causes the defragmentation process to create a file containing a list of all its operations. The information in the log file varies with the presence of the **/FULL** or **/BRIEF** qualifiers.

The **/FULL** qualifier lists all files analyzed by the defragmentation process and reports their status during each phase of defragmentation.

The **/BRIEF** qualifier lists only the statistics of each phase, such as number of files analyzed and number of files defragmented.

/MAIL_NOTIFICATION

/NOMAIL_NOTIFICATION (default)

The **/MAIL_NOTIFICATION** qualifier requests that brief status and termination messages regarding defragmentation processes be mailed to the distribution list provided in the file DFG \$MAIL_ADDRESS.

/NODE=(node-name[,...])

/NONODE (default)

The **/NODE** qualifier limits the scope of the **DEFRAGMENT** command to the node or set of nodes specified by **/NODE=(node-name)**. By default, defragmentation processes may execute on any LMF licensed node in the system.

/OPCOM_NOTIFICATION (default)

/NOOPCOM_NOTIFICATION

The **/OPCOM_NOTIFICATION** qualifier requests that brief status and termination messages regarding defragmentation processes be sent to OPCOM.

/OPTIONS={file-spec}

/NOOPTIONS (default)

The **/OPTIONS** qualifier provides an alternate method for associating files with the **/DORMANT**, **/EXCLUDE**, or **/FREQUENT** qualifiers. If the list of files for any of these qualifiers is long, you can create a separate file for this information. Each record in the options file consists of a file name followed by one of the following keywords:

- **/EXCLUDE**—File to be excluded

- /FREQUENT—File to be considered frequently accessed
- /DORMANT—File to be considered dormant

The following example shows the contents of an options file called MYOPTFILE.OPT. This file specifies that all files on the target device called FILE1.EXE are frequent files, all files in the [ARCHIVE] directory are to be excluded, and all files in the [OLDSTUFF] directory and its subdirectories are dormant.

```
FILE1.EXE/FREQUENT
[ARCHIVE]*.*;*/EXCLUDE
[OLDSTUFF...]*.*;*/DORMANT
```

Files indicated by the /EXCLUDE qualifier and by either the /DORMANT or /FREQUENT qualifier are excluded. Files indicated by the /DORMANT qualifier and by the /FREQUENT qualifier are considered dormant. This rule is applied equally across all files specified in the /OPTIONS file, as well as those associated with these qualifiers on the command line.

/PREDECESSOR={script-name}
/NOPREDECESSOR (default)

The /PREDECESSOR qualifier links scripts. These scripts execute in sequence; one successfully completes before the next one begins.

The predecessor script name must exist in the scheduling database or an error status message is returned. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

/PRIORITY=(DEFAULT=n, MINIMUM=n)
/NOPRIORITY (default)

The /PRIORITY qualifier specifies priorities for the defragment process. The minimum priority setting is used when the defragmentation process exceeds the value specified by either /CPU_LIMIT or /IO_LIMIT.

/PROLOGUE={file-name}
/NOPROLOGUE (default)

The /PROLOGUE qualifier identifies a DCL command file which DFO executes just before the Defragment operation starts.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/RVN=n
/RVN=0 (default)

When you have a bound volume set, use this qualifier to identify a disk from the volume set or all disks in the volume set. The default of zero tells the Disk File Optimizer to defragment all disks in the bound volume set. Specify a relative volume number (RVN) to defragment a specific disk that belongs to a bound volume set.

/SCRIPT={script-name}
/NOSCRIPT (default)

The **/SCRIPT** qualifier assigns a user-supplied name to a script. If a duplicate *script-name* is selected, defragmentation software issues a duplicate script name error message and does not enter the new script into the database—no defragmentation takes place.

A script name may be up to 64 characters in length and may contain any of the following characters:

- Any alphanumeric character
- Underscore (_)
- Hyphen (-)
- Dollar sign (\$)

Observe the following rules when naming a script:

- You may use any combination of uppercase and lowercase letters.
- DO not begin script names using the dollar sign. However, you can include the dollar sign within the script name. (The dollar sign is reserved for special use by VSI.)
- Do not end a script name with a hyphen, since the hyphen is the DCL continuation character.

/TIME_LIMIT={delta-time}
/NOTIME_LIMIT (default)

The **/TIME_LIMIT** qualifier specifies the maximum elapsed time allowed for a given defragmentation operation. If the time limit is exceeded, the defragmentation process is aborted.

Specify time as a delta time. Refer to the *VSI OpenVMS User's Manual* for more information on specifying times.

For linked scripts, the time limit of the first script applies to the whole set. Time limits specified with successor scripts apply to these scripts only.

If the time limit is set unreasonably low, the defragmentation process may not accomplish useful defragmentation.

/WRITE_CHECK
/NOWRITE_CHECK (default)

The **/WRITE_CHECK** qualifier requests that a read-after-write check be performed on all defragmented or placed files. This qualifier significantly extends the execution time of a defragmentation process.

Examples

The following examples show different ways to defragment a file or group of files and how to use the common qualifiers:

1. \$ DEFRAGMENT FILE *.C; *

This command defragments all files with a .C extension in the user's current default directory. To determine the device and directory in which the files are located, issue the SHOW DEFAULT command.

2. DFO> DEFRAGMENT FILE *.C; *

This command defragments all files with a .C extension on the user's current default device. The only difference between this command and the previous example is that this command is entered from within the Disk File Optimizer utility instead of from the DCL prompt.

3. \$ DEFRAGMENT FILE SYS\$SYSDEVICE:[000000...]*.C

This command is the same as the previous one, except that the RMS default string is now explicitly identified as SYS\$SYSDEVICE:[000000...]*.C;.*.

4. \$ DEFRAGMENT FILE SYS\$LOGIN:LOGIN.COM;2/EXCLUDE=*.OLD; *

This command defragments the user's LOGIN.COM file and excludes all files with a .OLD extension.

5. \$ DEFRAGMENT FILE/INHERIT=OTHER_FILE_SCRIPT.NOAFTER

This command creates a file script identical to the OTHER_FILE_SCRIPT and allows the Disk File Optimizer to name it (DFG\$XXX).

6. \$ DEFRAGMENT FILE SYS\$SYSTEM:[000000...]*.RDB/INHERIT=X/SCRIPT=Y

This command creates a script called Y. Other script characteristics, such as logfile, are inherited from script X.

DEFRAGMENT HELP

DEFRAGMENT HELP — The HELP subcommand invokes VMS Help for the Disk File Optimizer utility.

Format

DEFRAGMENT HELP

Parameters

None.

Description

The HELP subcommand invokes VMS HELP for the Disk File Optimizer. HELP only provides information on the version of the Disk File Optimizer installed on your system(Defrag or Defrag PLUS). You can also access HELP for the Disk File Optimizer directly from DCL by entering one of these commands:

\$ HELP DEFRAGMENT

or

\$ DEFRAGMENT HELP

Restrictions

None.

Qualifiers

None.

Examples

The following examples show the various ways you can access HELP for the Disk File Optimizer utility:

1. `$ DEFRAGMENT HELP`

This command invokes HELP for the Disk File Optimizer utility directly from DCL.

2. `$ HELP DEFRAGMENT`

This command invokes HELP for the Disk File Optimizer utility directly from DCL.

3. `DFO> HELP`

This command invokes HELP from within the Disk File Optimizer utility.

DEFRAGMENT MODIFY

DEFRAGMENT MODIFY — The MODIFY subcommand allows you to change the qualifiers associated with a particular script.

Format

DEFRAGMENT MODIFY [{ script-name } { /AFTER | /BRIEF | /CONSOLIDATE_FREESPACE | /CONTINUOUS | /DORMANT | /EPILOGUE | /EXCLUDE | /FILES | /FREQUENT | /FULL | /HOTFILE_DATABASE | /IGNORE | /INTERVAL | /LEVEL | /LOG | /MAIL_NOTIFICATION | /NODE | /OPCOM_NOTIFICATION | /OPTIONS | /PREDECESSOR | /PRIORITY | /PROLOGUE | /RVN | /SCRIPT | /TIME_LIMIT | /VOLUME | /WRITE_CHECK }]

Parameters

[script-name]

The script-name parameter identifies the script to modify. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

Description

The MODIFY subcommand modifies specified portions of an existing script. This is easier than using the INHERIT subcommand to do similar things.

Restrictions

A script that has the /CONTINUOUS qualifier may not be named as the predecessor of another script. If you specify the /INTERVAL qualifier, the /PREDECESSOR qualifier is not allowed. If /LEVEL=5 is selected, the /LOG qualifier is required. Do not use /BRIEF and /FULL concurrently. Do not use /CONTINUOUS and /INTERVAL for a script.

Qualifiers

/AFTER={time}
/AFTER={current_time}(default)
/NOAFTER

The /AFTER qualifier keeps the script from executing until after the specified time.

You can specify time as either an absolute time or as a combination of absolute and delta times. For complete information on specifying time values, see the *VSI OpenVMS User's Manual*.

The /AFTER qualifier has three states. It can be specified with a valid time, it can be omitted from the command line, or it can be negated (/NOAFTER). If /AFTER is specified with a valid time, the script is scheduled to execute after that time. If the /AFTER qualifier is omitted, the defragmentation software processes the command as if the /AFTER qualifier were specified with the current time. If /NOAFTER is specified, no starting time is associated with the script being defined. The /NOAFTER qualifier allows you to define a script without scheduling it to execute. To subsequently start the script, use the START subcommand.

/BRIEF (default)
/NOBRIEF

The /BRIEF qualifier may only be used when the /LOG qualifier is also present. The /BRIEF qualifier requests a brief log of all actions taken by the defragmentation process. This log includes beginning and ending volume status information. The /BRIEF and /FULL qualifiers are mutually exclusive, and /NOBRIEF produces the same result as /FULL. The /BRIEF qualifier is the default.

/CONSOLIDATE_FREESPACE
/NOCONSOLIDATE_FREESPACE (default)

The /CONSOLIDATE_FREESPACE qualifier tells the Disk File Optimizer to use its free space consolidation algorithm.

/CONTINUOUS
/NOCONTINUOUS (default)

The /CONTINUOUS qualifier causes the script to execute repeatedly. After each successful defragmentation operation, the script becomes eligible for execution again. As long as any PROCESS_LIMITs (see the SET subcommand) are not violated, the script executes repeatedly. If all PROCESS_LIMITs have been reached, the script is queued until an outstanding defragmentation process completes, then the script is allowed to execute.

By default, the defragmentation process associated with a script may execute on any appropriately licensed node in a cluster. Thus, successive defragmentation processes do not necessarily execute on the same node.

/DORMANT=(file-spec[,...])
/NODORMANT (default)

The /DORMANT qualifier indicates files to be considered dormant when final file placement decisions are made. Dormant files are placed at the high end of the volume.

If a file is indicated in both the DORMANT and FREQUENT lists, it is considered dormant. If a file is indicated in both the EXCLUDE and DORMANT lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

/EPILOGUE={file-name}
/NOEPILOGUE (default)

The /EPILOGUE qualifier identifies a DCL command file which DFO executes upon completion of the DEFRAGMENT operation.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/EXCLUDE=(file-spec[,...])
/NOEXCLUDE (default)

The /EXCLUDE qualifier specifies files to be excluded from processing. The defragmentation process ignores files matching the listed file specifications. Any file indicated in the EXCLUDE list and in either the FREQUENT or DORMANT lists is excluded from processing.

Note

The defragmentation process always excludes system files, open files, files excluded in an options file (/OPTIONS qualifier), and files excluded using the /IGNORE qualifier.

/FILES=(file-spec[,...])

The /FILES qualifier modifies the list of files to be defragmented.

/FREQUENT=(file-spec[,...])
/NOFREQUENT (default)

The /FREQUENT qualifier indicates files to be considered frequently accessed when final file placement decisions are made. Frequent files are placed toward the center of the volume.

If a file is indicated in both the FREQUENT and DORMANT lists, it is considered dormant. If a file is indicated in both the FREQUENT and EXCLUDE lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

See *Section 4.3, "Scripts"* for more information.

/FULL
/NOFULL (default)

The /FULL qualifier requests a full log of all actions taken by the defragmentation process. The /FULL qualifier has no meaning if the /LOG qualifier is absent. The /FULL and /BRIEF qualifiers are mutually exclusive, and /NOFULL provides the same functionality as /BRIEF.

/HOTFILE_DATABASE**/NOHOTFILE_DATABASE (default)**

The /HOTFILE_DATABASE qualifier causes the defragmenter to query the appropriate hotfile database and append the list of hotfiles to the defragmenter's /FREQUENT file list.

/IGNORE=(keyword[,...])**/IGNORE=(PLACED)(default)****/NOIGNORE**

The /IGNORE qualifier requests that the defragmentation process ignore certain types of files as specified by the following keywords:

- INDEXED—Ignores all RMS indexed files
- NOINDEXED—May defragment RMS indexed files
- PLACED—Ignores all files with explicit placement flags set in their headers
- NOPLACED—May defragment files with explicit placement flags

Files indicated by the /IGNORE qualifier are added to the EXCLUDE list.

The default value (/IGNORE=PLACED) indicates that files with explicit placement control should be ignored, and that RMS indexed files should be defragmented and placed in appropriate locations on the volume.

/INTERVAL={delta-time}**/NOINTERVAL (default)**

Causes a script to execute at regularly scheduled intervals. The /INTERVAL qualifier specifies the minimum time between two consecutive executions of a script.

Specify time as a delta time. Refer to the *VSI OpenVMS User's Manual* for more information on specifying times. If you do not specify a time with the /INTERVAL qualifier, a default value of 1 day is used. This value causes the associated script to run every 24 hours.

If you specify the /AFTER qualifier with the /INTERVAL qualifier, the first defragmentation operation occurs at or after the time specified by the /AFTER qualifier; all subsequent operations execute at intervals set according to the /INTERVAL qualifier.

/LEVEL={n}**/LEVEL=2 (default)**

The defragmentation process selects files for defragmentation (from among those not excluded) based on the /LEVEL qualifier. Once a file is selected as a candidate, the defragmentation process attempts to fully defragment that file (make it 100 percent contiguous). Specify the level number as an integer ranging from one to five:

- /LEVEL=1

Any file that is not 100 percent contiguous, or that can be selected using levels two and three, is a candidate for defragmentation.

- /LEVEL=2 (default)

Any file whose extents cannot be completely mapped by the mapping window, or that can be selected using level three, is a candidate for defragmentation. The default window value is seven and can be set for a given volume using:

```
$ SET VOLUME {device-spec}/WINDOWS=[n]
```

- **/LEVEL=3**

Any file with retrieval pointers in more than one file header is a candidate for defragmentation.

- **/LEVEL=4**

The defragmentation process for level four processing performs only free space consolidation. This level is used with the **/CONSOLIDATE_FREESPACE** qualifier.

- **/LEVEL=5**

This level requests a volume analysis only; no defragmentation occurs. The statistical results are available in the log file, which is required when **/LEVEL=5** is selected (see the **/LOG** qualifier). If level 5 is selected, the **/LEVEL=2** candidate file spec

/LOG={file-spec}

/NOLOG (default)

The **/LOG** qualifier causes the defragmentation process to create a file containing a list of all its operations. The information in the log file varies with the presence of the **/FULL** or **/BRIEF** qualifiers. The **/FULL** qualifier lists all files analyzed by the defragmentation process and reports their disposition during each phase of defragmentation. The **/BRIEF** qualifier lists only the statistics of each phase, such as number of files analyzed, and number of files defragmented.

/MAIL_NOTIFICATION

/NOMAIL_NOTIFICATION (default)

The **/MAIL_NOTIFICATION** qualifier requests that brief status and termination messages regarding defragmentation processes be mailed to the mail distribution list provided in the file **DFG \$MAIL_ADDRESS**.

/NODE=(node-name[,...])

/NONODE (default)

The **/NODE** qualifier limits the scope of the **DEFRAGMENT** command to the node or set of nodes specified by **/NODE=(node-name)**. By default, defragmentation processes may execute on any LMF licensed node in the system.

/OPCOM_NOTIFICATION (default)

/NOOPCOM_NOTIFICATION

The **/OPCOM_NOTIFICATION** qualifier requests that brief status and termination messages regarding defragmentation processes be sent to **OPCOM**.

/OPTIONS={file-spec}

/NOOPTIONS (default)

The **/OPTIONS** qualifier provides an alternate method for associating files with the **/DORMANT**, **/EXCLUDE**, or **/FREQUENT** qualifiers. If the list of files for any of these qualifiers is long, you

can create a separate file for this information. Each record in the options file consists of a file name followed by one of the following keywords:

- **/EXCLUDE**—File to be excluded
- **/FREQUENT**—File to be considered frequently accessed
- **/DORMANT**—File to be considered dormant

The following example shows the contents of an options file called MYOPTFILE.OPT. This file specifies that all files on the target device called FILE1.EXE are frequent files, all files in the [ARCHIVE] directory are to be excluded, and all files in the [OLDSTUFF] directory and its subdirectories are dormant.

```
FILE1.EXE/FREQUENT [ARCHIVE] *.*;*/EXCLUDE [OLDSTUFF...] *.*;*/DORMANT
```

Files indicated by the **/EXCLUDE** qualifier and by either the **/DORMANT** or **/FREQUENT** qualifier are excluded. Files indicated by the **/DORMANT** qualifier and by the **/FREQUENT** qualifier are considered dormant. This rule is applied equally across all files specified in the **/OPTIONS** file, as well as those associated with these qualifiers on the command line.

/PREDECESSOR={script-name}

/NOPREDECESSOR (default)

The **/PREDECESSOR** qualifier identifies the name of the script that precedes the execution of this script in a linked set of scripts. These scripts execute in sequence; one successfully completes before the next one begins.

The predecessor script name must exist in the scheduling database or an error status message is returned. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

/PRIORITY=(DEFAULT=n, MINIMUM=n)

/NOPRIORITY (default)

The **/PRIORITY** qualifier specifies priorities for the defragment process. The minimum priority setting is used when the defragmentation process exceeds the value specified by either **/CPU_LIMIT** or **/IO_LIMIT**.

/PROLOGUE={file-name}

/NOPROLOGUE (default)

The **/PROLOGUE** qualifier identifies a DCL command file which DFO executes just before the Defragment operation starts.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/RVN=n

/RVN=0 (default)

When you have a bound volume set, use this qualifier to identify a disk from the volume set or all disks in the volume set. The default of zero tells the Disk File Optimizer to defragment all disks in

the bound volume set. Specify a relative volume number (RVN) to defragment a specific disk that belongs to a bound volume set.

/SCRIPT={script-name}

/NOSCRIPT (default)

The /SCRIPT qualifier assigns a new name to a script. If a duplicate *script-name* is selected, defragmentation software issues a duplicate script name error message and does not enter the new script into the database—no defragmentation takes place.

A script name may be up to 64 characters in length and may contain any of the following characters:

- Any alphanumeric character
- Underscore (_)
- Hyphen (-)
- Dollar sign (\$)

Observe the following rules when naming a script:

- Use any combination of uppercase and lowercase letters.
- Do not begin script names using the dollar sign. However, you can include the dollar sign within the script name. (The dollar sign is reserved for special use by VSI.)
- Do not end a script name with a hyphen, since the hyphen is the DCL continuation character.

The *VSI OpenVMS User's Manual* contains more information on naming conventions.

/TIME_LIMIT={delta-time}

/NOTIME_LIMIT (default)

The /TIME_LIMIT qualifier specifies the maximum elapsed time allowed for a given defragmentation operation. If the time limit is exceeded, the defragmentation process is aborted.

Specify time as a delta time. Refer to the *VSI OpenVMS User's Manual* for more information on specifying times.

For linked scripts, the time limit of the first script applies to the whole set. Time limits specified with successor scripts apply to these scripts only.

If the time limit is set unreasonably low, the defragmentation process may not accomplish useful defragmentation.

/VOLUME={volume-name}

/NOVOLUME (default)

Identifies the volume where the files reside.

/WRITE_CHECK

/NOWRITE_CHECK (default)

The /WRITE_CHECK qualifier requests that a read-after-write check be performed on all defragmented or placed files. This qualifier significantly extends the execution time of a defragmentation process.

Examples

The following examples show how to modify an existing scripts using the common qualifiers:

1. `$ DEFRAGMENT MODIFY C/SCRIPT=A`

This command changes the name of script C to A.

2. `$ DEFRAGMENT MODIFY */MAIL_NOTIFICATION`

This command turns on mail notification for all scripts in the database.

3. `$ DEFRAGMENT MODIFY myscript/LOG=new_log/NOPREDECESSOR`

This command causes subsequent executions of myscript to generate a log file called NEW_LOG.LOG, located in the user's default directory. The NOPREDECESSOR qualifier disassociates myscript from its predecessor. However, any script for which myscript is a predecessor remains dependent upon the execution of myscript.

4. `$ DEFRAGMENT MODIFY *C/SCRIPT=*D`

This command changes the name of each script that ends in C to end in D. For instance, DAILYC becomes DAILYCD.

DEFRAGMENT MONITOR

DEFRAGMENT MONITOR — The MONITOR subcommand displays the progress of an ongoing defragmentation processes on a terminal.

Format

DEFRAGMENT MONITOR [{ script-name|volume-name } { /CONTINUOUS | /VOLUME }]

Parameters

[script-name|volume-name]

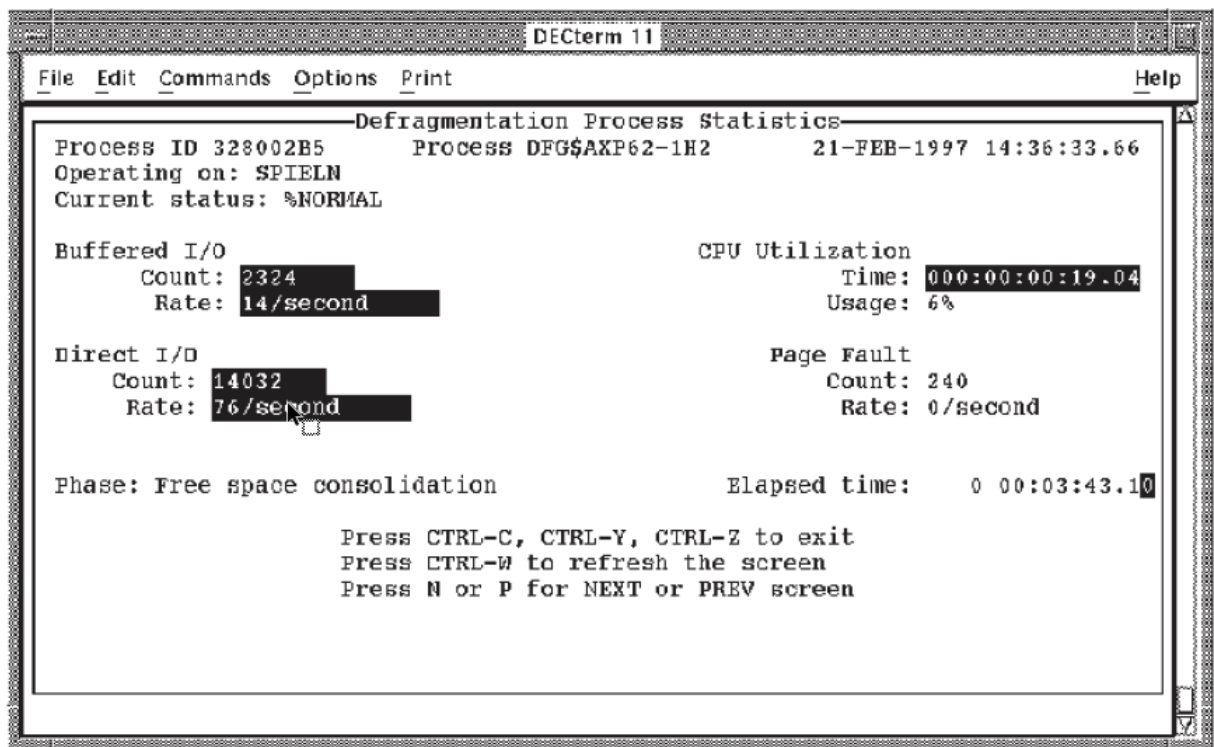
The parameter for the MONITOR subcommand is either a script name or a volume name for Defrag PLUS. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

If a volume name is supplied, then you must use the /VOLUME qualifier to tell Defrag PLUS the parameter is a volume name, rather than a script name.

Description

The MONITOR subcommand provides current process statistics for the defragmentation process and information about the volume being defragmented. The monitor displays this information on three window displays—the process statistics display, the process activities display, and the free space display.

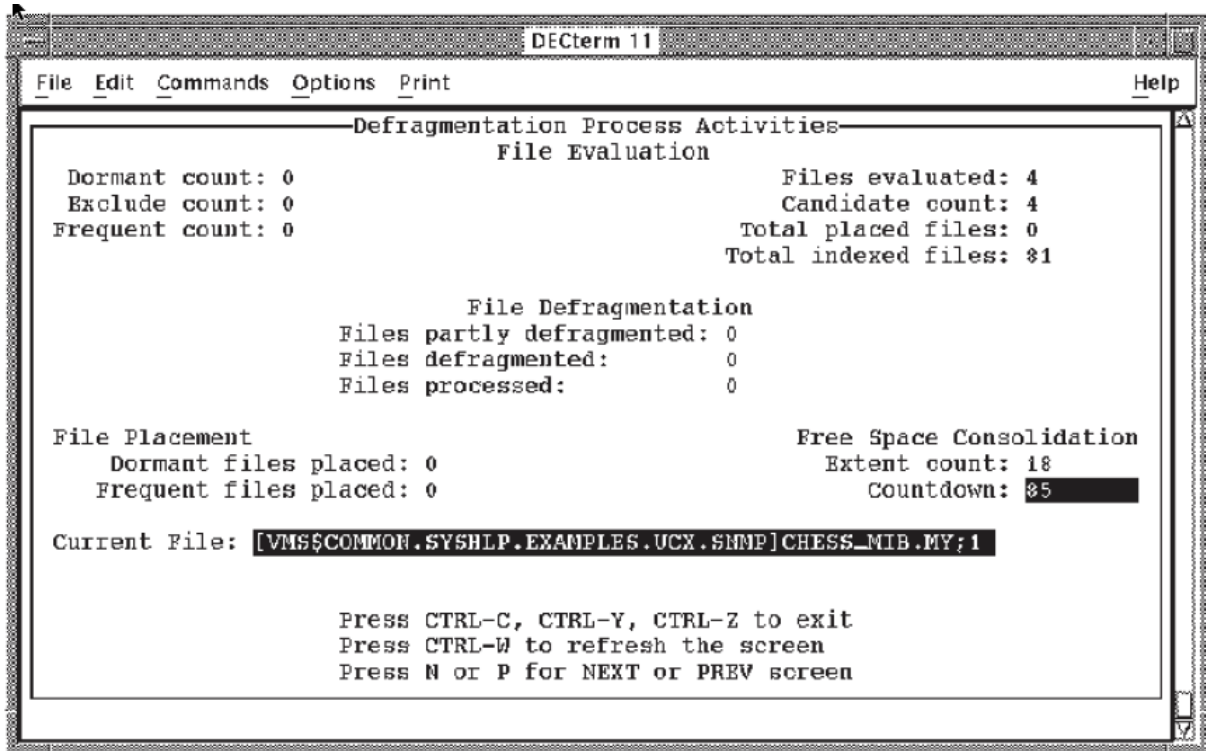
The process statistics display shows the following defragmentation process information (*Figure 4.1, "Process Statistics Display"*):

Figure 4.1. Process Statistics Display

- The defragmentation process ID
- The defragmentation process name
- The current date and time
- The name of the script being monitored
- The node on which the defragmentation process is operating
- The status of the defragmentation process*
- The buffered I/O count and rate of the defragmentation process*
- The CPU utilization (time and usage) of the defragmentation process*
- The direct I/O count and rate of the defragmentation process*
- The page fault count and rate of the defragmentation process*
- The current defragmentation process phase*
- The elapsed time since the start of the defragmentation process

The process information marked with an asterisk (*) in the preceding list appears highlighted on the process display whenever it changes.

To access the process activities display, enter N or press the Next Screen key. The process activities display shows the following process-related information (*Figure 4.2, "Process Activities Display"*):

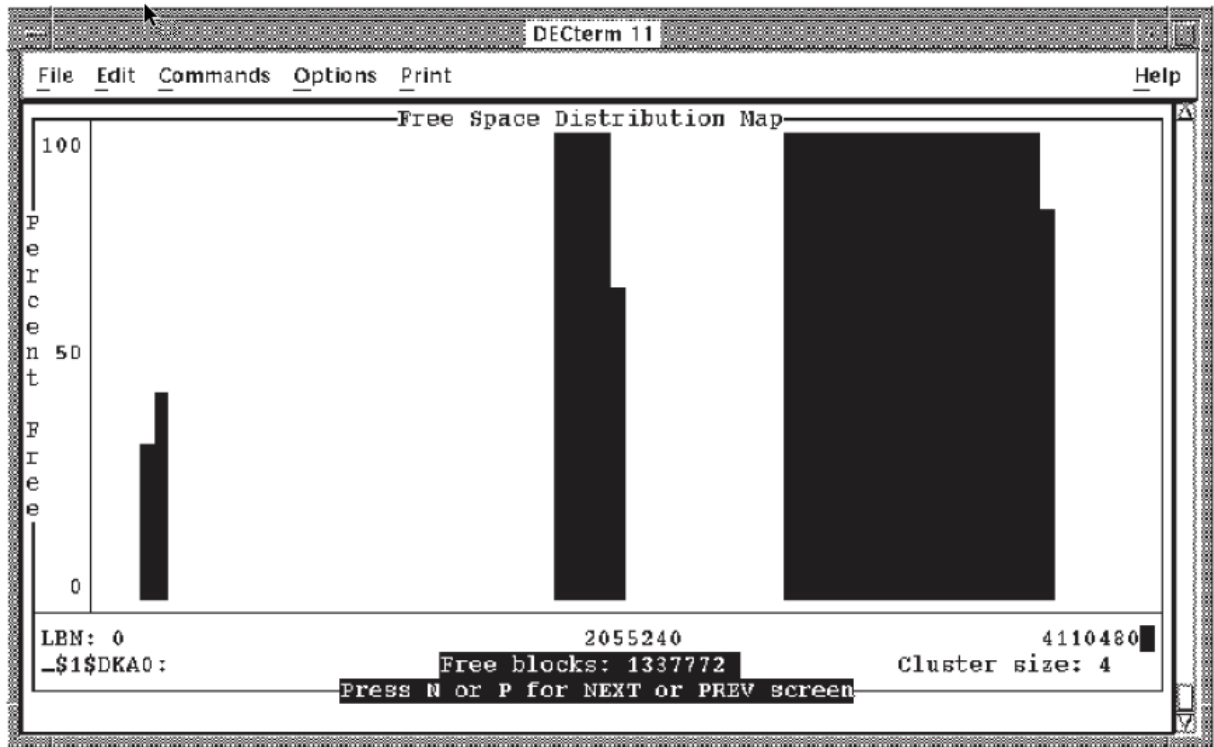
Figure 4.2. Process Activities Display

- Name of current defragmentation phase
- Number of files evaluated
- Number of candidate files for defragmentation
- Number of dormant files
- Total number of placed files
- Number of excluded files
- Total number of indexed files
- Number of frequent files
- For the file defragmentation phase:
 - Number of files partly defragmented
 - Number of files defragmented
 - Number of files processed
- For the file placement phase:
 - Number of dormant files placed
 - Number of frequent files placed
- For the free space consolidation phase:

- Number of extents to be consolidated into free space
- Number of extents left to be consolidated into free space
- Current file being evaluated

To access the free space display, enter N or press the NextScreen key. The free space display shows the following device-related information (*Figure 4.3, "Free Space Display"*):

Figure 4.3. Free Space Display



- An approximate free space distribution map*
- The name of the device being defragmented*
- The number of free blocks on the device*
- The cluster size of the device
- The maximum LBN on the device

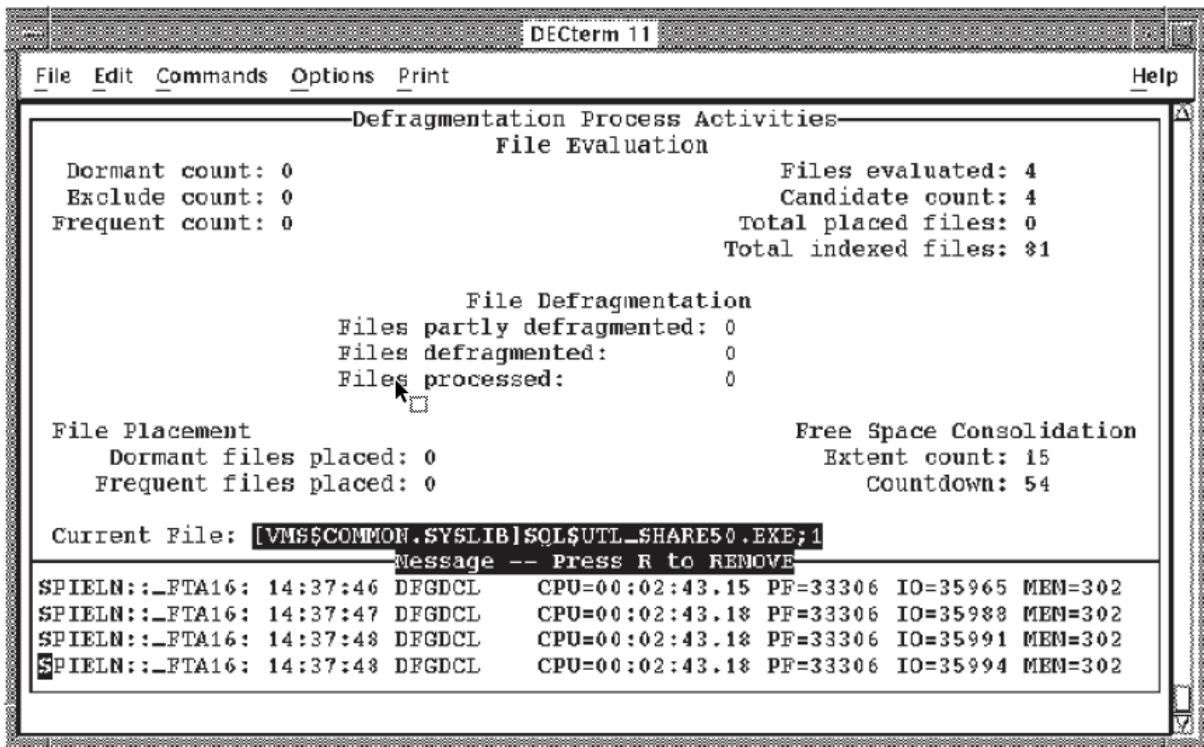
The device information marked with an asterisk (*) in the preceding list appears highlighted on the device display whenever it changes.

The free space map represents how free space is organized on the volume being defragmented. The Y axis indicates the percentage of free space in a given LBN range; the X axis indicates the LBN range.

A fourth display of information—the message display—appears on the bottom of your screen whenever a broadcast message is received by the process executing the monitor. The message display holds the last four broadcast messages (such as mail notification) received by the process. To erase this display, enter Ctrl/W; to remove this display entirely, enter R or press the Remove key. To scroll through the broadcast

messages, enter Ctrl/T. Figure 4.4, "Process Statistics Display with Message Display" shows the process statistics display with the additional message display.

Figure 4.4. Process Statistics Display with Message Display



The following list summarizes the keystrokes that manipulate the monitor displays:

- Ctrl/C, Ctrl/Y, or Ctrl/Z exit the monitor.
- Ctrl/W refreshes the screen and erases the message display.
- N or Next Screen switches to the next display.
- P or Previous Screen switches to previous display.
- R or Remove removes the message display.
- Ctrl/T scrolls through the broadcast messages.

Restrictions

If a volume name is entered as a parameter, the /VOLUME qualifier must be used.

Qualifiers

/CONTINUOUS (default)

/NOCONTINUOUS

The /CONTINUOUS qualifier invokes a continuous monitor display. Entering Ctrl/C, Ctrl/Y, or Ctrl/Z terminates this display.

Using /NOCONTINUOUS provides a snapshot display and then returns to the DCL prompt.

/VOLUME (default)
/NOVOLUME

The /VOLUME qualifier indicates that a volume name instead of a script name is present on the command line. The /VOLUME qualifier must be used when a volume name is provided as a parameter to the MONITOR subcommand.

Examples

The following are examples of invoking the monitor:

1. `$ DEFRAGMENT MONITOR MY_SCRIPT/NOVOLUME`

In this example, the monitor function displays ongoing run-time statistics for the defragmentation process associated with MY_SCRIPT. It also displays an approximate free-space map of the target volume. The display continues until interrupted with a Ctrl/C, Ctrl/Y, or Ctrl/Z because the /CONTINUOUS qualifier is present by default.

2. `$ DEFRAGMENT MONITOR DISK$ONE/NOCONTINUOUS`

In this example, the defragmentation process executing against DISK\$ONE is monitored. Since the /NOCONTINUOUS qualifier was specified, the monitor function samples the defragmentation process statistics and the free space on DISK\$ONE one time before exiting to the DCL prompt.

DEFRAGMENT OFFLINE_VOLUME

DEFRAGMENT OFFLINE_VOLUME — The DEFRAGMENT OFFLINE_VOLUME command defragments the INDEXF.SYS file on the offline volume given as a parameter to the command line according to the defragment command line qualifiers.

Format

DEFRAGMENT OFFLINE_VOLUME [{ volume-name } { /AFTER | /EPILOGUE | /INTERVAL | /LOG, brief only | /MAIL_NOTIFICATION | /NODE | /OPCOM_NOTIFICATION | /PROLOGUE | /SCRIPT }]

Parameters

[volume-name]

The volume-name is the name of the volume containing the INDEXF.SYS file to be defragmented.

Description

The OFFLINE_VOLUME subcommand defragments INDEXF.SYS on the specified volume according to the selected command line qualifiers. By definition INDEXF.SYS has a minimum of four extents. The extents consist of the following:

1. Bootstrap block and home block
2. Backup home block
3. Backup index file header

4. Index file bitmap and file headers

The minimum number of extents INDEXF.SYS can have after defragmentation is five.

Before defragmenting the INDEXF.SYS file, do the following:

1. Defragment and consolidate free space on the disk volume. Consolidating free space on the disk volume gives DFO a place to create a new defragmented INDEXF.SYS file.
2. Check the free space summary to ensure there is enough free space on the disk volume to create a defragmented INDEXF.SYS. Use the following command to check the free space summary:

```
$ DEFRAGMENT SHOW disk_volume:/VOL
```

An example of the free space summary report is shown below:

```
Freespace Summary:
  Total free space:      211072 blocks
  Percentage free:       10 (rounded)
  Total free extents:    23
  Maximum free extent:   53316 blocks, LBN: 671852
  Minimum free extent:   4 blocks, LBN: 730360
  Average free extent:   9177 blocks
  Median free extent:    584 blocks
```

If you defragment the INDEXF.SYS file without adequate free extent, the defragmentation of INDEXF.SYS could fail.

3. Analyze the disk volume to make sure it has no problems:

```
$ ANALYZE/DISK disk_name
```

If the disk needs to be repaired, use the following command to repair it:

```
$ ANALYZE/DISK/REPAIR disk_name
```

After defragmenting the INDEXF.SYS file, perform another ANALYZE/DISK/REPAIR.

This command may be permanently added to the scheduling and parameter database as a script. If no script name is specified using the /SCRIPT qualifier, the defragmentation software generates a unique script name.

Once the defragmentation command line is entered in the database, it is eligible for execution according to its scheduling parameters. If no scheduling information is provided, the script begins executing immediately and executes only once.

By default, script definitions remain in the scheduling database until their associated defragmentation process completes. However a script is permanently added to the scheduling database if any of the following qualifiers are present on the DEFRAGMENT OFFLINE_VOLUME command line:

```
/SCRIPT = {script name }
/INTERVAL = {delta time }
```

Restrictions

- This command is only valid on a disk device that is not mounted. If the device is mounted, an error occurs and the process will abort. This command cannot work on the system disk because it is always mounted.

- For a bound volume set, this utility only works on the "ROOT" volume.
- For disk volumes in a shadow set (single member or otherwise), the following restrictions apply:
 - Do not specify the virtual unit name to defragment (for example: DSAAnnnn:).
 - Specify only one physical device name (for example: \$1\$DUAnnnn:).
 - Use the following procedure to defragment INDEXF.SYS in a shadow set:
 1. Dismount both members of the shadow set.
 2. Mount one of the physical disk volumes with /OVERRIDE=SHADOW and then dismount it before doing the defragmentation. If you do not do this, you will receive the following error:

```
%SYSTEM-F-WRITLCK, write lock error
```
 3. Defragment the INDEXF.SYS on the disk volume.
 4. Mount the disk volume into the shadow set.
 5. Add the disk volume that was not defragmented into the shadow set. Mounting this disk volume causes a full copy operation.
 - If you experience a power failure during the defragmentation of INDEXF.SYS, you may need to do the following:
 - Repair alternate index file header and index file bitmap by doing a ANALYZE/DISK/REPAIR.
 - Delete disk:[000000]DFG\$DEFRAG_INDEXF.DAT.

Qualifiers

/AFTER={time}

/AFTER={current_time}(default)

/NOAFTER

The /AFTER qualifier keeps the script from executing until after the specified time.

You can specify time as either an absolute time or as a combination of absolute and delta times. For complete information on specifying time values, see the *VSI OpenVMS User's Manual*.

The /AFTER qualifier has three states. It can be specified with a valid time, it can be omitted from the command line, or it can be negated (/NOAFTER). If /AFTER is specified with a valid time, the script is scheduled to execute after that time. If the /AFTER qualifier is omitted, the defragmentation software processes the command as if the /AFTER qualifier were specified with the current time. If /NOAFTER is specified, no starting time is associated with the script being defined. The /NOAFTER qualifier allows you to define a script without scheduling it to execute. To subsequently start the script, use the START subcommand.

/EPILOGUE={file-name}

/NOEPILOGUE (default)

The /EPILOGUE qualifier identifies a DCL command file which DFO executes upon completion of the DEFRAGMENT operation.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/INTERVAL={delta-time}

/NOINTERVAL (default)

The /INTERVAL qualifier executes a script at regularly scheduled intervals. The /INTERVAL qualifier specifies the minimum time between two consecutive executions of a script.

Specify time as a delta time. Refer to the *VSI OpenVMS User's Manual* for more information on specifying times. If you do not specify a time with the /INTERVAL qualifier, a default value of 1 day is used. This value causes the associated script to run every 24 hours.

If you specify the /AFTER qualifier with the /INTERVAL qualifier, the first defragmentation operation occurs at or after the time specified by the /AFTER qualifier; all subsequent operations execute at intervals set according to the /INTERVAL qualifier.

/LOG={file-spec}

/NOLOG (default)

The /LOG qualifier causes the defragmentation process to create a file containing a list of all its operations.

/MAIL_NOTIFICATION

/NOMAIL_NOTIFICATION (default)

The /MAIL_NOTIFICATION qualifier requests that brief status and termination messages regarding defragmentation processes be mailed to the mail distribution list provided in the file DFG \$MAIL_ADDRESS.

/NODE=(node-name[,...])

/NONODE (default)

The /NODE qualifier limits the scope of the DEFRAGMENT command to the node or set of nodes specified by /NODE=(node-name). By default, defragmentation processes may execute on any LMF licensed node in the system.

/OPCOM_NOTIFICATION (default)

/NOOPCOM_NOTIFICATION

The /OPCOM_NOTIFICATION qualifier requests that brief status and termination messages regarding defragmentation processes be sent to OPCOM.

/PROLOGUE={file-name}

/NOPROLOGUE (default)

The /PROLOGUE qualifier identifies a DCL command file which DFO executes just before the Defragment operation starts.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/SCRIPT={script-name}
/NOSCRIPT (default)

The /SCRIPT qualifier assigns a user-supplied name to a script. If a duplicate *script-name* is selected, defragmentation software issues a duplicate script name error message and does not enter the new script into the database—no defragmentation takes place.

A script name may be up to 64 characters in length and may contain any of the following characters:

- Any alphanumeric character
- Underscore (_)
- Hyphen (-)
- Dollar sign (\$)

Observe the following rules when naming a script:

- You may use any combination of uppercase and lowercase letters.
- You cannot begin script names using the dollar sign, but you can include the dollar sign within the script name. (The dollar sign is reserved for special use by VSI.)
- You should not end a script name with a hyphen, since the hyphen is the DCL continuation character.

Examples

The following examples illustrate how to use the OFFLINE_VOLUME subcommand and its qualifiers.

```
$ DEFRAGMENT OFFLINE_VOLUME $1$DKB100:-  
    /SCRIPT=DEFRAG_OFFLINE_USER1-  
    /NOAFTER  
$ DISMOUNT/CLUSTER $1$DKB100:  
$ MOUNT/OVER=ID $1$DKB100:  
$ ANALYZE/DISK_STRUCTURE $1$DKB100:  
$ DISMOUNT $1$DKB100:  
$ DEFRAGMENT START DEFRAG_OFFLINE_USER1  
$ MOUNT/OVER=ID $1$DKB100:  
$ ANALYZE/DISK_STRUCTURE $1$DKB100:  
$ DISMOUNT $1$DKB100:  
$ MOUNT/CLUSTER $1$DKB100: USER1
```

In this example, the following is performed:

1. Create script to defragment INDEXF.SYS on volume
2. Dismount volume from the cluster
3. Mount volume privately to analyze disk structure
4. Check volume for readability and validity of Files-11 structure information
5. Dismount volume to defragment INDEXF.SYS
6. Start OFFLINE_VOLUME script

7. Mount volume privately to analyze disk structure
8. Check volume for readability and validity of Files-11 structure information
9. Dismount volume
10. Mount volume into the cluster

DEFRAGMENT REMOVE

DEFRAGMENT REMOVE — The REMOVE subcommand deletes the specified script or node from the defragmentation scheduling and parameter database.

Format

DEFRAGMENT REMOVE [{ script-name[,...] | node-name[,...] } { /AFTER | /ALL | /NODE }]

Parameters

[script-name]

The script-name parameter specifies the script to remove from the defragmentation database. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

This parameter is not allowed if the /ALL qualifier is used.

[node-name]

The node-name parameter specifies the node to remove from the defragmentation database.

Description

The REMOVE subcommand deletes the specified script or node from the defragmentation database. Scripts executing when the command is issued are allowed to complete before they are removed.

When a script that has a predecessor or a successor is removed, all forward and backward links are updated to reflect the removal. For example, assume that A is the predecessor of B, B is the predecessor of C, and B is removed. After B is removed, A becomes the predecessor of C.

Restrictions

If the /ALL qualifier is used, then the script-name parameter cannot be used. If the node is a script, it cannot be removed. If the node has a scheduler or defragmentation process running, it cannot be removed.

Qualifiers

/AFTER=time

/NOAFTER (default)

The /AFTER qualifier does not delete the script until after the specified time. Specify *time* as a VMS absolute time or combination absolute and delta time. Refer to the *VSI OpenVMS User's Manual* for more information on how to specify time.

/ALL**/NOALL (default)**

The /ALL qualifier indicates that all scripts are to be removed from the scheduling and parameter database.

If /ALL is used, the script-name parameter is not allowed.

/NODE

The /NODE qualifier removes the specified node name from the scheduling and parameter database.

Examples

The following example shows how to remove a script after a specified time:

1. `$ DEFRAGMENT REMOVE MY_SCRIPT/AFTER=16:30`

In this example, MY_SCRIPT is removed from the database. It is removed after 16:30 on the day this command is entered.

2. `$ DEFRAGMENT REMOVE MYNODE/NODE`

In this example, MYNODE is removed from the database.

DEFRAGMENT SET

DEFRAGMENT SET — The SET subcommand defines or changes the value of the defragmentation process limits.

Format

DEFRAGMENT SET [{ (parameter=value[,...]) } { /NODE }]

Parameters

[keywords:]

- CPU_LIMIT specifies the percent of CPU usage for any one defragmentation process. The CPU percent is the percent of a single CPU.
- IO_LIMIT specifies the maximum direct I/O rate (per second) of any one defragmentation process.
- PROCESS_LIMIT specifies the maximum number of defragmentation processes per node.

Description

The CPU and I/O limits levels, when exceeded, cause the scheduler to decrement the process priority level of the defragmentation process once every minute until:

- The limits are no longer exceeded.
- The defragmentation process priority is at the SYSGEN parameter, DEFPRI, minus 1.

Defragmentation processes have a base priority equal to the SYSGEN parameter, DEFPRI, or the current priority of the scheduler, whichever is greater.

If there are more than the specified number of defragmentation processes running on any of the given nodes when the PROCESS_LIMIT is set, the excess processes are allowed to continue. However, new processes are not started until the number of executing defragmentation processes drops below the PROCESS_LIMIT value.

Enter values for CPU_LIMIT as integers ranging from 0 to 100 (inclusive). Enter values for IO_LIMIT and PROCESS_LIMIT as integers ≥ 0 .

A value of zero for any of these parameters imposes no limit.

Restrictions

None.

Qualifiers

/NODE=(node-name[,...])

/NONODE (default)

The /NODE qualifier limits the scope of the SET subcommand. By default, a given parameter is set identically on all nodes in the scheduling database. To limit parameters to fewer nodes, specify the node names with the /NODE qualifier.

Examples

The following examples show how to set limits using the SET subcommand and its qualifiers:

1. `$ DEFRAGMENT SET PROCESS_LIMIT=3, IO_LIMIT=14 /NODE=NODE_1`

In this example, the maximum number of defragmentation processes that can occur on NODE_1 is 3. Further, a restriction of 14 direct I/Os per second is placed on each of these 3 defragmentation processes.

2. `$ DEFRAGMENT SET CPU_LIMIT=50`

In this example, the maximum CPU resources allocated to any defragmentation process is set to 50 percent on all nodes in the system. If a defragmentation process exceeds this limit, its priority is decremented each minute (approximately) until it reaches the SYSGEN parameter, DEFPRI, minus 1.

DEFRAGMENT SHOW

DEFRAGMENT SHOW — The SHOW subcommand displays information about the fragmentation state of the given volume and information from the scheduling database.

Format

DEFRAGMENT SHOW [{ script-name | volume-name } { /ALL | /COMMAND_LINE | /DEVICE | /FILE_LIST | /FREE_LIST | /HISTOGRAM | /LOCATION | /NAMES | /NODE | /OUTPUT | /PARAMETERS | /RUNNING | /STATISTICS | /VOLUME }]

Parameters

[script-name |volume-name]

The parameter for the SHOW subcommand is either a script name or a volume name. Names can be omitted if the /ALL qualifier is used. If a script name is supplied or if the /PARAMETERS qualifier is specified, the SHOW command queries the scheduling database and displays this information. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

If a volume name is supplied (/VOLUME required), the SHOW command displays information about the fragmentation state of the specified volume. A list of names can be provided, but the list cannot contain both script and volume names.

Description

Information displayed as a result of showing the volume fragmentation state includes:

- A list of files with a user-selected minimum level of fragmentation
- A histogram of the file fragmentation and free space fragmentation state of the volume
- A list of all free space extents on the volume
- Summary statistics on both file fragmentation and free space fragmentation
- The location of user selected files

All information in the volume fragmentation analysis is optional. By default, only the summary statistics are provided.

The SHOW subcommand also can display data contained in the scheduling database. Depending on additional qualifiers, the output consists of the SET parameters and script information. Further, the output can be directed to a file and can be displayed in two formats.

Restrictions

- If showing a script or volume, do not use the /ALL qualifier
- If showing a volume, then you must use the /VOLUME qualifier

Qualifiers

/ALL

/NOALL (default)

The /ALL qualifier is used to show information from the scheduling database. It displays information about all scripts and parameters in the scheduling database. If /ALL is used, the script-name parameter is not allowed.

/COMMAND_LINE

/NOCOMMAND_LINE (default)

The /COMMAND_LINE qualifier is used to show information from the scheduling database. It formats the output of the SHOW subcommand as a valid DEFRAGMENT command line. Thus,

all the data in the scheduling database can be extracted into a file (/ALL/COMMAND_LINE/OUTPUT=file.ext).

/DEVICE=(device_spec[,device_spec])...

The /DEVICE qualifier selects only the named scripts on the specified devices.

/FILE_LIST=(min_extents)

/FILE_LIST=(VOL_MAP_WINDOW + 1) (default)

The /FILE_LIST qualifier is used to show the fragmentation state of a volume. It lists all files on the volume that have *min_extents* or more extents (fragments). By default, the number of file window mapping pointers set for the volume (INITIALIZE {volume}/WINDOW=n) plus one is used. This value lists all files that require at least one window turn to completely map.

/FREE_LIST

/NOFREE_LIST (default)

The /FREE_LIST qualifier is used to show the fragmentation state of the volume. It lists the location and size of all of the free space extents on the volume.

/HISTOGRAM

/NOHISTOGRAM (default)

The /HISTOGRAM qualifier is used to show the fragmentation state of a volume. It generates two histograms. The first plots the number of files against the number of extents per file and provides information about how many files are fragmented and how badly they are fragmented. The second histogram plots the number of free space extents (holes) against the size of these free space extents. It provides information about how many free space extents are on the volume and their sizes.

/LOCATION=(file-spec[,...])

/NOLOCATION

The /LOCATION qualifier is used to show the fragmentation state of the volume. It prints the extents (fragments) associated with all specified files. If no files are specified, the /LOCATION qualifier appends location information to all files listed with the /FILE_LIST qualifier. The supplied file-spec may contain wildcards in place of the directory, file name, extension, or version fields.

/NAMES

The /NAMES qualifier lists only the first line of the usual show output. The first line of the script includes the name of the script, whether the script is enabled or disabled, and the target device for the script.

/NODE=(node-name[,...])

/NONODE (default)

The /NODE qualifier selects only those scripts and node names that are specified on the command line.

When used with the /PARAMETERS qualifier, the /NODE qualifier displays CPU_LIMIT, IO_LIMIT, and PROCESS_LIMIT information for specified nodes only. If the /NODE qualifier is not used, the /PARAMETERS qualifier displays this information for all nodes in the scheduling database.

/OUTPUT={file-spec}

/OUTPUT=SYS\$OUTPUT (default)

The /OUTPUT qualifier may be used to show the volume fragmentation state or to show information from the scheduling database. It directs the output of the SHOW subcommand to a file. By default, output is directed to SYS\$OUTPUT.

/PARAMETERS

/NOPARAMETERS (default)

The /PARAMETERS qualifier is used to show information from the scheduling database. It displays the CPU_LIMIT, IO_LIMIT, and PROCESS_LIMIT settings for all nodes in the scheduling database (unless specific nodes are designated with the /NODE qualifier). By default, these settings are not shown.

/RUNNING

The /RUNNING qualifier shows scripts that are currently running.

/STATISTICS (default)

/NOSTATISTICS

The /STATISTICS qualifier is used to show the volume fragmentation state. It provides a summary of the file and free space fragmentation for the volume. Refer to *Appendix C, "DEFRAGMENT SHOW Command Report"* for more information on this report.

/VOLUME

/NOVOLUME (default)

The /VOLUME qualifier must be used to show the fragmentation state of a volume. It indicates that any parameter supplied on the command line is a volume name instead of a script name. The /VOLUME qualifier is incompatible with any of the qualifiers used to show database information except the /OUTPUT qualifier.

Examples

The following are examples of the SHOW subcommand and its qualifiers:

1. `$ DEFRAGMENT SHOW/PARAMETERS`

In this example, the current values of CPU_LIMIT, IO_LIMIT and PROCESS_LIMIT for all nodes in the scheduling database are shown.

2. `$ DEFRAGMENT SHOW MY_SCRIPT/OUTPUT=OUTPUT.TXT`

In this example, information associated with MY_SCRIPT is displayed. No parameter information is displayed. Output is directed to a file called OUTPUT.TXT.

Note

If output is directed to a filename without extension, the file is given a default extension of .lis.

3. `$ DEFRAGMENT SHOW MY_SCRIPT/COMMAND_LINE/PARAMETERS`

In this example, information associated with MY_SCRIPT is displayed. The current values of CPU_LIMIT, IO_LIMIT and PROCESS_LIMIT for all nodes in the scheduling database are also

shown. The format for the information is a valid DEFRAGMENT command line. Output is directed to the user's SYS\$OUTPUT device. If output is directed to a filename without extension, the file is given a default extension of .lis.

4. \$ DEFRAGMENT SHOW MY_VOLUME/OUTPUT=OUTPUT.TXT

In this example, information associated with MY_VOLUME is displayed. Output is directed to a file called OUTPUT.TXT.

5. \$ DEFRAGMENT SHOW DISK\$ONE/VOLUME/HISTOGRAM

In this example, the volume fragmentation report includes the summary statistics (/STATISTICS is default) and histograms for both file fragmentation and free space fragmentation. The report is sent to the SYS\$OUTPUT device.

DEFRAGMENT SPAWN

DEFRAGMENT SPAWN — The Disk File Optimizer SPAWN subcommand creates a subprocess and executes a DCL command in the subprocess, or transfers terminal control to the subprocess if no command is supplied.

Format

DEFRAGMENT SPAWN [{ command }]

Parameters

[command]

This is a DCL command string of less than 132 characters that is to be executed in the context of the created subprocess. After the command is executed, the subprocess terminates and control returns to the parent process. A command is optional. If no command is supplied, control passes to DCL.

Description

The Disk File Optimizer SPAWN command creates a subprocess of your current process.

Use the DCL LOGOUT command to terminate the subprocess and return to the parent process. You can also use the DCL ATTACH command to transfer control of the terminal to another process in the subprocess tree, including the parent process.

If the Disk File Optimizer SPAWN command is invoked with a DCL command specified as the *command* parameter, that command is executed in the subprocess, and control returns immediately to the parent process.

If the *command* parameter is not specified, then control is passed to the created DCL subprocess until it is deleted or until an ATTACH command is issued to transfer control to another process.

Restrictions

None.

Qualifiers

None.

Examples

The following are examples of the SPAWN command:

1.

```
$ DEFRAGMENT
DFO> SPAWN
$
```

The SPAWN command creates a subprocess and transfers terminal control to the created subprocess.

2.

```
DFO> SPAWN SHOW DEFAULT    DISK$1:[JONES]
DFO>
```

The SPAWN command creates a subprocess and executes the DCL command SHOW DEFAULT in the created subprocess. Control then returns to the Disk File Optimizer parent process.

DEFRAGMENT START

DEFRAGMENT START — The START subcommand starts the specified script.

Format

DEFRAGMENT START [{ script-name[,...] } { /AFTER }]

Parameters

[script-name]

The script-name parameter is the name of the script(s) to be started. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

Description

The START subcommand starts any script not currently executing. You may use it to start scripts halted due to unusual system events, such as system failures. If a defragmentation process fails, successor scripts are not automatically started. Use the START subcommand to start these successor scripts.

Restrictions

None.

Qualifiers

/AFTER=time

/NOAFTER (default)

The /AFTER qualifier keeps the script from starting until after the specified time.

You can specify time as either an absolute time or as a combination of absolute and delta times. For complete information on specifying time values, see the *VSI OpenVMS User's Manual*.

Examples

The following example shows how to start a script:

```
$ DEFRAGMENT START MY_SCRIPT/AFTER=16:30
```

In this example, MY_SCRIPT is started. The /AFTER qualifier stipulates that MY_SCRIPT should start after 16:30 of the day this command is submitted. If the script is periodic, the period is unchanged.

DEFRAGMENT VOLUME

DEFRAGMENT VOLUME — The DEFRAGMENT VOLUME subcommand defragments the volume named on the command line according to the defragment command line qualifiers.

Format

```
DEFRAGMENT VOLUME [ { volume-name } { /AFTER | /BRIEF | /CONSOLIDATE_FREESPACE  
| /CONTINUOUS | /DORMANT | /EPILOGUE | /EXCLUDE | /FREQUENT | /FULL  
| /HOTFILE_DATABASE | /IGNORE | /INHERIT | /INTERVAL | /LEVEL | /LOG | /  
MAIL_NOTIFICATION | /NODE | /OPCOM_NOTIFICATION | /OPTIONS | /PREDECESSOR | /  
PRIORITY | /PROLOGUE | /RVN | /SCRIPT | /TIME_LIMIT | /WRITE_CHECK } ]
```

Parameters

[volume-name]

The volume-name parameter is the name of the volume to be defragmented.

Description

The VOLUME subcommand defragments the specified volume according to the selected command line qualifiers. This command may be permanently added to the scheduling and parameter database as a script. If no script name is specified using the /SCRIPT qualifier, the defragmentation software generates a unique script name of the form DFG\$SCRIPTX[X...], where X is a hexadecimal number. This number is chosen to keep all script names unique. Names beginning with DFG\$ are reserved for use by the defragmentation software.

Once the defragmentation command line is entered in the database, it is eligible for execution according to its scheduling parameters. If no scheduling information is provided, the script begins executing immediately and executes only once.

By default, script definitions remain in the scheduling database until their associated defragmentation process completes. However, a script is permanently added to the scheduling database if it is named as a predecessor by another script or if any of the following qualifiers are present on the DEFRAGMENT VOLUME command line:

- /SCRIPT={script-name}
- /INTERVAL={delta-time}
- /PREDECESSOR={script-name}

Restrictions

A script that has the /CONTINUOUS qualifier may not be named as the predecessor of another script. If you specify the /INTERVAL qualifier, the /PREDECESSOR qualifier is not allowed. If /LEVEL=5 is selected, the /LOG qualifier is required. Do not use /BRIEF and /FULL concurrently. Do not use /CONTINUOUS and /INTERVAL for a script.

Qualifiers

/AFTER={time}
/AFTER={current_time}(default)
/NOAFTER

The /AFTER qualifier keeps the script from executing until after the specified time.

You can specify time as either an absolute time or as a combination of absolute and delta times. For complete information on specifying time values, see the *VSI OpenVMS User's Manual*.

The /AFTER qualifier has three states. It can be specified with a valid time, it can be omitted from the command line, or it can be negated (/NOAFTER). If /AFTER is specified with a valid time, the script is scheduled to execute after that time. If the /AFTER qualifier is omitted, the defragmentation software processes the command as if the /AFTER qualifier were specified with the current time. If /NOAFTER is specified, no starting time is associated with the script being defined. The /NOAFTER qualifier allows you to define a script without scheduling it to execute. To subsequently start the script, use the START subcommand.

/BRIEF (default)
/NOBRIEF

The /BRIEF qualifier has meaning only when the /LOG qualifier is also present. The /BRIEF qualifier requests a brief log of all actions taken by the defragmentation process. This log includes beginning and ending volume status information. The /BRIEF and /FULL qualifiers are mutually exclusive, and /NOBRIEF produces the same result as /FULL. The /BRIEF qualifier is the default.

/CONSOLIDATE_FREESPACE (default)
/NOCONSOLIDATE_FREESPACE

The /CONSOLIDATE_FREESPACE qualifier tells the Disk File Optimizer to use its free space consolidation algorithm.

/CONTINUOUS
/NOCONTINUOUS (default)

The /CONTINUOUS qualifier causes the script to execute repeatedly. After each successful defragmentation operation, the script becomes eligible for execution again. As long as any PROCESS_LIMITs (see the SET subcommand) are not violated, the script executes repeatedly. If all PROCESS_LIMITs have been reached, the script is queued until an outstanding defragmentation process completes, then the script is allowed to execute.

By default, the defragmentation process associated with a script may execute on any appropriately licensed node in a cluster. Thus, successive defragmentation processes do not necessarily execute on the same node.

/DORMANT=(file-spec[,...])
/NODORMANT (default)

The /DORMANT qualifier indicates files to be considered dormant when final file placement decisions are made. Dormant files are placed at the high end of the volume.

If a file is indicated in both the DORMANT and FREQUENT lists, it is considered dormant. If a file is indicated in both the EXCLUDE and DORMANT lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

/EPILOGUE={file-name}
/NOEPILOGUE (default)

The /EPILOGUE qualifier identifies a DCL command file which DFO executes upon completion of the DEFRAGMENT operation.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/EXCLUDE=(file-spec[,...])
/NOEXCLUDE (default)

The /EXCLUDE qualifier specifies files to be excluded from processing. The defragmentation process ignores files matching the listed file specifications. Any file indicated in the EXCLUDE list and in either the FREQUENT or DORMANT lists is excluded from processing.

Note

The defragmentation process always excludes system files, open files, files excluded in an options file (/OPTIONS qualifier), and files excluded using the /IGNORE qualifier.

/FREQUENT=(file-spec[,...])
/NOFREQUENT (default)

The /FREQUENT qualifier indicates files to be considered frequently accessed when final file placement decisions are made. Frequent files are placed to optimize access speed.

If a file is indicated in both the FREQUENT and DORMANT lists, it is considered dormant. If a file is indicated in both the FREQUENT and EXCLUDE lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

See *Section 4.3, "Scripts"* for more information.

/FULL
/NOFULL (default)

The /FULL qualifier requests a full log of all actions taken by the defragmentation process. The /FULL qualifier has no meaning if the /LOG qualifier is absent. The /FULL and /BRIEF qualifiers are mutually exclusive, and /NOFULL provides the same functionality as /BRIEF.

/HOTFILE_DATABASE**/NOHOTFILE_DATABASE (default)**

The /HOTFILE_DATABASE qualifier causes the defragmenter to query the appropriate hotfile database and append the list of hotfiles to the defragmenter's /FREQUENT file list.

/IGNORE=(keyword[,...])**/IGNORE=(PLACED)(default)****/NOIGNORE**

The /IGNORE qualifier requests that the defragmentation process ignore certain types of files as specified by the following keywords:

- INDEXED—Ignores all RMS indexed files
- NOINDEXED—May defragment RMS indexed files
- PLACED—Ignores all files with explicit placement flags set in their headers
- NOPLACED—May defragment files with explicit placement flags

Files indicated by the /IGNORE qualifier are added to the EXCLUDE list.

The default value (/IGNORE=PLACED) indicates that files with explicit placement control should be ignored, and that RMS indexed files should be defragmented and placed in appropriate locations on the volume.

/INHERIT={script-name}**/NOINHERIT (default)**

The purpose of the /INHERIT qualifier is to make script creation easier. By inheriting most of the command data from a previously entered script and modifying only a few items (such as volume name), a completely new script can be created. The /INHERIT qualifier creates a database entry for a new script in the following way:

- Copies all fields from the specified script to the new script.
- Overwrites any qualifier or parameter values in the new script with those specified on the command line.
- Names the new script as appropriate. The user may specify a script name using the /SCRIPT={script-name} qualifier. Otherwise, the defragmentation software creates a unique script name. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

/INTERVAL={delta-time}**/NOINTERVAL (default)**

The /NOINTERVAL qualifier executes a script at regularly scheduled intervals. The /INTERVAL qualifier specifies the minimum time between consecutive executions of a script.

Specify time as a delta time. Refer to the *VSI OpenVMS User's Manual* for more information on specifying times. If you do not specify a time with the /INTERVAL qualifier, the default value of 24 hours is used.

If you specify the `/AFTER` qualifier with the `/INTERVAL` qualifier, the first defragmentation operation occurs at or after the time specified by the `/AFTER` qualifier; all subsequent operations execute at intervals set with the `/INTERVAL` qualifier.

`/LEVEL={n}`

`/LEVEL=2 (default)`

The defragmentation process selects files for defragmentation (from among those not excluded) based on the `/LEVEL` qualifier. Once a file is selected as a candidate, the defragmentation process attempts to fully defragment that file (make it 100 percent contiguous). Specify the level number as an integer ranging from one to five:

- `/LEVEL=1`

Any file that is not 100 percent contiguous, or that can be selected using levels two and three, is a candidate for defragmentation.

- `/LEVEL=2 (default)`

Any file whose extents cannot be completely mapped by the mapping window, or that can be selected using level three, is a candidate for defragmentation. The default window value is seven and can be set for a given volume using:

```
$ SET VOLUME {device-spec}/WINDOWS=[n]
```

- `/LEVEL=3`

Any file with retrieval pointers in more than one file header is a candidate for defragmentation.

- `/LEVEL=4`

The defragmentation process for level four processing performs only free space consolidation. This level is used with the `/CONSOLIDATE_FREESPACE` qualifier.

- `/LEVEL=5`

This level requests a volume analysis only; no defragmentation occurs. The statistical results are available in the log file, which is required when `/LEVEL=5` is selected (see the `/LOG` qualifier). If level 5 is selected, the `/LEVEL=2` candidate file spec

`/LOG={file-spec}`

`/NOLOG (default)`

The `/LOG` qualifier causes the defragmentation process to create a file containing a list of all its operations. The information in the log file varies with the presence of the `/FULL` or `/BRIEF` qualifiers. The `/FULL` qualifier lists all files analyzed by the defragmentation process and reports their disposition during each phase of defragmentation. The `/BRIEF` qualifier lists only the statistics of each phase, such as number of files analyzed, and number of files defragmented.

`/MAIL_NOTIFICATION`

`/NOMAIL_NOTIFICATION (default)`

The `/MAIL_NOTIFICATION` qualifier requests that brief status and termination messages regarding defragmentation processes be mailed to the mail distribution list provided in the file `DFG $MAIL_ADDRESS`.

/NODE=(node-name[,...])

/NONODE (default)

The /NODE qualifier limits the scope of the DEFRAGMENT command to the node or set of nodes specified by /NODE=(node-name). By default, defragmentation processes may execute on any LMF licensed node in the system.

/OPCOM_NOTIFICATION (default)

/NOOPCOM_NOTIFICATION

The /OPCOM_NOTIFICATION qualifier requests that brief status and termination messages regarding defragmentation processes be sent to OPCOM.

/OPTIONS={file-spec}

/NOOPTIONS (default)

The /OPTIONS qualifier provides an alternate method for associating files with the /DORMANT, /EXCLUDE, or /FREQUENT qualifiers. If the list of files for any of these qualifiers is long, you can create a separate file for this information. Each record in the options file consists of a file name followed by one of the following keywords:

- /EXCLUDE—File to be excluded
- /FREQUENT—File to be considered frequently accessed
- /DORMANT—File to be considered dormant

The following example shows the contents of an options file called MYOPTFILE.OPT. This file specifies that all files on the target device called FILE1.EXE are frequent files, all files in the [ARCHIVE] directory are to be excluded, and all files in the [OLDSTUFF] directory and its subdirectories are dormant.

```
FILE1.EXE/FREQUENT
[ARCHIVE]*.*;*/EXCLUDE
[OLDSTUFF...]*.*;*/DORMANT
```

Files indicated by the /EXCLUDE qualifier and by either the /DORMANT or /FREQUENT qualifier are excluded. Files indicated by the /DORMANT qualifier and by the /FREQUENT qualifier are considered dormant. This rule is applied equally across all files specified in the /OPTIONS file and on the command line.

/PREDECESSOR={script-name}

/NOPREDECESSOR (default)

The /PREDECESSOR qualifier links scripts. These scripts execute in sequence; one successfully completes before the next one begins.

The predecessor script name must exist in the scheduling database or an error status message is returned. You can use wildcards in the script-name specification. Use an asterisk (*) to take action on all scripts on the VAXcluster system; use a percent sign (%) to match a single character in the script-name specification.

/PRIORITY=(DEFAULT=n, MINIMUM=n)

/NOPRIORITY (default)

The /PRIORITY qualifier specifies priorities for the defragment process. The minimum priority setting is used when the defragmentation process exceeds the value specified by either /CPU_LIMIT or /IO_LIMIT.

/PROLOGUE={file-name}
/NOPROLOGUE (default)

The /PROLOGUE qualifier identifies a DCL command file which DFO executes just before the Defragment operation starts.

Mounting or dismounting the disks cannot be done through the Command file specified.

Refer to *Appendix B, "Disk File Optimizer Logicals and Files"* for logicals that are available to these command procedures.

/RVN=n
/RVN=0 (default)

When you have a bound volume set, use this qualifier to identify a disk from the volume set or all disks in the volume set. The default of zero tells the Disk File Optimizer to defragment all disks in the bound volume set. Specify a relative volume number (RVN) to defragment a specific disk that belongs to a bound volume set.

/SCRIPT={script-name}
/NOSCRIPT (default)

The /SCRIPT qualifier assigns a user-supplied name to a script. If a duplicate *script-name* is selected, defragmentation software issues an error message and does not enter the new script into the database—no defragmentation takes place.

A script name may be up to 64 characters in length and may contain any of the following characters:

- Any alphanumeric character
- Underscore (_)
- Hyphen (-)
- Dollar sign (\$)

Observe the following rules when naming a script:

- Use any combination of uppercase and lowercase letters.
- Do not begin script names using the dollar sign, however you can include the dollar sign within the script name. (The dollar sign is reserved for special use by VSI.)
- Do not end a script name with a hyphen, since the hyphen is the DCL continuation character.

If you do not supply a script name using the /SCRIPT qualifier, the defragmentation software generates a unique script name. This internally generated name has the format DFG \$SCRIPTX[X...], where X is a hexadecimal number that keeps the script names unique. Names beginning with DFG\$ are reserved for use by the defragmentation software.

/TIME_LIMIT={delta-time}
/NOTIME_LIMIT (default)

The /TIME_LIMIT qualifier specifies the maximum elapsed time allowed for a given defragmentation operation. If the time limit is exceeded, the defragmentation process is aborted.

Specify time as a delta time. Refer to the *VSI OpenVMS User's Manual* for more information on specifying times.

For linked scripts, the time limit of the first script applies to the whole set. Time limits specified with successor scripts apply to these scripts only.

If the time limit is set unreasonably low, the defragmentation process may not accomplish useful defragmentation.

/WRITE_CHECK

/NOWRITE_CHECK (default)

The /WRITE_CHECK qualifier requests that a read-after-write check be performed on all defragmented or placed files. This qualifier significantly extends the execution time of a defragmentation process.

Examples

The following are examples of the DEFRAGMENT VOLUME command:

1. `$ DEFRAGMENT VOLUME DISK$USER_1`

This command defragments DISK\$USER_1 using default qualifiers. The defragmentation process executes only once. In a VAXcluster system, the defragment operation begins on a node selected at random from among those nodes in the scheduling database having fewer executing processes than the limit imposed by the SETPROCESS_LIMIT={value} subcommand. If all nodes are busy with a maximum number of defragment processes, the script does not execute until one of the outstanding defragmentation operations concludes.

After the defragmentation process stops, the script is removed from the scheduling database.

2. `$ DEFRAGMENT VOLUME DISK$ONE/SCRIPT=MY_SCRIPT/IGNORE=(INDEXED,PLACED) -
/INTERVAL=2-00:00/FREQUENT=*.EXE/LOG=MYLOG.LOG/FULL/
DORMANT=*.ARCHIVE-
/NODE=(N1, N2)`

This command does the following:

- Defragments DISK\$ONE.
- Names the script MY_SCRIPT.
- Instructs the defragmentation process to ignore all RMS indexed files and files with explicit placement control information in their headers. (These files are added to the EXCLUDE list.)
- Instructs the scheduler to execute this script every two days.
- Instructs the defragmentation process that all .EXE files on the entire volume should be considered frequently accessed when final file placement decisions are made.
- Requests a log file, MYLOG.LOG, which is placed in the directory from which this command was entered.
- Requests a /FULL log file.
- Instructs the defragmentation process that all .ARCHIVE files on the entire volume should be considered dormant (rarely accessed) when final file placement decisions are made.

- Limits the nodes available for the defragmentation process to nodes N1 and N2.

3. `$ DEFRAGMENT VOLUME DISK$TWO/INHERIT=MY_SCRIPT/SCRIPT=FIRST_SCRIPT`

This command inherits the script defined in the previous example. Thus, it creates a new script that is the same as MY_SCRIPT, except for the following differences:

- The script name is FIRST_SCRIPT instead of MY_SCRIPT.
- The target volume is DISK\$TWO instead of DISK\$ONE.

In this case, MYLOG.LOG is also inherited. Thus, one version of the log file contains data from MY_SCRIPT, and the other version contains data from FIRST_SCRIPT. To prevent log file inheritance, specify the /LOG qualifier.

4. `$ DEFRAGMENT VOLUME DISK$ONE/SCRIPT=SECOND_SCRIPT-
/PREDECESSOR=FIRST_SCRIPT`

This commands links SECOND_SCRIPT to FIRST_SCRIPT using the /PREDECESSOR qualifier. SECOND_SCRIPT does not execute until FIRST_SCRIPT completes successfully. SECOND_SCRIPT uses default parameters. Note that this example uses the DEFRAGMENT VOLUME command from within the Disk File Optimizer utility.

5. `$ DEFRAGMENT VOLUME DISK$ONE/SCRIPT=AFTER_SCRIPT/AFTER="+00:05"`

This command creates the script AFTER_SCRIPT, and it will start executing in 5 minutes.

Chapter 5. The DECwindows Interface for Defrag

5.1. Introduction

In addition to the DCL command line interface, the Disk File Optimizer (DFO) software includes a DECwindows interface. The DECwindows interface gives you a visual reference when performing tasks with the Disk File Optimizer software.

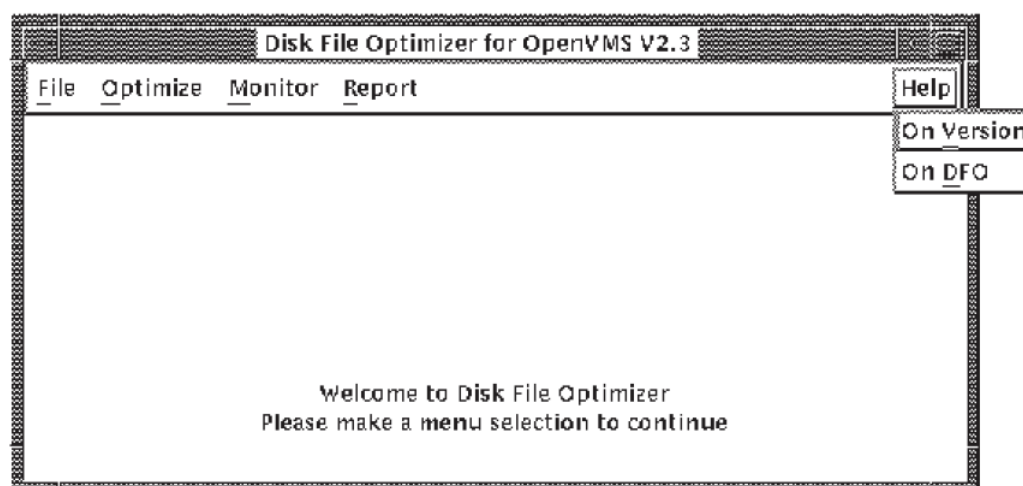
The Disk File Optimizer DECwindows interface allows you to initiate, control, and stop defragmentation operations.

To start the DECwindows interface, enter this command:

```
$ DEFRAGMENT/INTERFACE=DECWINDOWS
```

The Disk File Optimizer initial window appears. *Figure 5.1, "Initial Window"* shows the initial window.

Figure 5.1. Initial Window



The interface allows you to perform the following actions:

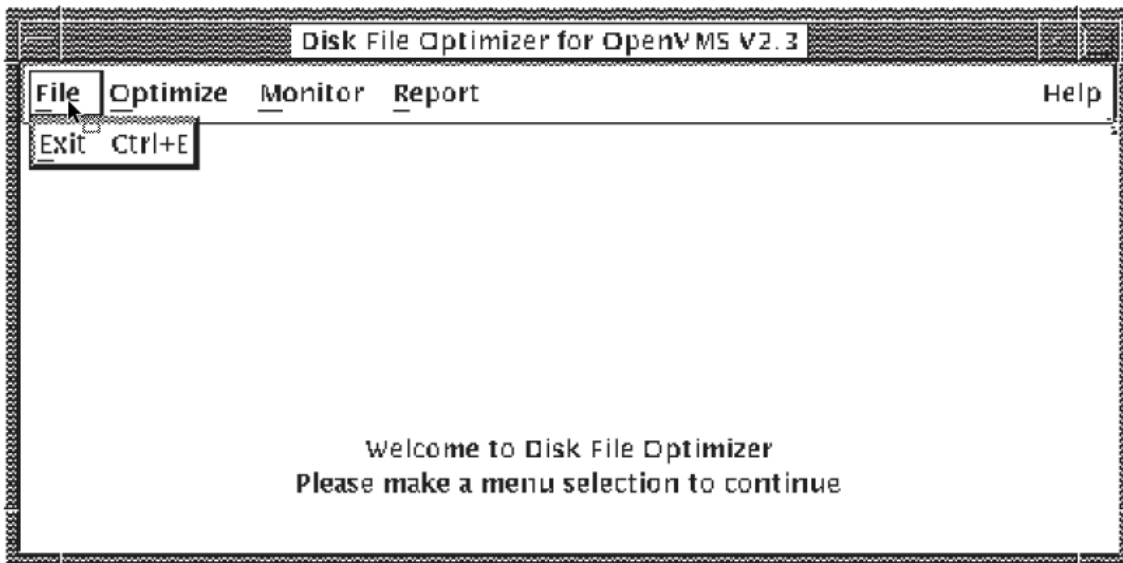
- File – exit the DECwindows interface.
- Optimize – optimize a volume, selected files, or a disks volume index file.
- Monitor – monitor the defragmentation process by selecting a volume to monitor.
- Report – review the current state of a volume to determine if defragmentation is needed.

5.2. File Menu

The File menu serves only one purpose. It allows you to exit this windows interface application. To do so, click on the File topic and bring down the Exit box. Click on Exit to leave this application.

Figure 5.2, "File Menu" shows the File pull-down menu.

Figure 5.2. File Menu

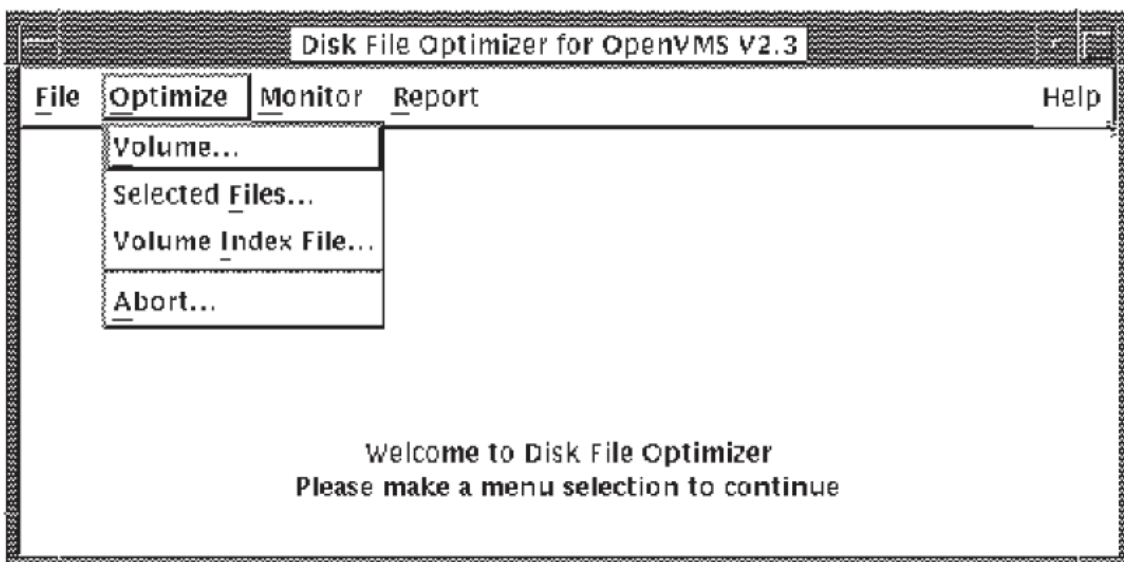


5.3. Optimize Menu

The Optimize menu permits you to optimize a volume, file, volume index file, or abort the optimize process.

Figure 5.3, "Optimize Menu" shows the optimize menu.

Figure 5.3. Optimize Menu



5.3.1. Optimize Volume

The Optimize Volume menu selection permits you to optimize a volume. Use the following procedure to optimize a volume:

1. Pull down the Optimize menu.
2. Select Volume.
3. Define the optimize parameters using the procedures in *Section 5.3.1.1, "Define the Optimize Parameters"*.
4. Restrict special files by pressing the restrict button and using the procedure in *Section 5.3.1.2, "Restrict Special Files"*.
5. Schedule the optimize process using the procedure in *Section 5.3.1.3, "Schedule Optimization"*.
6. Start the optimize process by pressing the OK or Apply button.

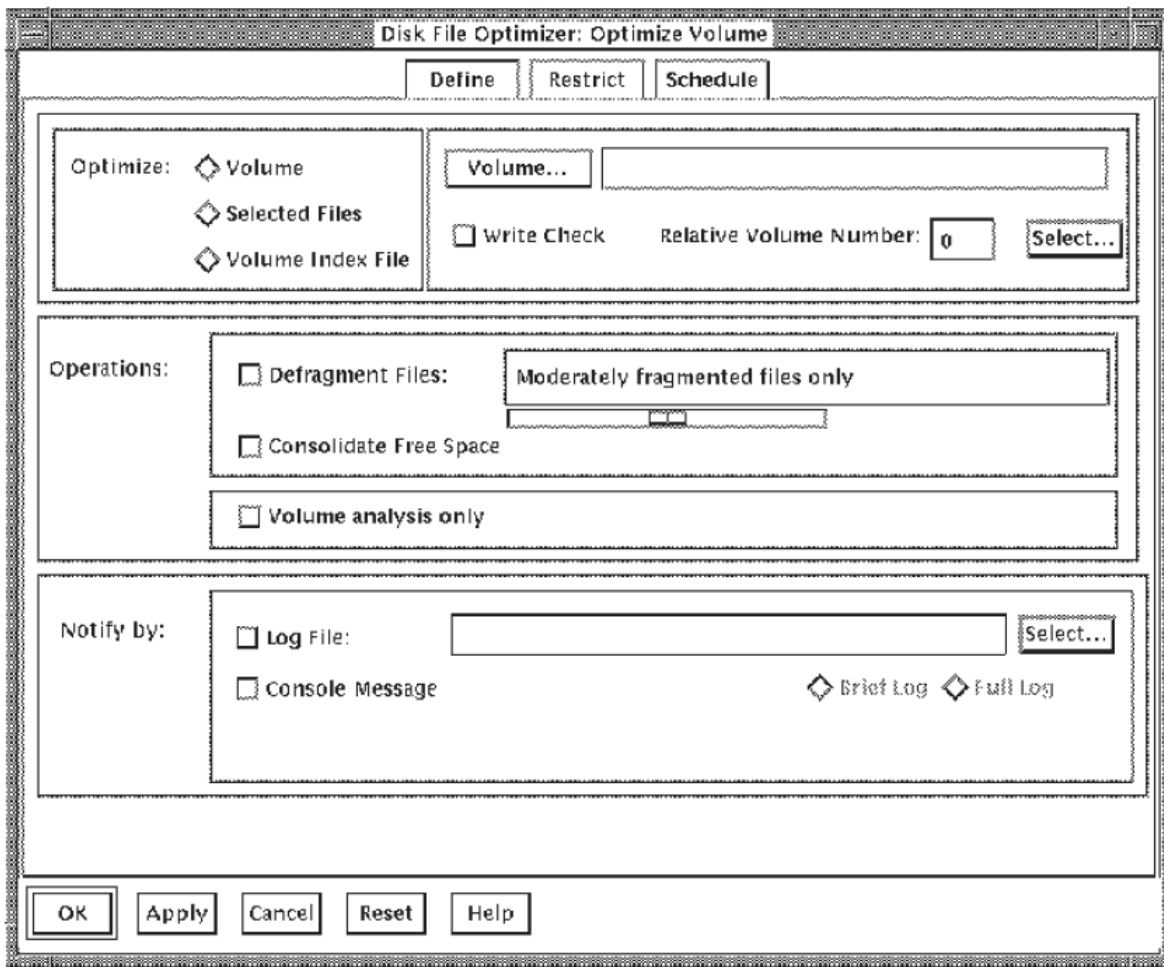
5.3.1.1. Define the Optimize Parameters

Before you can optimize a volume or file, you have to define the optimize parameters using Define dialog box.

In the Define dialog box for a volume, you can choose from the following selections:

- Select a volume name.
- Select to enable or disable write check.
- Select the relative volume number of a volume set.
- Select whether to defragment files and what kind of files to defragment.
- Select whether to consolidate free space or not.
- Select whether or not to do a volume analysis.
- Select whether to notify you by log file or console message.
- Select whether to report brief or full log files.

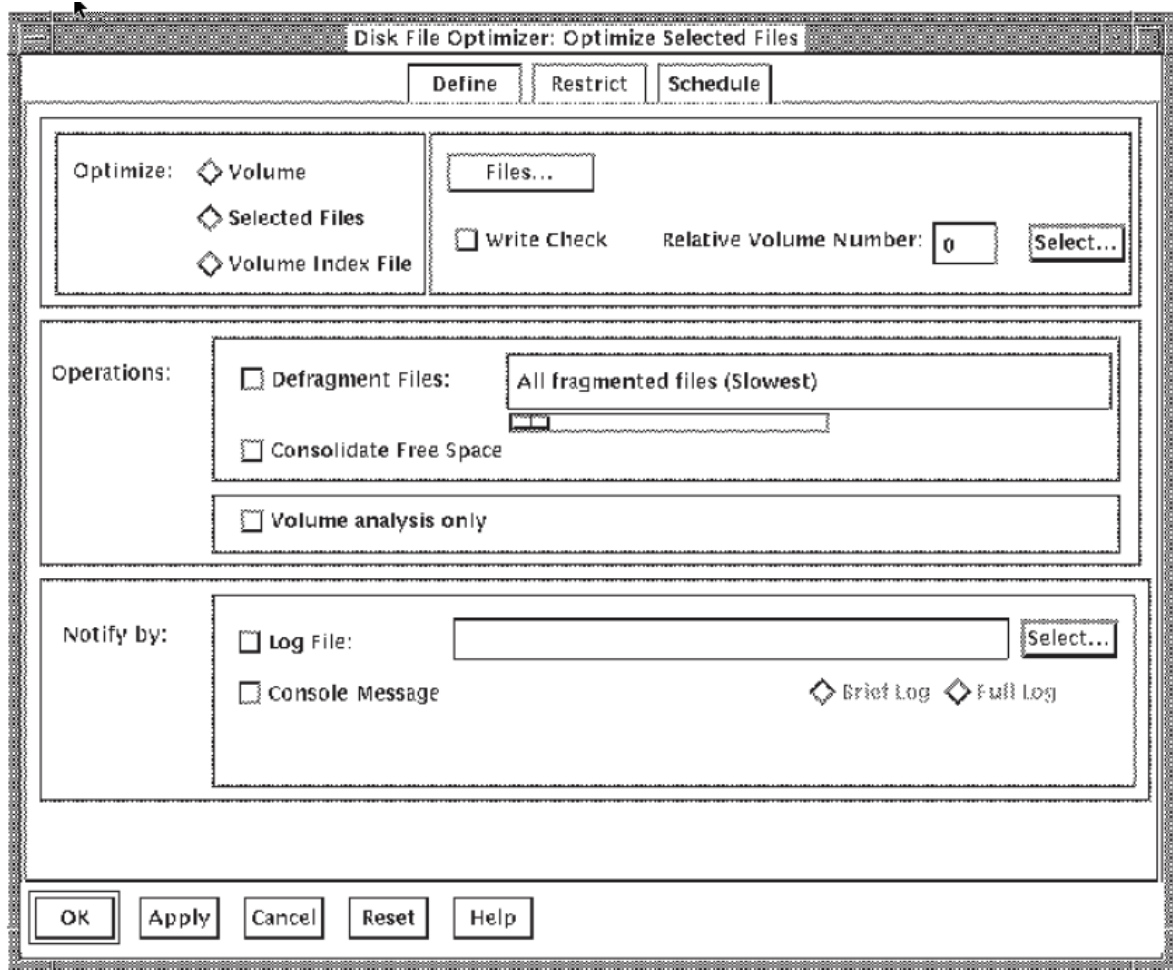
Figure 5.4, "Define Dialog Box for a Volume" shows the define dialog box for a volume. Refer to *Table 5.1, "Define Dialog Box Fields"* for a description of the information fields.

Figure 5.4. Define Dialog Box for a Volume

In the Define dialog box for a file, you can choose from the following selections:

- Select one or more file names.
- Select to enable or disable write check.
- Select the relative volume number of a volume set.
- Select whether to defragment files and what kind of files to defragment.
- Select whether to consolidate free space or not.
- Select whether or not to do a volume analysis.
- Select whether to notify you by log file or console message.
- Select whether to report brief or full log files.

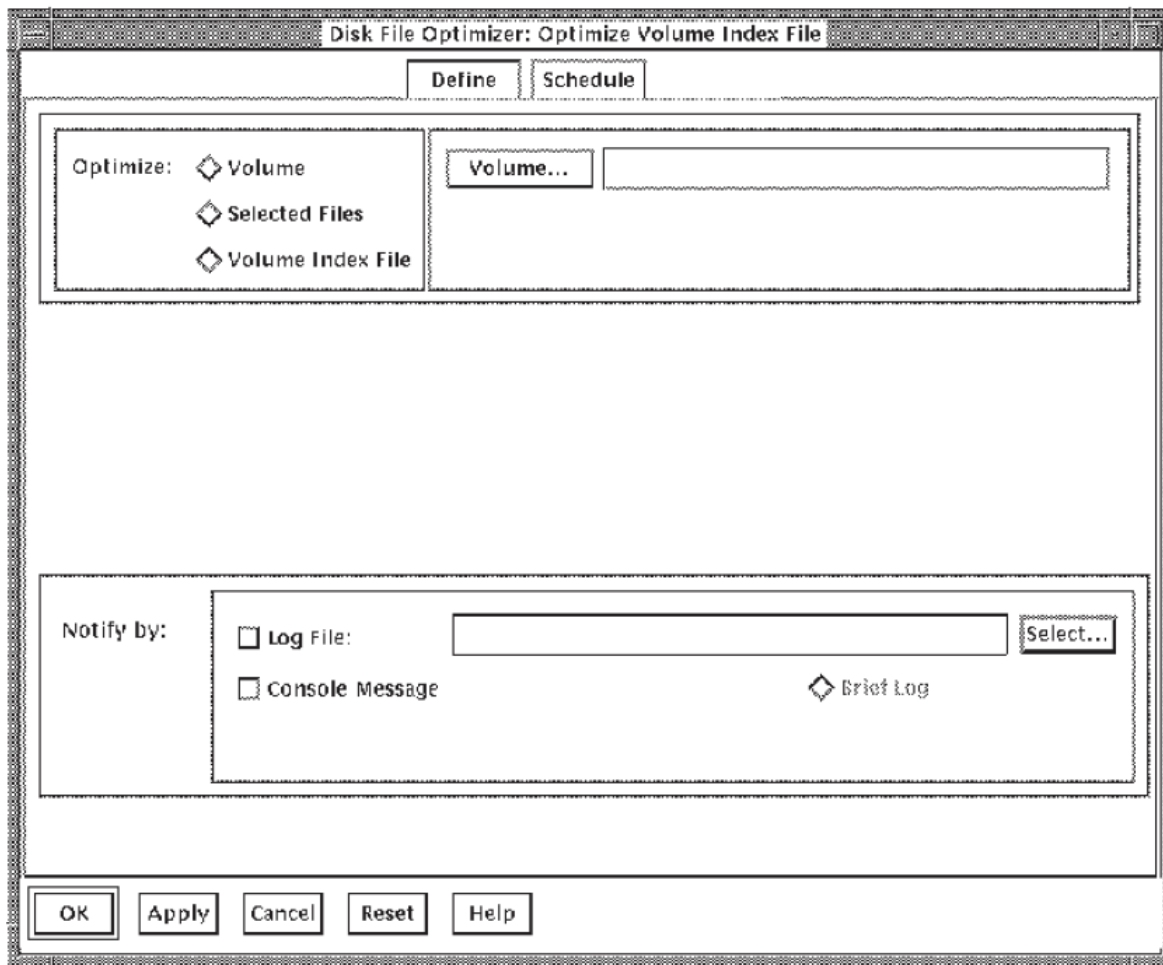
Figure 5.5, "Define Dialog Box for a File" shows the script define dialog box for a file.

Figure 5.5. Define Dialog Box for a File

In the Define dialog box for a volume index file, you can choose from the following selections:

- Select a disk.
- Select whether to notify you by log file or console message.
- Select whether to report brief or full log files.

Figure 5.6, "Define Dialog Box for a Volume Index File" shows the define dialog box for a volume index file.

Figure 5.6. Define Dialog Box for a Volume Index File**Table 5.1. Define Dialog Box Fields**

Field	Description
Volume	Defragments the volume named in the volume field. This has the same effect as using the DEFRAGMENT VOLUME DCL command.
Selected Files	Defragments the file or files named in the files field. This has the same effect as using the DEFRAGMENT FILE DCL command.
Volume Index Files	Defragments the INDEXF.SYS of the disk named in the disk field. This has the same effect as using the DEFRAGMENT OFFLINE_VOLUME DCL command.
Volume...	The Volume... button only appears when the Volume or Volume Index File buttons are depressed. Enter the volume name or use the Volume... button to bring up a list of volumes or disks.
Files...	The Files... button only appears when the Selected Files button is depressed. Push the File... button file to bring up a list of files.
Write Check	The Write Check option requests that a read-after-write check be performed on all defragmented or placed files. This qualifier significantly extends the execution time of a defragmentation process. This has the same effect as the /WRITE_CHECK DCL qualifier.

Field	Description
Relative Volume Number	Allows you to select the relative volume number of a volume set. This has the same effect as the /RVN= DCL qualifier.
Operations	You may elect to defragment specific files, consolidate disk space, or do an analysis check only. File defragmentation and space consolidation can be done together. Defragment Files: Allows you to select which files to defragment. Use the three-position sliding bar: ● LEFT - Files that are not 100% contiguous. ● MIDDLE - Files whose extent cannot be completely mapped. ● RIGHT - Files with retrieval points in more than one file header. Consolidated Freespace: Consolidate the free space on the volume. Volume analysis only: Requests a volume analysis only, no defragmentation occurs. The results are available in the log file. The operations has the same effect as the /LEVEL=x DCL qualifiers as shown in <i>Table 5.2, "Translations of Operations to DCL Command/LEVEL=x"</i>.
Log File	The Log File option causes the defragmentation process to create a file containing a list of all its operations. The information in the log file varies with the presence of the Full or Brief option selections: ● The Brief option lists only the statistics of each phase, such as number of files analyzed and number of files defragmented. ● The Full option lists all files analyzed by the defragmentation process and reports their disposition during each phase of defragmentation. This has the same effect as the /LOG=log_file and /FULL or /BRIEF DCL qualifier.
Console Message	The console message option requests that brief status and termination messages regarding defragmentation processes be sent to OPCOM. This has the same effect as the /OPCOM_NOTIFICATION DCL qualifier.

Table 5.2. Translations of Operations to DCL Command/LEVEL=x

Defragment Files	Sliding Bar	Consolidate Free Space	Volume Analysis Only	/LEVEL=
Selected	LEFT			1
Selected	MIDDLE			2
Selected	RIGHT			3
Not Selected		Selected		4
Not Selected		Not Selected	Selected	5

5.3.1.2. Restrict Special Files

The Restrict Special Files dialog box allows you to include or exclude any special files during the defragmentation process. You can exclude files such as:

- Files placed by the User
- RMS Indexed Files
- Excluded Files

You can also exclude the following types of Placed Files:

- Files in the hotfile database
- Dormant files
- Frequent files

Figure 5.7, "Restrict Special File Dialog Box" shows the Restrict Special Files dialog box for a volume and Table 5.3, "Restrict File of Dialog Box Fields" describes the information fields.

Figure 5.7. Restrict Special File Dialog Box

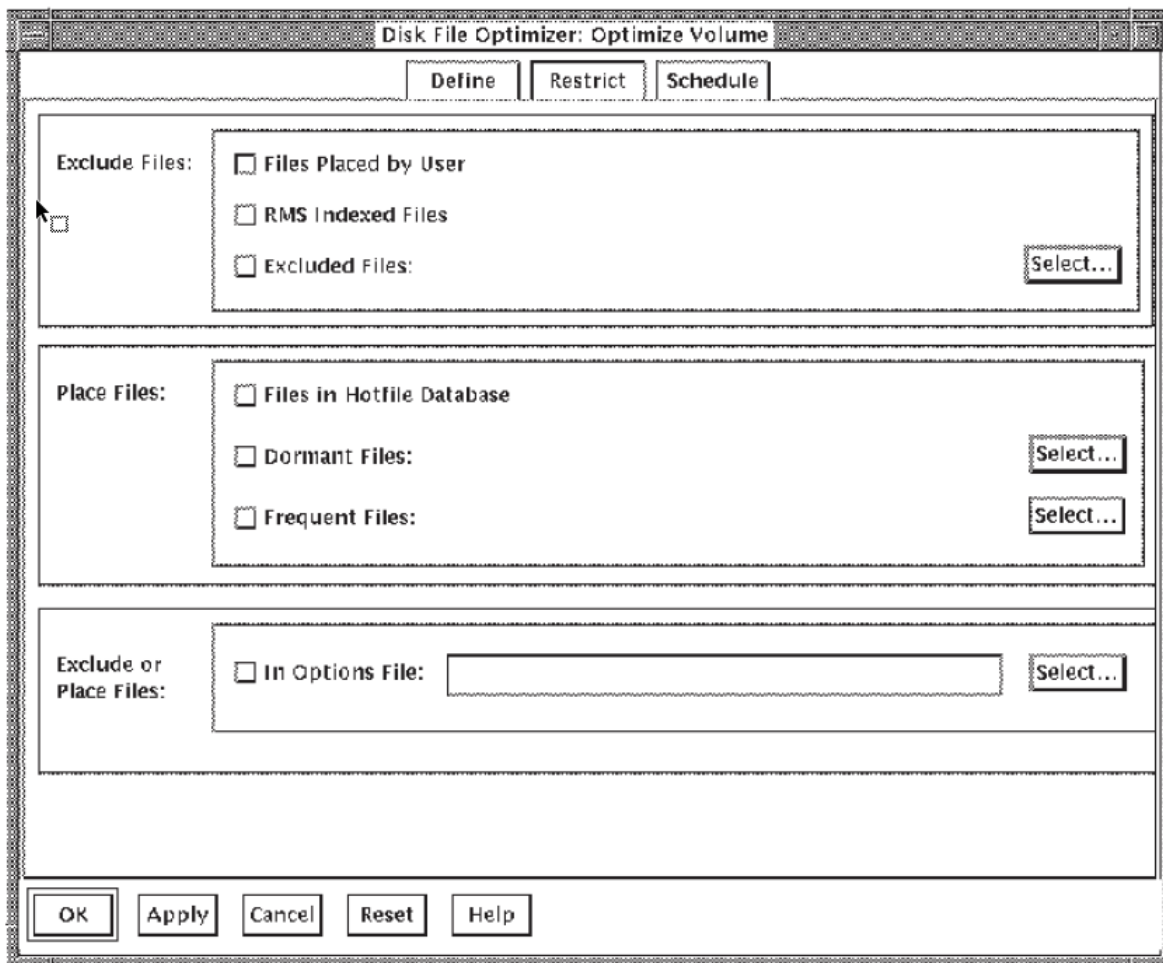


Table 5.3. Restrict File of Dialog Box Fields

Field	Description
Files Placed by User	Select this button if you wish to ignore explicitly placed files. This has the same effect at the /IGNORE=(PLACED) DCL qualifier.
RMS Indexed Files	Select this button if you wish to ignore RMS indexed files.
Excluded Files	The Exclude Files option specifies files to be excluded from processing. You may enter a file name in the file field or use the file list button to bring up a list of files to select from. The defragmentation process ignores files matching the listed file specifications. Any file indicated in the EXCLUDE list and in either the FREQUENT or DORMANT lists is excluded from processing. The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*;*. Specifying *.TMP;* selects all *.TMP;* files on the entire volume. The defragmentation process always excludes files marked NOMOVE, open files, files excluded in an options file, and files excluded using the Ignore option.
Files in Hotfile Database	This option causes the defragmenter to query the appropriate hotfile database and append the list of hotfiles to the defragmenter's frequent file list.
Dormant Files	The Dormant Files option indicates files to be considered dormant when final file placement decisions are made. You may enter the file name or use the file list button to bring up a list of files to chose from. Dormant files are placed on the volume to allow placing other files in optimum disk locations. In general, they are placed further away from the ODS-2 data structures than Frequent files and files not classified in any manner. If a file is indicated in both the DORMANT and FREQUENT lists, it is considered dormant. If a file is indicated in both the EXCLUDE and DORMANT lists, it is excluded. Note The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described. The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*;*. Specifying *.TMP;* selects all *.TMP;* files on the entire volume.

Field	Description
Frequent Files	The Frequent Files option indicates files to be considered frequently accessed when final file placement decisions are made. You may enter the file name or use the file list button to bring up a list of files to chose from. Frequent files are placed to optimize access speed. If a file is indicated in both the FREQUENT and DORMANT lists, it is considered dormant. If a file is indicated in both the FREQUENT and EXCLUDE lists, it is excluded. Note The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described. The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*;*. Specifying *.TMP;* selects all *.TMP;* files on the entire volume.
In Options File	The Options File option provides an alternate method for associating files with the Dormant File, Exclude File, or Frequent File options. You may enter the file name or use the file list button to bring up a list of files to chose from. If the list of files for any of these options is long, you can create a separate file for this information. Each record of the options file consists of a file name followed by one of these keywords: ● /EXCLUDE—File to be excluded ● /FREQUENT—File to be considered frequently accessed ● /DORMANT—File to be considered dormant The following example shows the contents of an options file called MYOPTFILE.OPT. This file specifies that all files on the target device called FILE1.EXE are frequent files, all files in the [ARCHIVE] directory are to be excluded, and all files in the [OLDSTUFF] directory and its subdirectories are dormant. <pre>FILE1.EXE/FREQUENT [ARCHIVE]*.*;*/EXCLUDE [OLDSTUFF...] *.*;*/DORMANT</pre>

5.3.1.3. Schedule Optimization

The Schedule Definition dialog box allows you choose many different defragmentation schedule parameters. The types of parameters that you'll find in the dialog box are:

- Wait for completion or not
- CPU Limit
- I/O Limit
- Time limit
- Optimize process priority
- Execute a prologue command file before running the optimize process
- Execute an epilogue command file after running the optimize process

Figure 5.8, "Schedule Definition" shows the Schedule Definition dialog box and Table 5.4, "Schedule Definition Dialog Box Fields" describes the dialog box information fields.

Figure 5.8. Schedule Definition

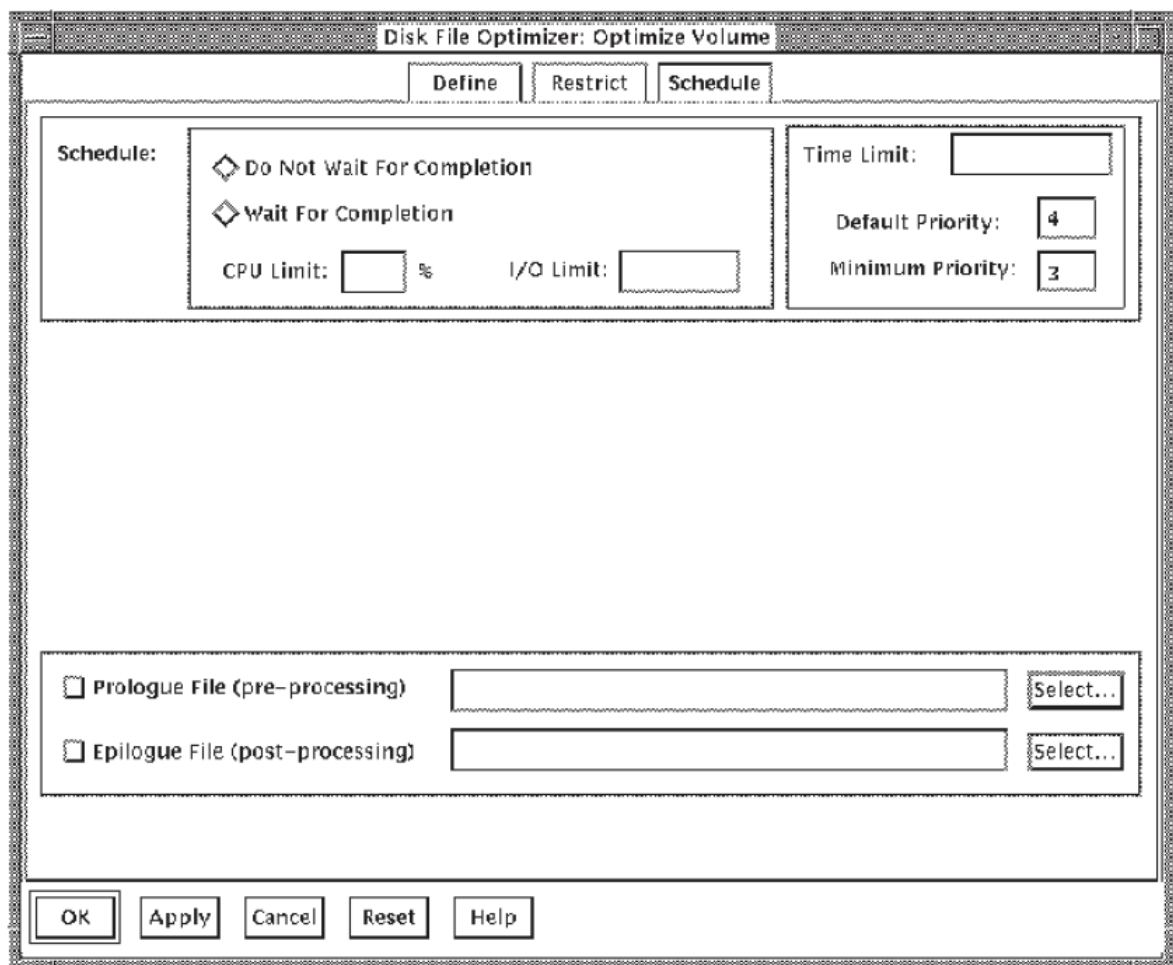


Table 5.4. Schedule Definition Dialog Box Fields

Field	Description
Do Not Wait For Completion	This option allows the interface to be free while the optimization process completes. Selecting the Wait For Completion keeps the interface under the defragmentation process until it completes. This has the same effect as the /WAIT DCL qualifier.
CPU Limit:	Enter into this field the percentage of CPU usage for any one optimization process. The CPU percent is the percent of a single CPU resources. This has the same effect as the /CPU_LIMIT qualifier.
I/O Limit:	Enter into this field the maximum direct I/O rate (per second) of any one optimize process. This has the same effect as the /IO_LIMIT qualifier.
Time Limit:	Specifies the delta time allowed for a given defragmentation operation. This has the same effect as the /TIME_LIMIT= DCL qualifier.
Priority	This is the priority that the OpenVMS scheduler uses for the defragment volume operation. The default priority is set at 4. The minimum priority is set at 3. This has the same effect as the /PRIORITY=(DEFAULT=n, MINIMUM=n) DCL qualifier.
Prologue File	The Prologue File option identifies a DCL command file that DFO executes just before the defragmentation operation starts. This has the same effect as the /PROLOGUE= DCL qualifier.
Epilogue File	The Epilogue File option identifies a DCL command file that DFO executes upon completion of the defragmentation operation. This has the same effect as the /EPILOGUE= DCL qualifier.

5.3.2. Optimize Selected Files

The Optimize Selected Files menu selection permits you to optimize a particular file or files.

1. Pull down the Optimize menu.
2. Select Volume.
3. Define the optimize parameters using the procedures in *Section 5.3.1.1, "Define the Optimize Parameters"*.
4. Restrict special files by pressing the restrict button and using the procedure in *Section 5.3.1.2, "Restrict Special Files"*.
5. Schedule the optimize process using the procedure in *Section 5.3.1.3, "Schedule Optimization"*.
6. Start the optimize process by pressing the OK or Apply button.

5.3.3. Optimize Volume Index File

The Optimize Volume Index File selection permits you to defragment the INDEXF.SYS file from a selected disk.

1. Pull down the Optimize menu.
2. Select Volume.
3. Define the optimize parameters using the procedures in *Section 5.3.1.1, "Define the Optimize Parameters"*.
4. Restrict special files by pressing the restrict button and using the procedure in *Section 5.3.1.2, "Restrict Special Files"*.
5. Schedule the optimize process using the procedure in *Section 5.3.1.3, "Schedule Optimization"*.
6. Start the optimize process by pressing the OK or Apply button.

5.3.4. Abort an Optimize Process

Use this option to abort an optimize process that is currently executing.

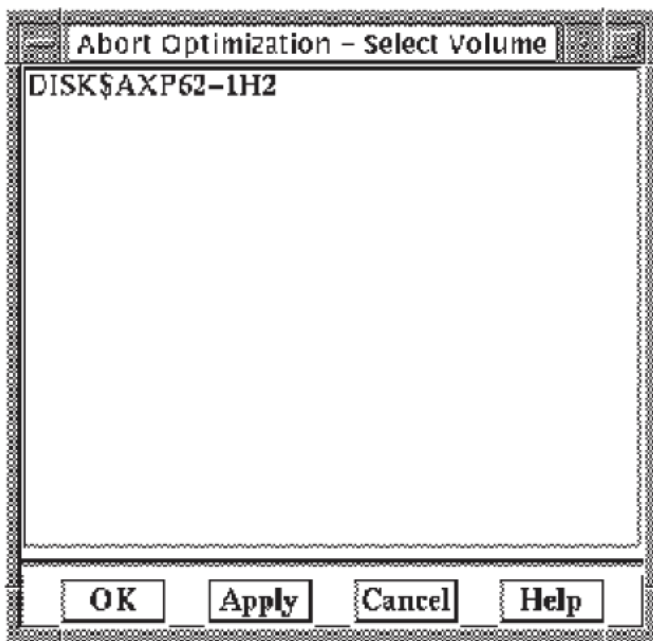
Use the following procedure to abort an active optimize process:

1. Select the Optimize pull-down menu.
2. Select the Abort option.
3. Select the volume to abort from the Volume List box. Use caution because a double click on a entry will abort that optimize process.

Point to any one or more of the volumes listed, clicking once on each one you want to abort.

4. Finish the abort action by selecting:
 - OK to abort the selected volumes and exit the Abort option
 - Apply to abort the selected volumes and remain in the Abort option
 - Cancel to leave the Abort option without taking any action

Figure 5.9, "The Abort Dialog Box" shows a list of volumes in the Abort dialog box.

Figure 5.9. The Abort Dialog Box

5.4. Monitor a Volume

The Monitor menu allows you to monitor the defragmentation process by selecting a volume. You can view how the space is allocated on the full volume or on a partial volume in a sliding window. When selecting the Monitor option, DFO displays two windows:

- A process statistics window showing process information for the defragmentation process. Refer to *Figure 5.12, "Defragmentation Process Statistics Window"* and *Table 5.5, "Defragmentation Process Statistics Window Description"*.
- A monitor map window showing free space and allocated space on the volume. Refer to *Figure 5.13, "Defragmentation Full Volume Map Window"* and *Figure 5.14, "Defragmentation Partial Volume Map Window"*.

This procedure describes how to monitor allocated space on a disk volume.

1. Select the Monitor pull-down menu.
2. Select the means by which you want to monitor allocated space. Select the Volume or Full Volume option of the Monitor pull-down menu. Refer to *Figure 5.10, "Monitor Menu"*.
 - **IF** you want to identify the volume by name, select the Volume option, then select the volume name from the Volume List. To view different parts of the entire volume map, use the arrows and scroll bars around the map on the defragmentation volume map window. If you chose to monitor the Volume, a Defragmentation Process Statistic Window and a Defragmentation Partial Volume Map Window will appear. Refer to *Figure 5.12, "Defragmentation Process Statistics Window"*, *Figure 5.14, "Defragmentation Partial Volume Map Window"* and *Table 5.6, "Defragmentation Volume Map Window Description"*.
 - **IF** you want to view the full volume, select the Full Volume option, then select the volume name from the Volume List. If you chose to monitor the Full Volume, a Defragmentation Process Statistic Window and a Defragmentation Full Volume Map Window will appear. Refer to *Figure*

5.12, "Defragmentation Process Statistics Window", Figure 5.13, "Defragmentation Full Volume Map Window" and Table 5.5, "Defragmentation Process Statistics Window Description".

3. To view more detailed information about the allocated space shown on the Defragmentation Volume Map, press mouse button 1 and drag the cursor over the area of the map you want to examine. A pop-up window identifies the files taking space on that area of the volume.

Figure 5.11, "Volume Map File Information" is an example of the volume map file information.

Figure 5.10. Monitor Menu

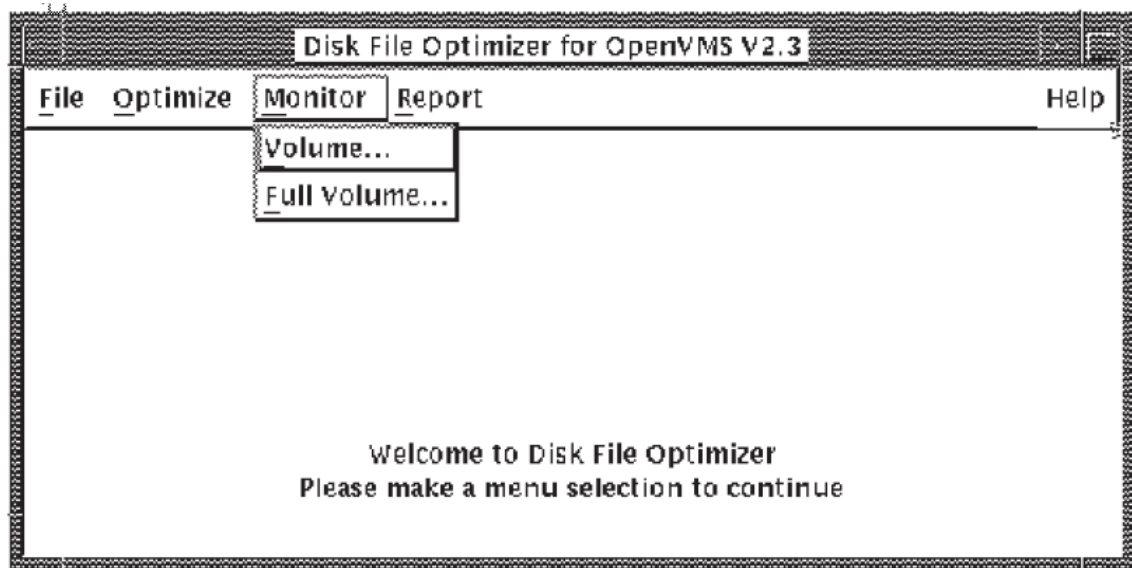
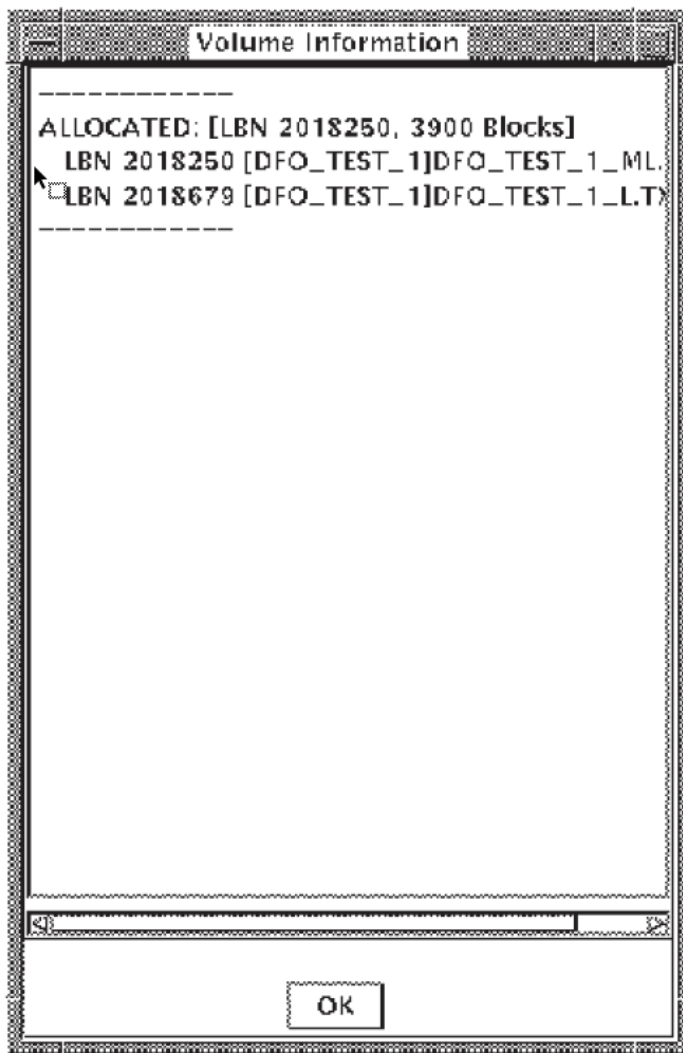


Figure 5.11. Volume Map File Information

5.4.1. Defragmentation Process Statistics Window

The Defragmentation Process Statistics Window shows the process information and statistics on how files and space on the volume are allocated during defragmentation.

Figure 5.12, "Defragmentation Process Statistics Window" shows the Defragmentation Process Statistics window, and Table 5.5, "Defragmentation Process Statistics Window Description" describes the data field information.

Figure 5.12. Defragmentation Process Statistics Window

Disk File Optimizer: Process Statistics

Process Id: Process: Date:

Script Name:

Operating On: Current Status:

Phase: Elapsed Time:

Buffered I/O Direct I/O CPU Utilization Page Fault

Count: Count: Time: Count:

Rate: Rate: Usage: Rate:

Files Evaluated : Dormant Count:

Candidate Count: Exclude Count:

Placed Files: Frequent Count:

Indexed Files:

Files Partly Defragmented: Dormant Files Placed:

Files Defragmented: Frequent Files Placed:

Files Processed: Free Space Extent Count:

Free Space Extent Countdown:

Current File

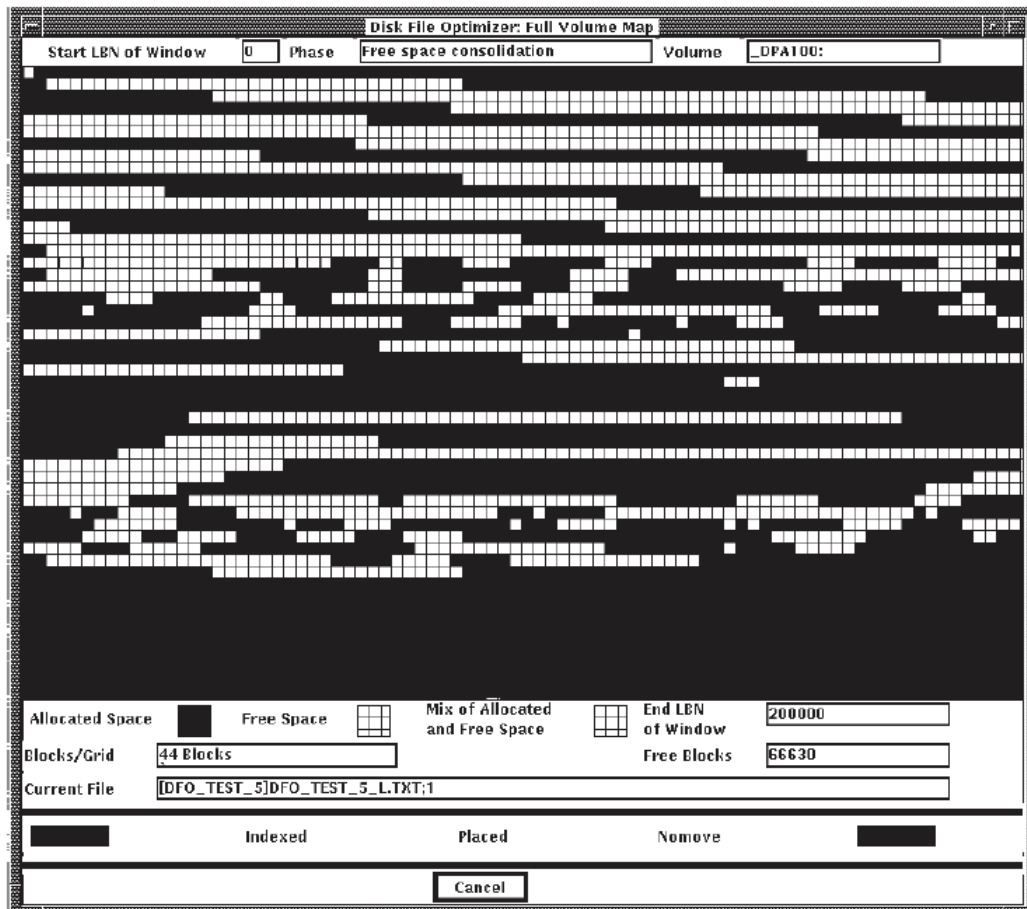
Table 5.5. Defragmentation Process Statistics Window Description

Field	Description
Process Id:	The defragmentation process ID number.
Process:	The defragmentation process name.
Date:	The current date and time.
Script Name:	The name of the script being monitored. (The field applies to the Defrag PLUS variant only.)
Operating On:	The node on which the defragmentation process is operating.
Current Status:	The status of the defragmentation process.
Phase:	The current defragmentation process phase.
Elapsed Time:	The elapsed time since the start of the defragmentation process.
Buffered I/O Count: and Rate:	The buffered I/O count and rate of the defragmentation process.
Direct I/O Count: and Rate:	The direct I/O count and rate of the defragmentation process.
CPU Utilization Time: and Usage	The CPU utilization (time and usage) of the defragmentation process.

Field	Description
Page Fault Count: and Rate:	The page fault count and rate of the defragmentation process.
Files Evaluated:	Number of files evaluated.
Dormant Count:	Number of dormant files.
Candidate Count:	Number of candidate files for defragmentation.
Exclude Count:	Number of excluded files.
Placed Files:	Number of placed files.
Frequent Count:	Number of frequent files.
Indexed Files:	Number of indexed files.
Files Partly Defragmented:	Number of files partly defragmented.
Dormant Files Placed:	Number of dormant files placed.
Files Defragmented:	Number of files defragmented.
Frequent Files Placed:	Number of frequent files placed.
Files Processed:	Number of files processed.
Free Space Extent Count:	Number of extents to be consolidated into free space.
Free Space Extent Countdown:	Number of extents left to be consolidated into free space.
Current File:	Current file being evaluated.

5.4.2. Full Volume Window

The defragmentation full volume map window allows you to see how the storage space and free space on the full volume are allocated. *Figure 5.13, "Defragmentation Full Volume Map Window"* show the full volume map window and *Table 5.6, "Defragmentation Volume Map Window Description"* describes the window contents and how it functions.

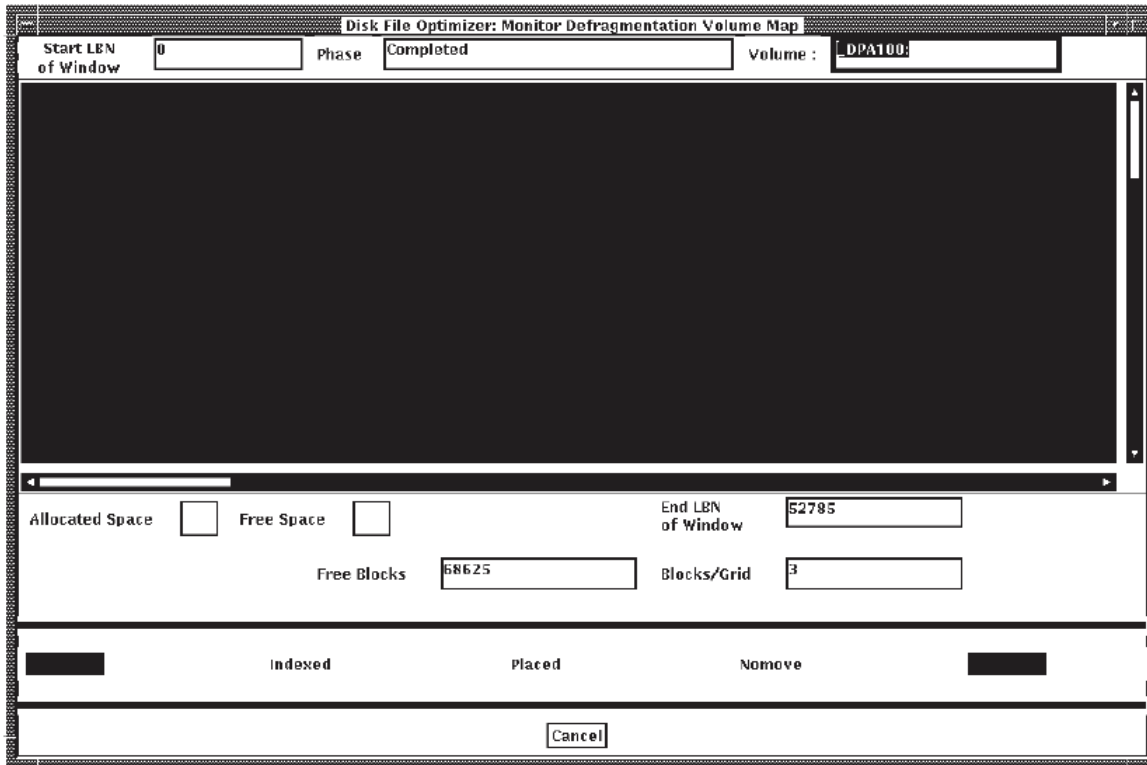
Figure 5.13. Defragmentation Full Volume Map Window**Table 5.6. Defragmentation Volume Map Window Description**

Field	Description
Start LBN of Window	The starting LBN at the upper left corner of the viewport.
Phase	The current phase of the defragmentation process.
Volume	The name of the volume that is being defragmented.
Allocated	Space Shows the color of allocated space on the disk volume.
Free Space	Shows the color of free space on the disk volume.
Mix of Allocated and Free Space	Shows the color of space that is partially allocated and free space. This entry is only on the full volume map.
End LBN of Window	The ending LBN at the lower right corner of the viewport.
Free Blocks	The number of free blocks on the disk volume.
Blocks/Grid	The number of blocks in each grid on the view port.
Current File	The current file the defragmentation process is working on. This entry is only on the full volume map.

5.4.3. Partial Volume Window

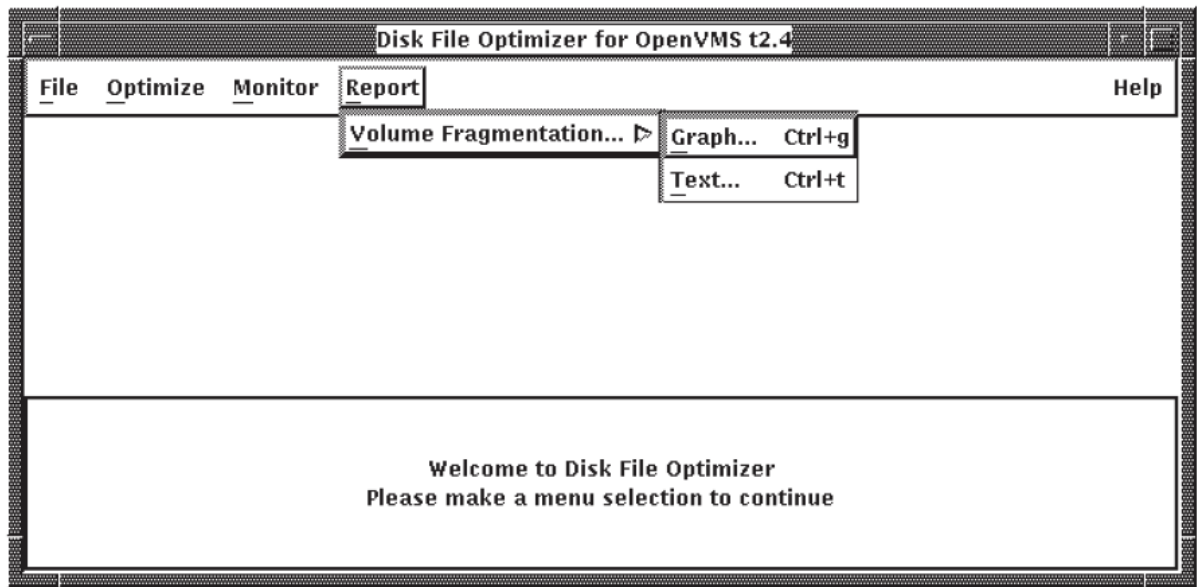
The partial volume map window provides a closeup detailed view of how the storage space is allocated on a smaller segment of the whole volume. *Table 5.6, "Defragmentation Volume Map Window Description"* describes the window contents and how it functions.

Figure 5.14. Defragmentation Partial Volume Map Window



5.5. Report on the State of a Volume

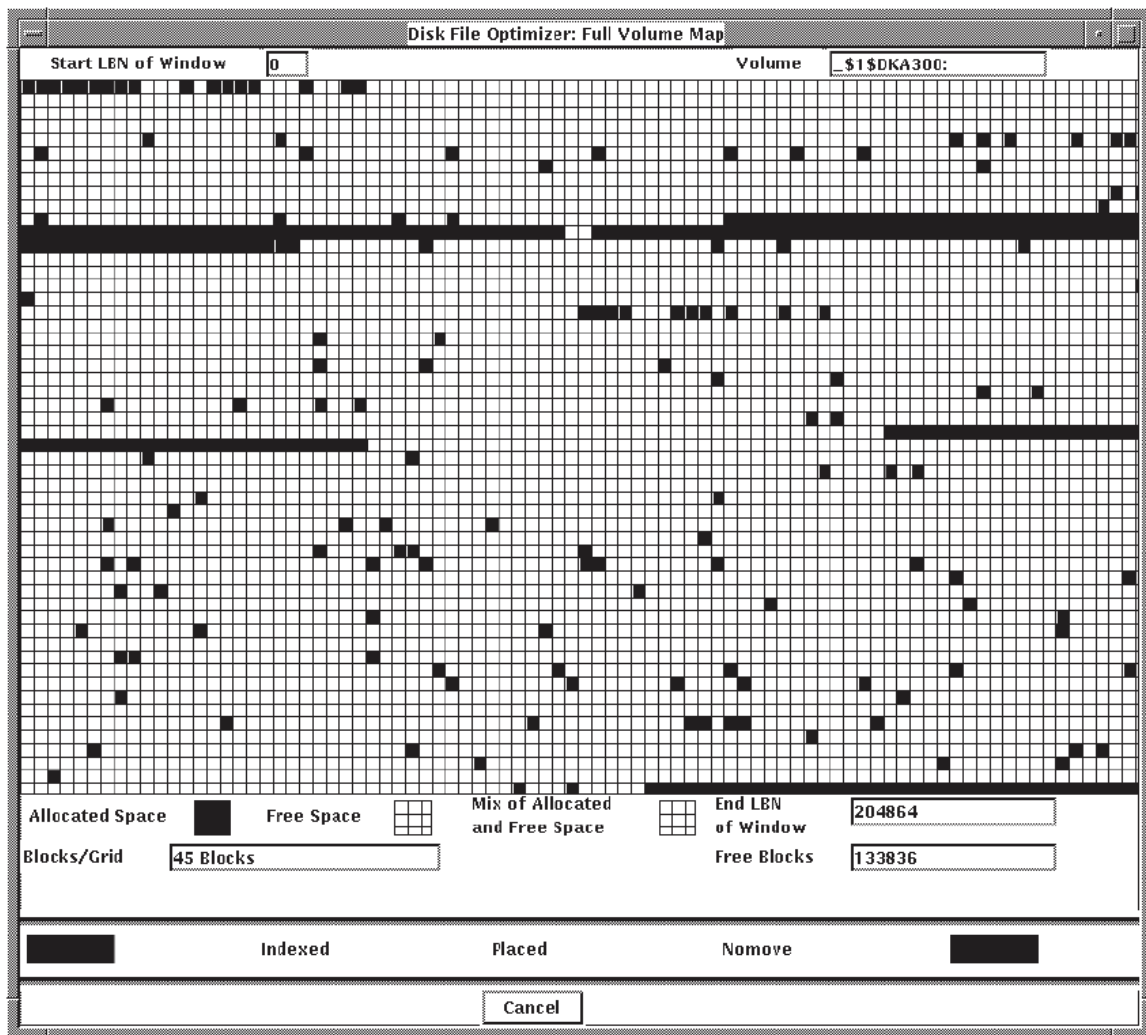
The report menu allows you to review the current state of disk volumes to determine if defragmentation is needed. Click on the Volume Fragmentation option of the Report menu for the two options it supports. See *Figure 5.15, "Report Menu"*. You can opt for a graphic or a text report of the state of the volumes.

Figure 5.15. Report Menu

5.5.1. Graphics Format

Use the following procedure for a graphics display of the selected volume(s).

1. Click on the Graph... suboption of the Report-Volume Fragmentation menu. See *Figure 5.15, "Report Menu"*. The Report - Select Volume dialog appears. See *Figure 5.17, "The Report Volume Selection Dialog Box"* and *Table 5.7, "Report Volume Option and Field Descriptions"*.
2. Note that the first entry in the list is marked selected by default. You can select any other volume for the respective report. To select another volume, move mouse pointer to it and click.
3. Click on Ok. A graphical report of the state of the selected volume(s) appears on screen. See *Figure 5.16, "Report Graphics Full Volume Map"*. Click on Cancel to close this dialog.

Figure 5.16. Report Graphics Full Volume Map

This dialog is nothing but a color coded display of how the volume is shared by its files and directories. While Start LBN and Volume are self explanatory, the following may require explanation:

- The colors green, cyan, blue, magenta and red represent directories, indexed files, placed files, no move files and open files respectively.
 - Three gray levels mark different extents. Darkest gray - allocated space, Lightest gray - free space and medium gray - mix of allocated and free space.
 - Blocks/grid, End LBN of Window and Free Block, are again self explanatory.
4. Click on a directory (green) or an open file (red) for the Volume Map File Information dialog see - *Figure 5.11, "Volume Map File Information"*. It has details concerning the selected entity with respect to its volume.
 5. Click on OK to close that dialog.

Note

Click on Apply in the Graphic - Full volume Map dialog to find out if the selected volume is being defragmented currently.

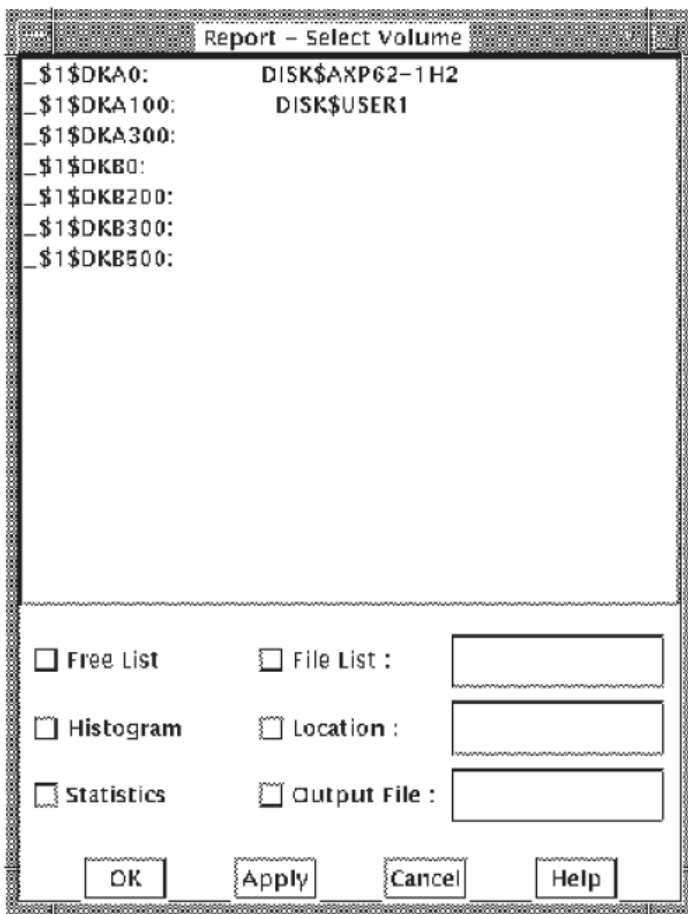
5.5.2. Text Format

Use the following procedure for a text display of the selected volume(s).

1. Click on the Text... suboption of the Report-Volume Fragmentation menu. See *Figure 5.15, "Report Menu"* for the Report - Select Volume dialog.

Use the following procedure to show volume information:

- a. Select the Report pull-down menu. See *Figure 5.15, "Report Menu"*.
- b. Click on the Volume fragmentation option of the Report pull-down menu for the report type list - Graph/Text report. Click on 'Text' to select it. You will see the 'Report - Select Volume' dialog. See *Figure 5.17, "The Report Volume Selection Dialog Box"* and to *Table 5.7, "Report Volume Option and Field Descriptions"*.
- c. Select a volume from the list in the Show Volume options dialog box.
- d. Select any of the options for showing volume information. *Table 5.7, "Report Volume Option and Field Descriptions"* describes the options.
 - IF you keep the Statistics option(default), and double click on the volume or click on 'OK' you will see a scrollable page of a Fragmentation Report showing fragmentation index, Freespace Summary, and File Fragmentation Summary, plus more information that you can scroll down. Refer to *Figure 5.18, "Report Volume: Statistics"*.
 - IF you select the Histogram option, you will see that portion of the Fragmentation Report that shows the File Fragmentation Histogram for the number of files with a given number of extents and the Volume Free Space Histogram which shows the free space size in LBNs. Refer to *Figure 5.19, "Report Volume: File Fragmentation Histogram"* and *Figure 5.20, "Report Volume: Volume Space Histogram"*.
- e. Finish the Report Volume Defragmentation operation by selecting:
 - OK to show the volume report and optionally create the output file you named in the Output File field. The dialog box disappears.
 - Apply to show the volume report and optionally create the output file you named in the Output File field. The dialog box remains.
 - Cancel to leave the Report option.

Figure 5.17. The Report Volume Selection Dialog Box**Table 5.7. Report Volume Option and Field Descriptions**

Field	Description
Free List	The Free List option is used to show the fragmentation state of the volume. It lists the location and size of all the free space extents on the volume.
Histogram	The Histogram option is used to show the fragmentation state of a volume. It generates two histograms. The first plots the number of files against the number of extents per file and provides information about how many files are fragmented and how badly they are fragmented. The second histogram plots the number of free space extents (holes) against the size of these free space extents. It provides information about how many free space extents are on the volume and their sizes.
Statistics	The Statistics option is used to show the volume fragmentation state. It provides a summary of the file and free space fragmentation for the volume.
File List	The File List option is used to show the fragmentation state of a volume. It lists all files on the volume that have min_extents or more extents (fragments). By default, the number of file window mapping pointers set for the volume (INITIALIZE {volume} /WINDOW=n) plus one is used. This value lists all files that require at least one window turn to completely map. Use the field to specify the files for which you want to show the window mapping.

Field	Description
Location	The Location option shows the extents (fragments) associated with all files specified in the field next to this option. If no files are specified, the Location option appends location information to all files listed with File List option. The supplied file-spec may contain wildcards in place of the directory, file name, extension, or version fields.
Output File	The Output File option is used to show the volume fragmentation state or to show information from the scheduling database. It directs the output of the SHOW subcommand to a file.

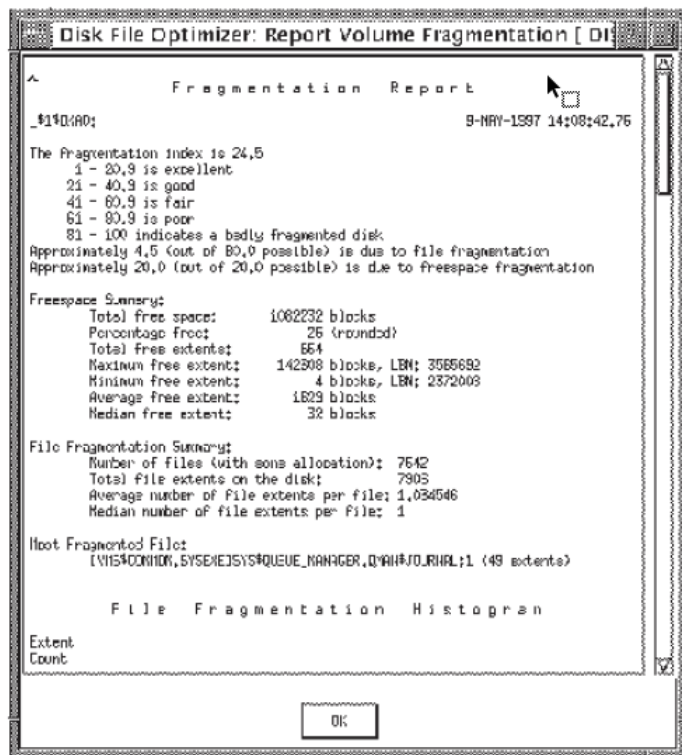
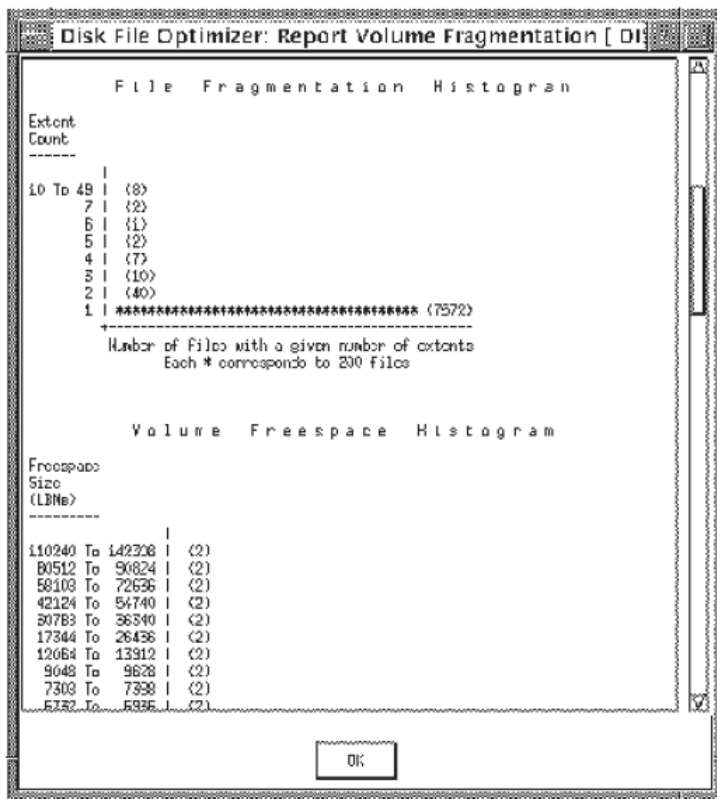
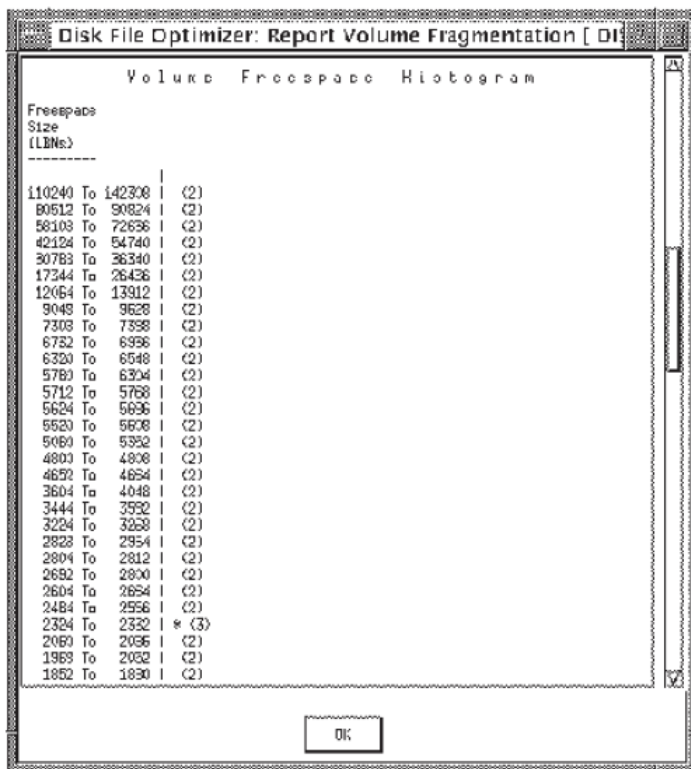
Figure 5.18. Report Volume: Statistics

Figure 5.19. Report Volume: File Fragmentation Histogram**Figure 5.20. Report Volume: Volume Space Histogram**

Chapter 6. The DECwindows Interface for Defrag Plus

In addition to the DCL command line interface, the Disk File Optimizer (DFO) software includes a DECwindows interface. This interface gives a visual reference for tasks built into the DFO software.

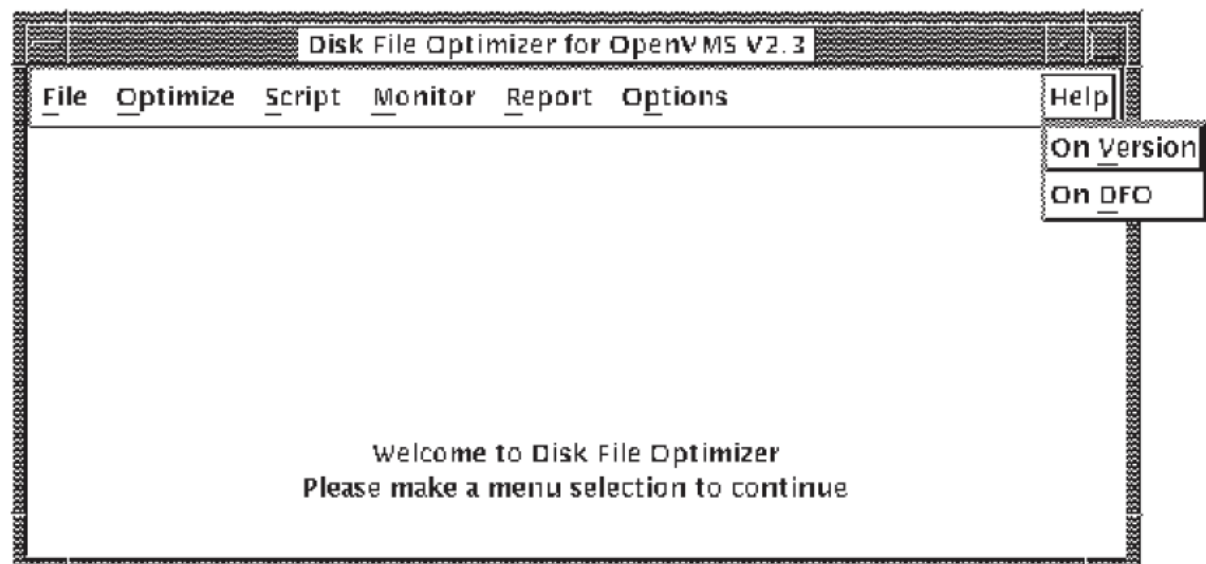
6.1. Introduction

The DFO DECwindows interface allows you to initiate, control, and stop defragmentation operations. To start the DECwindows interface, enter this command:

```
$ DEFRAGMENT / INTERFACE=DECWINDOWS
```

The Disk File Optimizer initial window appears. *Figure 6.1, "Initial Window"* shows the initial window.

Figure 6.1. Initial Window

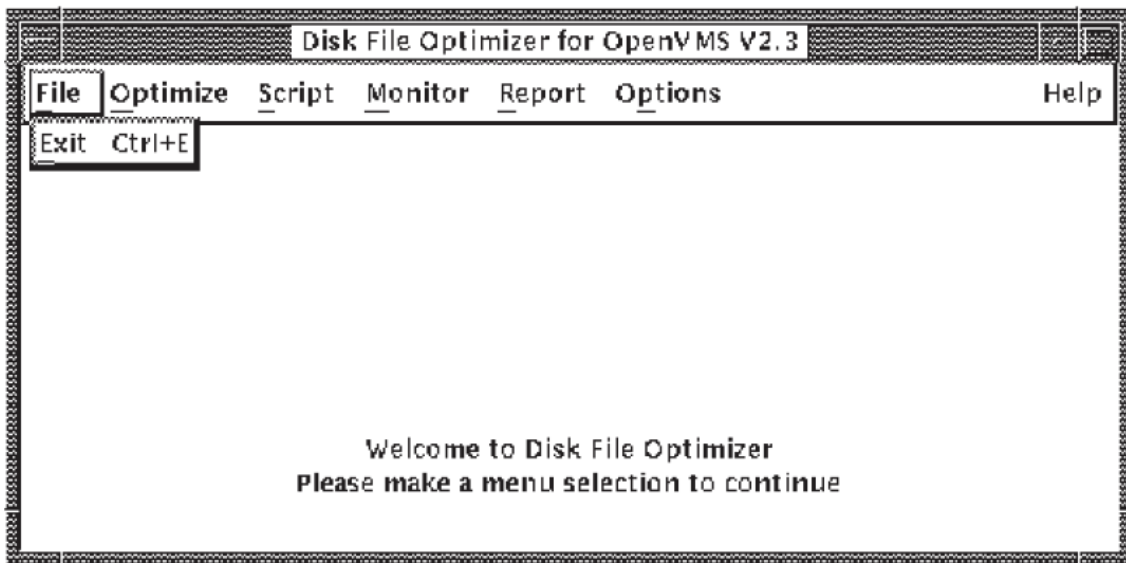


The interface allows you to perform the following actions:

- File – exit the DECwindows interface.
- Optimize – create a script to optimize a volume, selected files, or a disks volume index file.
- Script – create, modify, remove, show, schedule, and abort scripts for files or volumes.
- Monitor – monitor the defragmentation process by selecting a script or volume to monitor.
- Report – review the current state of a volume to determine if defragmentation is needed.
- Options – set or show the Disk File Optimizer process parameters.

6.2. File Menu

The File menu serves only one purpose. It allows you to exit this instance of Disk File Optimizer application. To do so, click on the File menu for the Exit option. Click on 'Exit' to close and exit this application. *Figure 6.2, "File Menu"* shows the File pull-down menu.

Figure 6.2. File Menu

6.3. Optimize Menu

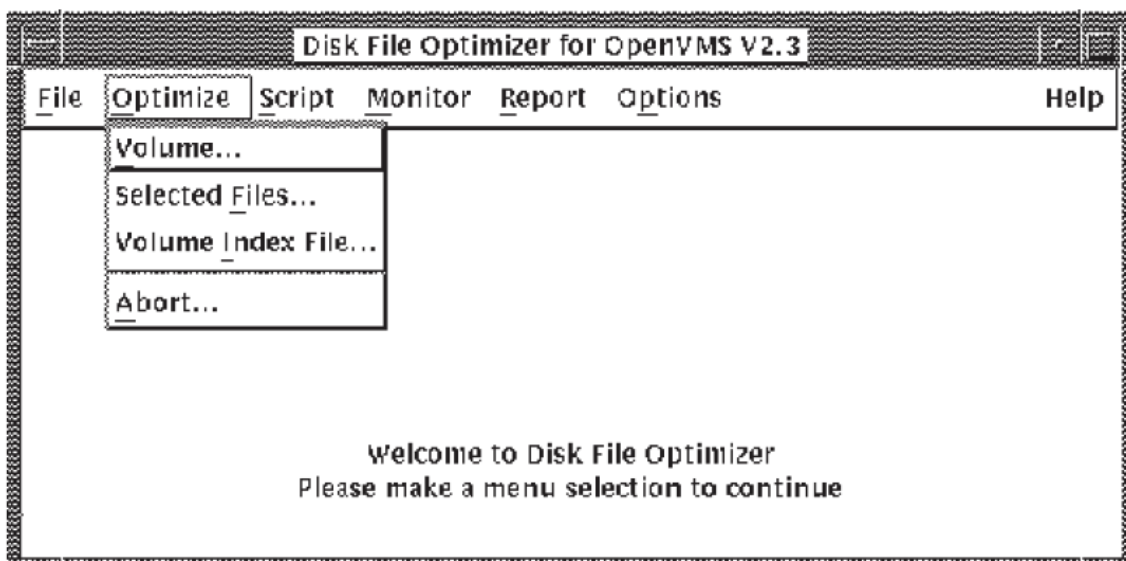
The Optimize menu has options for:

- Optimizing specific targets - a volume, file or an index file
- aborting the optimize process

Note that while you use the 'Optimize' option, you will create a script to optimize the 'target'. This step is the same as creating a script from the 'Scripts' menu.

Use the following procedure to optimize a volume or a file.

1. Pull down the Optimize menu see *Figure 6.3, "Optimize Menu"*.

Figure 6.3. Optimize Menu

2. Select the target you wish to optimize:

- a volume
- selected files
- a volume index file

The 'Create Script to optimize a Volume' appears as in *Table 6.1, "Define Dialog Box Fields"*

3. To specify a script name type the script name in the script field box. If you do not enter a script name, this instance of the Disk File Optimizer application will assign a default script name for you that resembles DFG\$SCRIPTxxx, where x is any hexadecimal character between 0 through F. This default script will be deleted upon successful completion of the defragmentation process.

- Click on 'Reset' to restore the content of all fields to their defaults.
- Click on 'Inherit' to browse a script list and select the script you wish to inherit properties from.

4. Choose a Volume, Selected Files, or Volume Index File.

- If you choose the Volume option, enter a specific Volume name on the Volume insert line or click on the volume list button to get a list of volumes to select from. Refer to *Figure 6.5, "Script Define Dialog Box for a Volume"*.
- If you choose the Selected Files option, enter a specific file name on the file insert line or click on the file list button for a list of files to select from. Refer to *Figure 6.6, "Script Define Dialog Box for a File"*.
- If you choose the Volume Index File option, enter a specific disk name or names on the Disk insert line or click on the disk list button for a list of disks to select from. Refer to *Figure 6.7, "Script Define Dialog Box for a Volume Index File"*.

5. Fill out the rest of the Define dialog box according to the field description information in *Table 6.1, "Define Dialog Box Fields"*.

6. Click on the Restrict button in this dialog box and fill in the information fields according to the information in *Section 6.4.1.2, "Restrict Special Files Dialog Box"*.

7. Click on the Schedule button in this dialog box and fill in the schedule script information fields according to the procedure in *Section 6.4.1.2, "Restrict Special Files Dialog Box"*.

8. To complete the script creation process:

- click on 'OK' to complete the script and close the dialog box.
- click on 'Cancel' to close this dialog box and return to the previous screen.
- click on 'Apply' to save the script and keep the present dialog box open.

6.4. Script Menu

The Script menu allows you to:

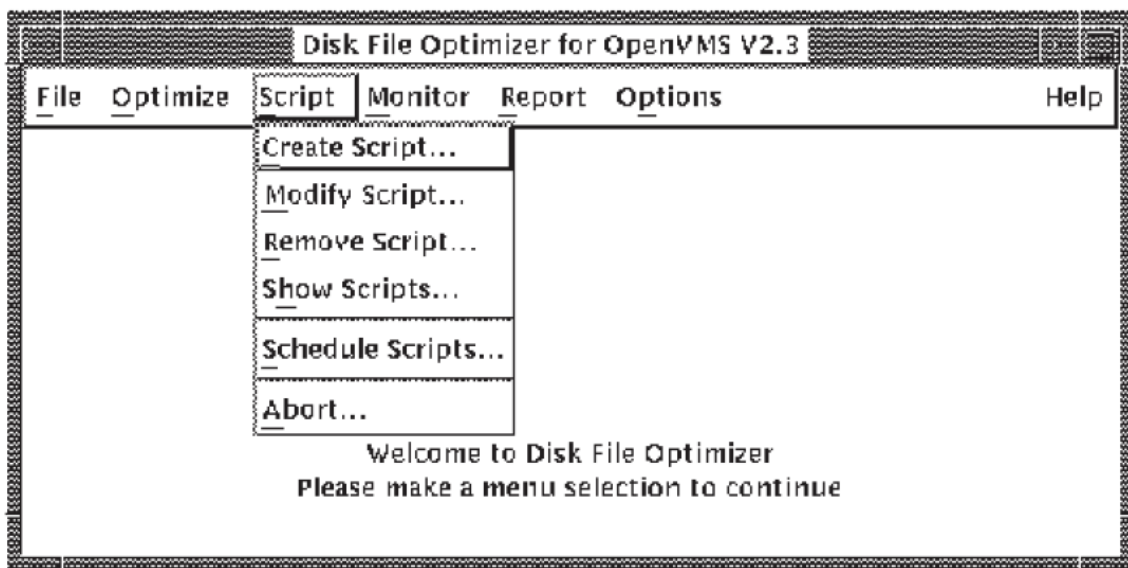
- create

- modify
- remove
- show
- schedule and
- abort

scripts for files or volumes.

Figure 6.4, "Script Menu" shows the Script pull-down menu.

Figure 6.4. Script Menu



6.4.1. Create Scripts

Use the following procedure to create a script for a volume or file.

1. Pull down the Script menu.
2. Select the create script option and designate the target you wish to create a script for: a Volume, Selected Files, or a Volume Index File.

Note

Note that you must define a script using the 'Script Define' buttons in the respective dialog box - see *Section 6.4.1.1, "Define a Script"*.

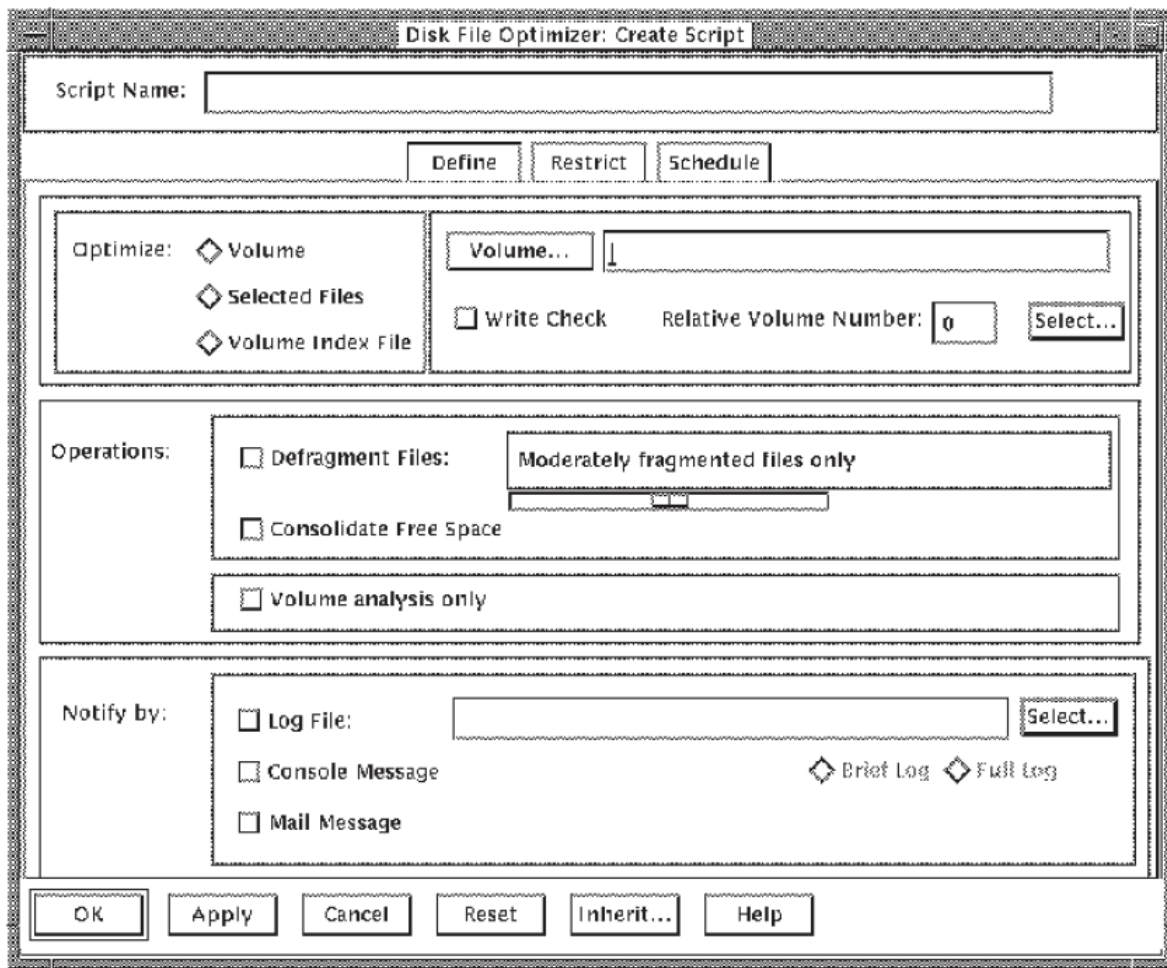
3. To specify a script name type the script name in the script field box. If you do not enter a script name, this instance of the Disk File Optimizer application will assign a default script name for you that resembles DFG\$SCRIPTxxx, where x is any hexadecimal character between 0 through F. This default script will be deleted upon successful completion of the defragmentation process.
 - Click on 'Reset' to restore the content of all fields to their defaults.

- Click on 'Inherit' to browse a script list and select the script you wish to inherit properties from.
4. Choose a Volume, Selected Files, or Volume Index File.
 - If you choose the Volume option, enter a specific Volume name on the Volume insert line or click on the volume list button to get a list of volumes to select from. Refer to *Figure 6.5, "Script Define Dialog Box for a Volume"*.
 - If you choose the Selected Files option, enter a specific file name on the file insert line or click on the file list button for a list of files to select from. Refer to *Figure 6.6, "Script Define Dialog Box for a File"*.
 - If you choose the Volume Index File option, enter a specific disk name or names on the Disk insert line or click on the disk list button for a list of disks to select from. Refer to *Figure 6.7, "Script Define Dialog Box for a Volume Index File"*.
 5. Fill out the rest of the Define dialog box according to the field description information in *Table 6.1, "Define Dialog Box Fields"*.
 6. Click on the Restrict button in this dialog box and fill in the information fields according to the information in *Section 6.4.1.2, "Restrict Special Files Dialog Box"*.
 7. Click on the Schedule button in this dialog box and fill in the schedule script information fields according to the procedure in *Section 6.4.1.2, "Restrict Special Files Dialog Box"*.
 8. To complete the script creation process:
 - click on 'OK' to complete the script and close the dialog box.
 - click on 'Cancel' to close this dialog box and return to the previous screen.
 - click on 'Apply' to save the script and keep the present dialog box open.

6.4.1.1. Define a Script

Before you can create a script or optimize a volume or file, you have to define a script using one of the Script Define dialog boxes for Volume, Selected Files, or Volume Index File.

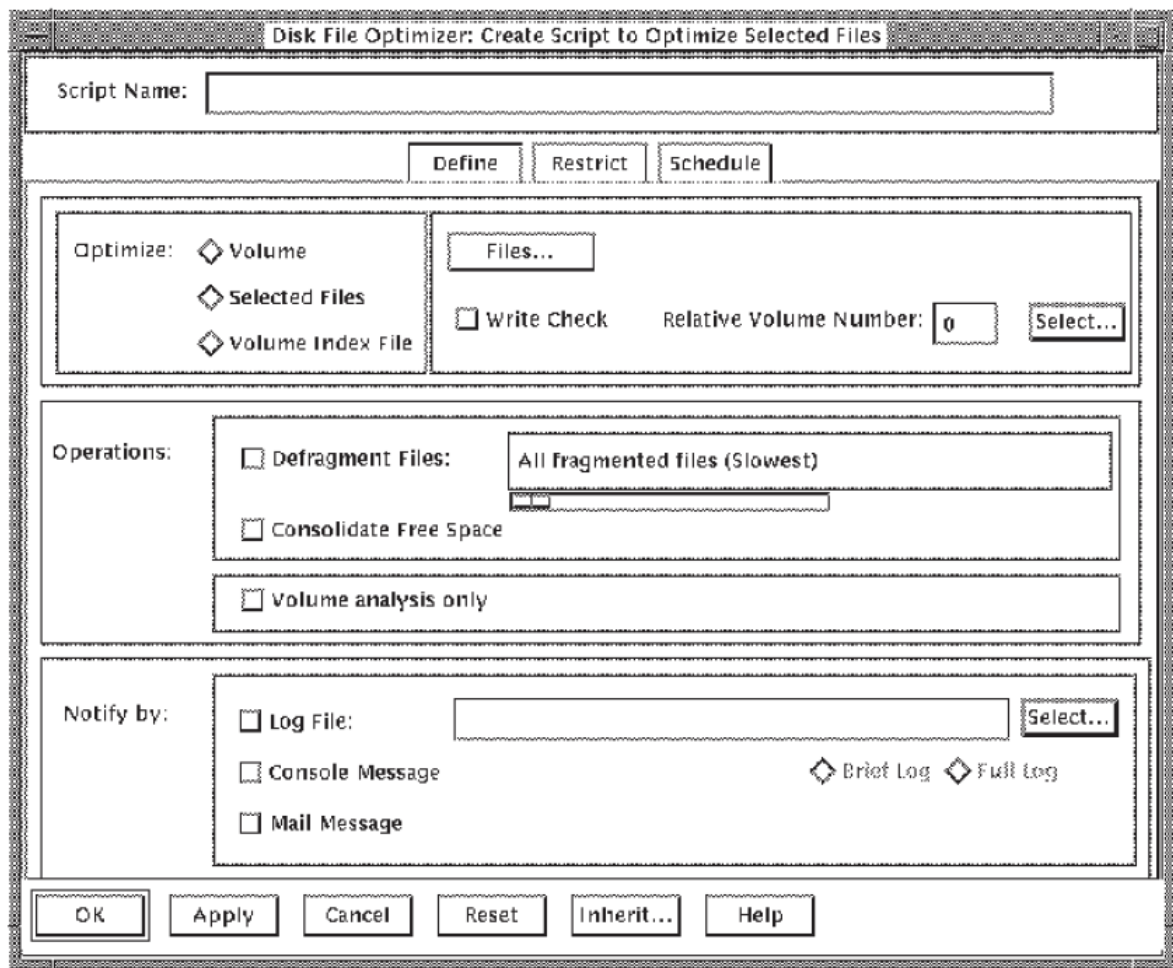
Figure 6.5, "Script Define Dialog Box for a Volume" shows the script define dialog box for a volume. Refer to *Table 6.1, "Define Dialog Box Fields"* for a description of the information fields.

Figure 6.5. Script Define Dialog Box for a Volume

In the Script Define dialog box for a volume, you can:

- Select a volume name.
- Select to enable or disable write check.
- Select the relative volume number of a volume set.
- Select whether to defragment files and what kind of files to defragment.
- Select whether to consolidate free space or not.
- Select whether or not to do a volume analysis.
- Select whether to notify you by log file or console message.
- Select whether to report brief or full log files.

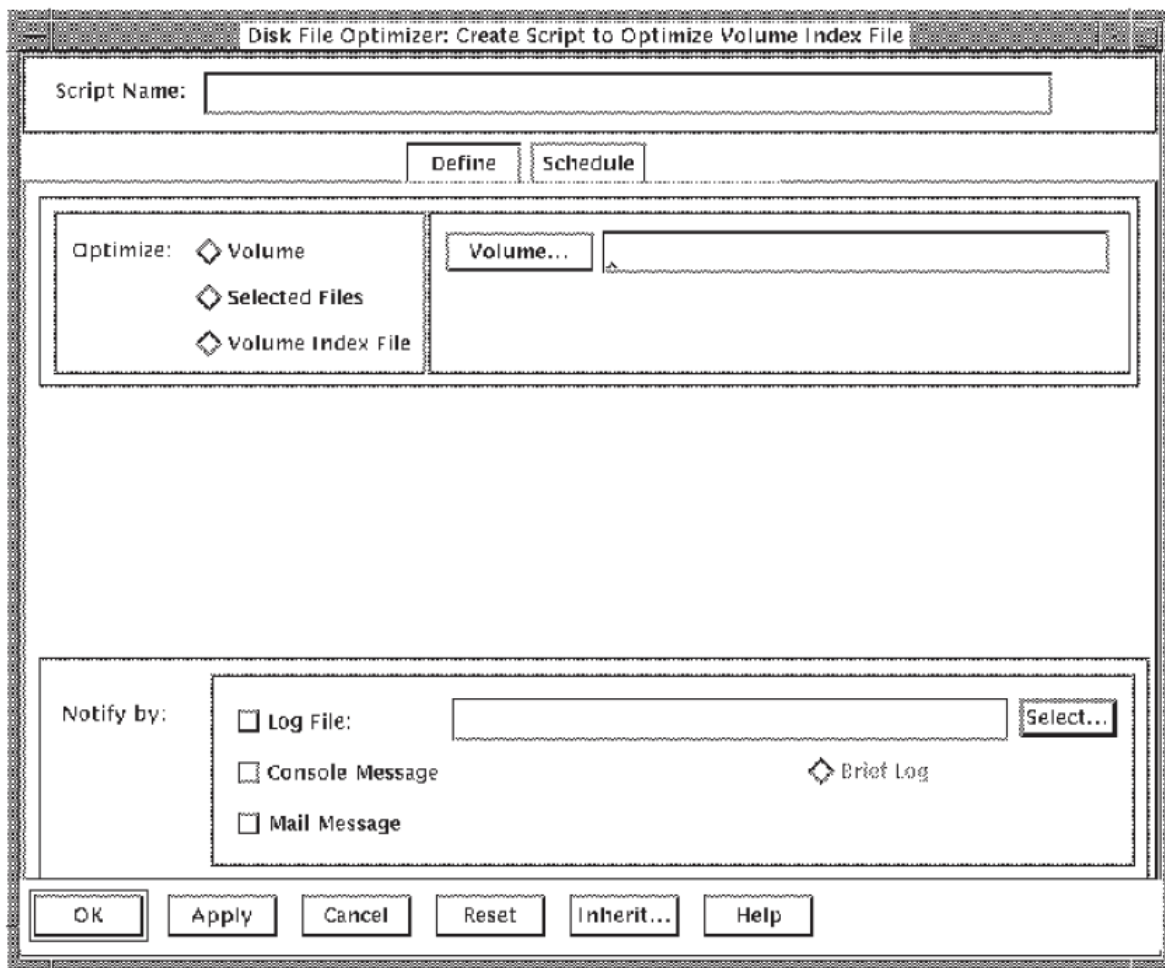
Figure 6.6, "Script Define Dialog Box for a File" shows the script define dialog box for a file.

Figure 6.6. Script Define Dialog Box for a File

In the Script Define dialog box for a file, you can:

- Select a file name.
- Select to enable or disable write check.
- Select the relative volume number of a volume set.
- Select whether to defragment files and what kind of files to defragment.
- Select whether to consolidate free space or not.
- Select whether or not to do a volume analysis.
- Select whether to notify you by log file or console message.
- Select whether to report brief or full log files.

Figure 6.7, "Script Define Dialog Box for a Volume Index File" shows the script define dialog box for a volume index file.

Figure 6.7. Script Define Dialog Box for a Volume Index File

In the Script Define dialog box for a volume index file, you can:

- Select a disk.
- Select whether to notify you by log file, console message, or mail message, or by all three.
- Select whether to report brief or full log files.
- Select whether to inherit the properties of another script.

Table 6.1. Define Dialog Box Fields

Field	Description
Volume	Defragments the volume named in the volume field. This has the same effect as using the DEFRAGMENT VOLUME DCL command.
Selected Files	Defragments the file or files named in the files field. This has the same effect as using the DEFRAGMENT FILE DCL command.
Volume Index Files	Defragments the INDEXF.SYS of the disk named in the disk field. This has the same effect as using the DEFRAGMENT OFFLINE_VOLUME DCL command.

Field	Description
Volume...	The Volume... button only appears when the Volume or Volume Index File buttons are depressed. Enter the volume name or use the Volume... button to bring up a list of volumes or disks.
Files...	The Files... button only appears when the Selected Files button is depressed. Push the File... button file to bring up a list of files.
Write Check	The Write Check option requests that a read-after-write check be performed on all defragmented or placed files. This qualifier significantly extends the execution time of a defragmentation process. This has the same effect as the /WRITE_CHECK DCL qualifier.
Relative Volume Number	Allows you to select the relative volume number of a volume set. This has the same effect as the /RVN= DCL qualifier.
Operations	You may elect to defragment specific files, consolidate disk space, or do an analysis check only. File defragmentation and space consolidation can be done together. Defragment Files: Allows you to select which files to defragment. Use the three-position sliding bar: ● LEFT - Files that are not 100% contiguous. ● MIDDLE - Files whose extent cannot be completely mapped. ● RIGHT - Files with retrieval points in more than one file header. Consolidated Freespace: Consolidate the free space on the volume. Volume analysis only: Requests a volume analysis only, no defragmentation occurs. The results are available in the log file. The operations has the same effect as the /LEVEL=x DCL qualifiers as shown in <i>Table 6.2, "Translations of Operations to DCL Command/LEVEL=x"</i>.
Log File	The Log File option causes the defragmentation process to create a file containing a list of all its operations. The information in the log file varies with the presence of the Full or Brief option selections: ● The Brief option lists only the statistics of each phase, such as number of files analyzed and number of files defragmented. ● The Full option lists all files analyzed by the defragmentation process and reports their disposition during each phase of defragmentation. This has the same effect as the /LOG=log_file and /FULL or /BRIEF DCL qualifier.
Console Message	The console message option requests that brief status and termination messages regarding defragmentation processes be sent to OPCOM. This has the same effect as the /OPCOM_NOTIFICATION DCL qualifier.

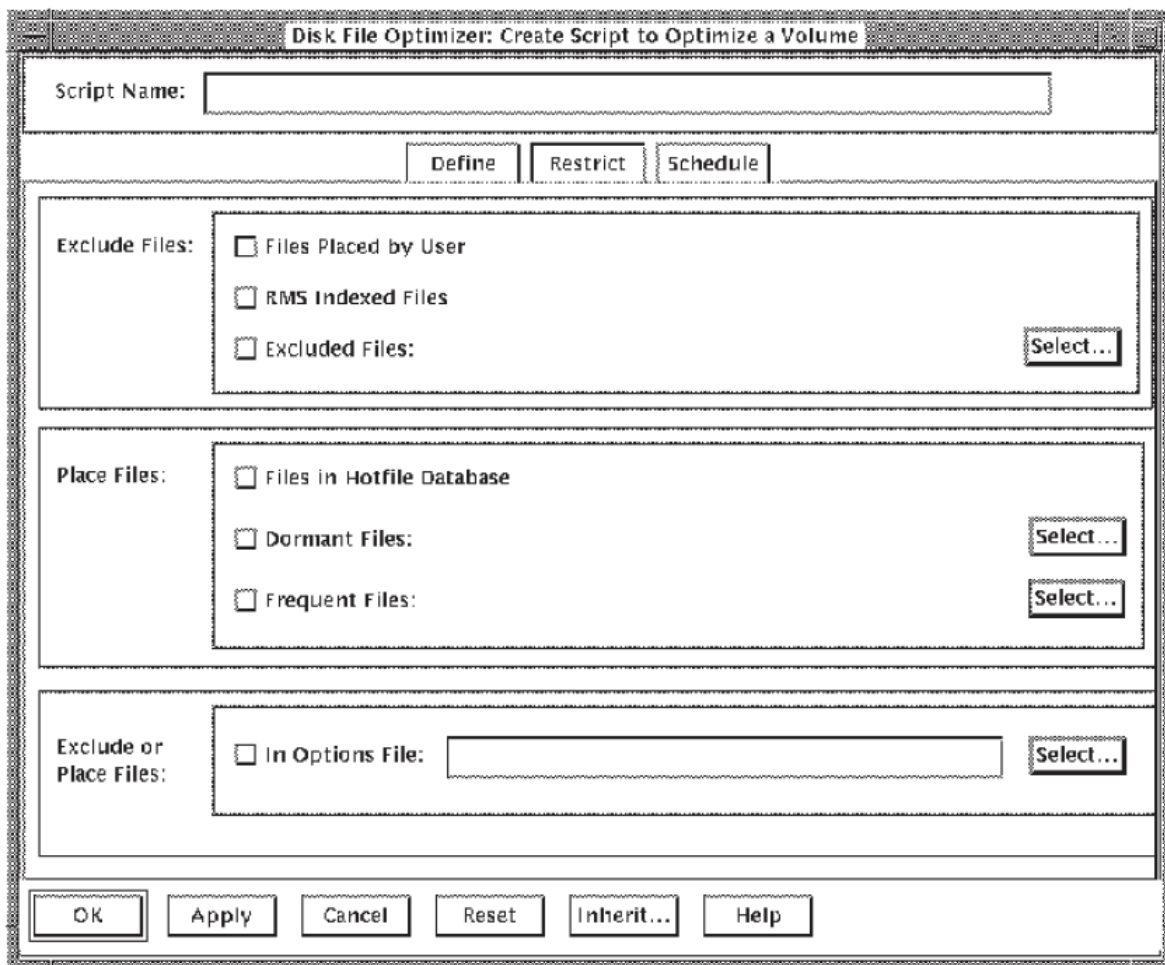
Table 6.2. Translations of Operations to DCL Command/LEVEL=x

Defragment Files	Sliding Bar	Consolidate Free Space	Volume Analysis Only	/LEVEL=
Selected	LEFT			1
Selected	MIDDLE			2
Selected	RIGHT			3
Not Selected		Selected		4
Not Selected		Not Selected	Selected	5

6.4.1.2. Restrict Special Files Dialog Box

The Restrict Special Files dialog box allows you to include or exclude any special files during the defragmentation process.

Figure 6.8, "Restrict Special File Dialog Box" shows the Restrict Special Files dialog box for a volume and Table 6.3, "Restrict File of Dialog Box Fields" describes the information fields.

Figure 6.8. Restrict Special File Dialog Box

You can exclude files such as:

- Files placed by the User
- RMS Indexed Files
- Excluded Files

You can also exclude the following types of Place Files:

- Files in the hotfile database
- Dormant files
- Frequent files

Table 6.3. Restrict File of Dialog Box Fields

Field	Description
Files Placed by User	Select this button if you wish to ignore explicitly placed files. This has the same effect as the <code>/IGNORE=(PLACED)</code> DCL qualifier.
RMS Indexed Files	Select this button if you wish to ignore RMS indexed files.
Excluded Files	The Exclude Files option specifies files to be excluded from processing. You may enter a file name in the file field or use the file list button to bring up a list of files to select from. The defragmentation process ignores files matching the listed file specifications. Any file indicated in the EXCLUDE list and in either the FREQUENT or DORMANT lists is excluded from processing. The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is <code>DISK:[000000...]*.*.*</code>. Specifying <code>*.TMP;*</code> selects all <code>*.TMP;*</code> files on the entire volume. The defragmentation process always excludes files marked NOMOVE, open files, files excluded in an options file, and files excluded using the Ignore option.
Files in Hotfile Database	This option causes the defragmenter to query the appropriate hotfile database and append the list of hotfiles to the defragmenter's frequent file list.
Dormant Files	The Dormant Files option indicates files to be considered dormant when final file placement decisions are made. You may enter the file name or use the file list button to bring up a list of files to choose from. Dormant files are placed on the volume to allow placing other files in optimum disk locations. In general, they are placed further away from the ODS-2 data structures than Frequent files and files not classified in any manner. If a file is indicated in both the DORMANT and FREQUENT lists, it is considered dormant. If a file is indicated in both the EXCLUDE and DORMANT lists, it is excluded.

Field	Description
	Note The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described. The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*;*. Specifying *.TMP;* selects all *.TMP;* files on the entire volume.
Frequent Files	The Frequent Files option indicates files to be considered frequently accessed when final file placement decisions are made. You may enter the file name or use the file list button to bring up a list of files to choose from. Frequent files are placed to optimize access speed. If a file is indicated in both the FREQUENT and DORMANT lists, it is considered dormant. If a file is indicated in both the FREQUENT and EXCLUDE lists, it is excluded. Note The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described. The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*;*. Specifying *.TMP;* selects all *.TMP;* files on the entire volume.
In Options File	The Options File option provides an alternate method for associating files with the Dormant File, Exclude File, or Frequent File options. You may enter the file name or use the file list button to bring up a list of files to choose from. If the list of files for any of these options is long, you can create a separate file for this information. Each record of the options file consists of a file name followed by one of these keywords: ● /EXCLUDE—File to be excluded ● /FREQUENT—File to be considered frequently accessed ● /DORMANT—File to be considered dormant

Field	Description
	The following example shows the contents of an options file called MYOPTFILE.OPT. This file specifies that all files on the target device called FILE1.EXE are frequent files, all files in the [ARCHIVE] directory are to be excluded, and all files in the [OLDSTUFF] directory and its subdirectories are dormant. <pre>FILE1.EXE/FREQUENT [ARCHIVE] *.*;*/EXCLUDE [OLDSTUFF...] *.*;*/DORMANT</pre>

6.4.1.3. Schedule a Script Dialog Box

The Schedule Definition dialog box allows you choose many different script or defragmentation schedule parameters.

Figure 6.9, "Schedule Definition" shows the Schedule Definition dialog box and Table 6.4, "Schedule Definition Dialog Box Fields" describes the dialog box information fields.

Figure 6.9. Schedule Definition

Disk File Optimizer: Create Script

Script Name:

Define **Restrict** **Schedule**

Schedule:

- ☒ Start After This Time:
- ☒ Wait for START Command
- All scheduling is: ☒ Enabled ☐ Disabled

Time Limit:

Time Limit:

Default Priority:

Minimum Priority:

Reschedule:

- ☒ Don't reschedule
- ☒ At This Interval: [dddd-hh:mm:ss.cc]
- ☒ Continuously
- ☒ To Follow Script:

☐ Schedule on Nodes:

☐ Prologue File (pre-processing)

☐ Epilogue File (post-processing)

The parameters in the dialog box are:

- Start after this time

- Wait for start time
- Don't reschedule
- Reschedule at this interval
- Reschedule continuously
- Reschedule to follow script
- Schedule on nodes:
- Time limit
- Execute a prologue command file before running the script
- Execute an epilogue command file after running the script

Table 6.4. Schedule Definition Dialog Box Fields

Field	Description
Script Name	Identifies the name of the script to be scheduled.
Start After This Time	Starts execution of the script after the specified time. This has the same effect as the DCL qualifier <i>/AFTER=</i> .
Wait for Start Command	Keeps the specified script from executing until started by the Schedule Scripts option in the Script pull down menu. This has the same effect as the DCL qualifier <i>/NOAFTER</i> .
Priority	This is the priority that the OpenVMS scheduler uses for the defragment volume operation. The default priority is set at 4. The minimum priority is set at 3. This has the same effect as the DCL qualifier <i>/PRIORITY=(DEFAULT=n, MINIMUM=n)</i> .
Time Limit	Specifies the delta time allowed for a given defragmentation operation. This has the same effect as the DCL qualifier <i>/TIME_LIMIT=</i> .
Do not Reschedule	The DFO schedule will not reschedule this script.
At This Interval	Specifies the delta time of the interval at which the script will be executed. This has the same effect as the DCL qualifier <i>/INTERVAL=</i> .
Continuously	Causes the script to run continuously. When selected, the defragmentation script used for the process must be aborted or canceled when no longer needed. This has the same effect as the DCL qualifier <i>/CONTINUOUSLY</i> .
To Follow Script:	Names the script that will run before the current script is executed. The script must exist in the database. Enter a script name or click on the script list button for the list of scripts to choose from. This has the same effect as the DCL qualifier <i>/PREDECESSOR=</i> .
Schedule on Nodes:	Limits the defragmentation to only those nodes listed. Enter a node, or nodes, or click on the node list button for a list of nodes to choose from. This has the same effect as the DCL qualifier <i>/NODE=</i> .
Epilogue File	The Epilogue File option identifies a DCL command file that DFO executes upon completion of the defragmentation operation. This has the same effect as the <i>/EPILOGUE=</i> DCL qualifier.

Field	Description
Prologue File	The Prologue File option identifies a DCL command file that DFO executes just before the defragmentation operation starts. This has the same effect as the /PROLOGUE= DCL qualifier.

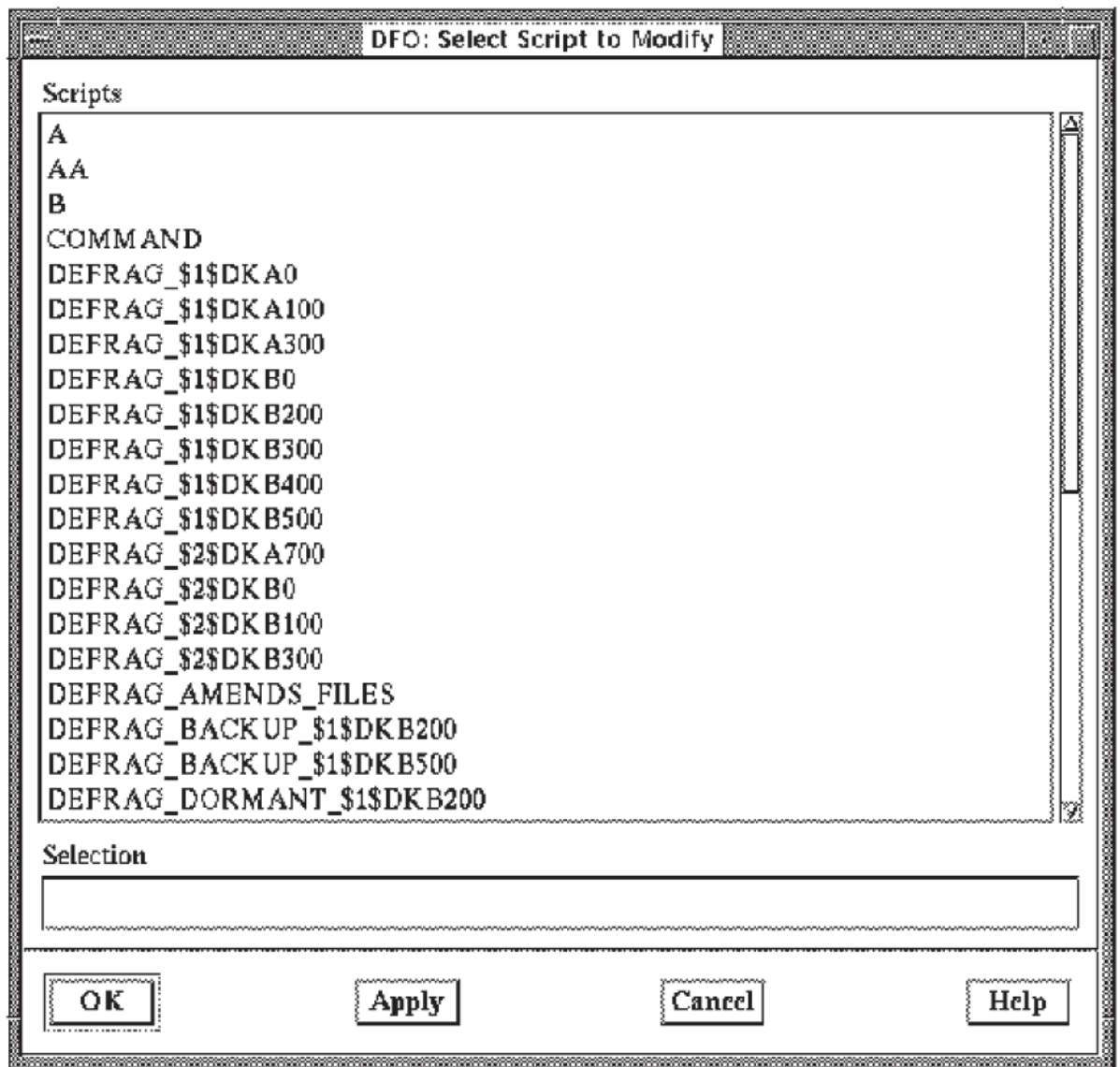
6.4.2. Modify a Script

This section describes how to modify existing scripts.

Use the following procedure to modify a script:

1. Pull down the Script menu.
2. Select the Modify Script option and a Script List selection box appears. See *Figure 6.10, "The Modify Script Dialog Box"*

Figure 6.10. The Modify Script Dialog Box



3. Select a script from the script list selection box.

4. Select either the Apply or the OK button to bring up the Modify Script dialog box.
5. From this point on, follow the same procedure as if you were creating a script. These procedures are described in the *Section 6.4.1.1, "Define a Script"*, *Section 6.4.1.2, "Restrict Special Files Dialog Box"*, and *Section 6.4.1.3, "Schedule a Script Dialog Box"*.

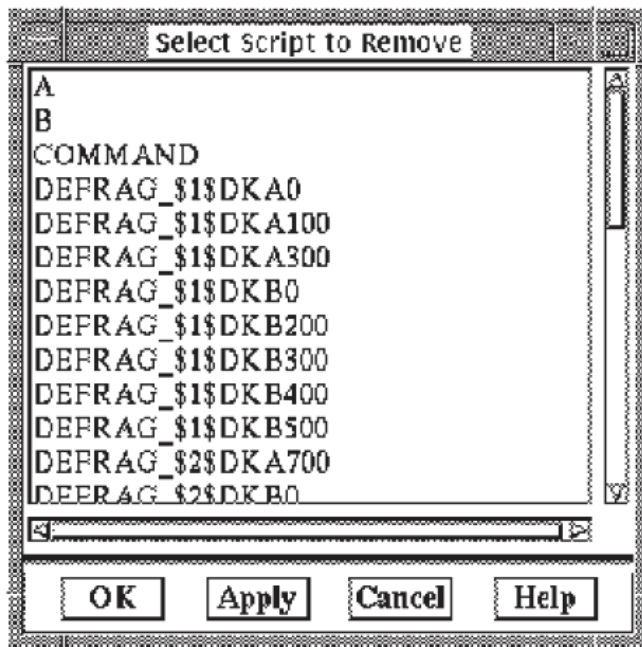
6.4.3. Remove a Script

This option allows you to remove scripts from the database.

Use the following procedure to remove a script:

1. Select the Remove Script option of the Script pull-down menu.

Figure 6.11. The Remove Script Dialog Box



2. Select the scripts from the Script List see *Figure 6.11, "The Remove Script Dialog Box"*.

Point to and click on the script names in the list to mark that you wish to remove them.

3. To complete the remove, click on:
 - OK to remove the selected script(s) and exit the Remove Script option.
 - Cancel to leave the Remove Script option without taking any action.
 - Apply to remove the selected script(s) and remain in the Remove Script option.

6.4.4. Show a Script

The Show dialog box provides three ways of sorting the scripts you would like to show:

- Scripts that apply to specific nodes, either exclusively or shared by other nodes

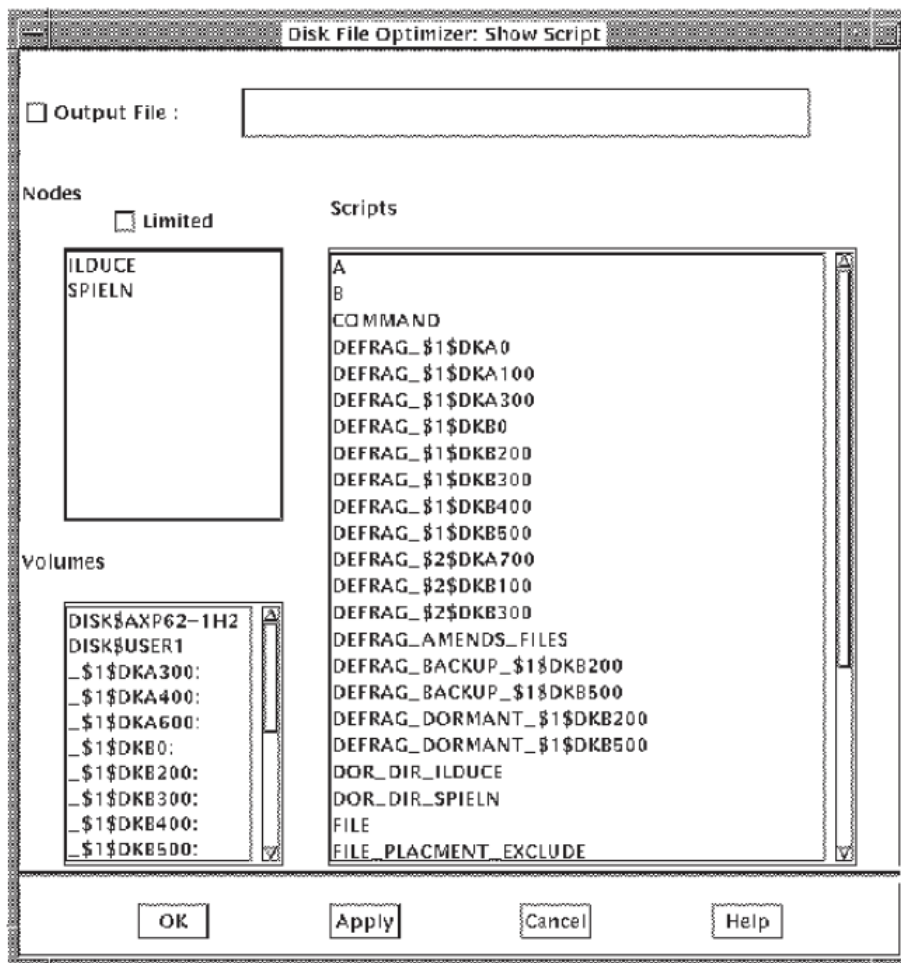
- Scripts that apply to specific disk volumes
- All scripts in the database

You also have the option of creating an output file for selected scripts.

Use the following procedure to show scripts:

1. Select the Show Scripts option of the Script menu.

Figure 6.12. Show Script Dialog Box



2. Click twice on a script name in the 'Scripts' box to select it.
3. Choose the kind of output you wish for the selected script:

IF you keep the Output File option, enter the name of the output file you want to create.

IF you do not select the output file option, the script file is visible only while using this option.

4. If you want to show scripts sorted by volume, proceed with Step 4, otherwise, continue to show scripts by node.

To 'limit' the scripts display to those belonging to the selected node leave the 'Limited' option selected (this is the default).

IF you select the Limited option, you limit the candidate list of scripts to those that apply *only* to the selected node. **IF** you do not wish to use the 'Limited' option, click on the option box to deselect it. The candidate list of scripts will then include every script that runs on all nodes.

Select a script in the Node box, and note that the script name gets highlighted in the Scripts box.

Proceed with Step 6.

5. To show the scripts that apply to a volume, select the volume. Scripts that apply to the volume are listed in the Scripts box.

Select a script that applies to a Volume, and note the script name gets highlighted in the Scripts box.

6. To see a script, double click on the script name in the Scripts field or press Return, or click on the Apply button. The contents of the script appear in the area below the script name.

This feature allows you to see more than one script at a time. To see additional scripts, select their names.

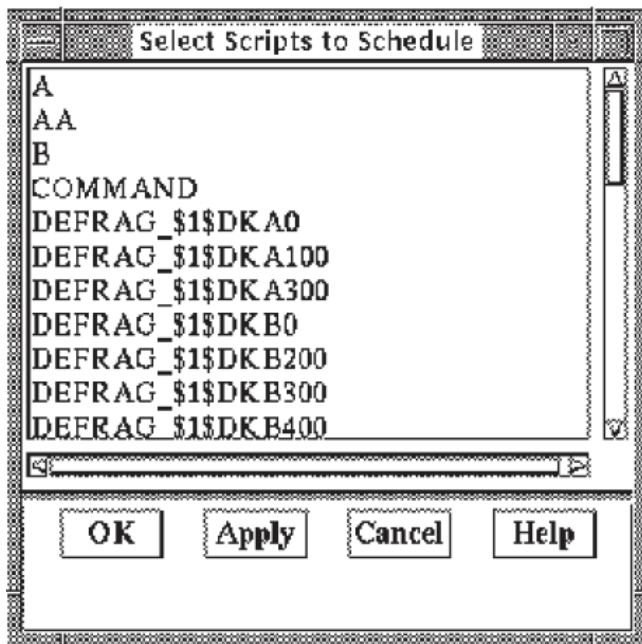
7. Finish the show script operation by clicking on:
 - OK to produce the output file named in the Output File field
 - Apply to leave the Show Script dialog box and optionally produce the output file named in the Output File field
 - Cancel to quit and not produce any further output

6.4.5. Schedule a Script

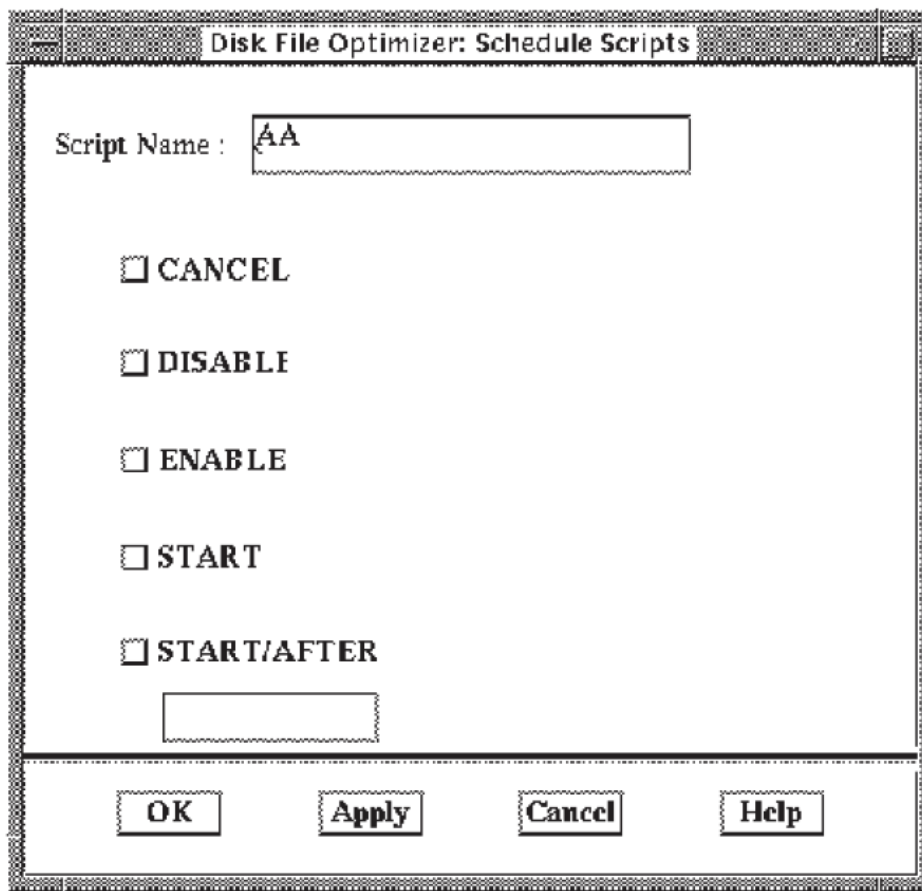
This section describes how to schedule a defragmentation script.

Use the following procedure to schedule scripts and manage script schedules:

1. Select the Script pull-down menu.
2. Select the Schedule Script option of the Script pull-down menu for the script list box. See *Figure 6.13, "Script List Dialog Box"*.

Figure 6.13. Script List Dialog Box

3. Select all the scripts you want for scheduling actions from the Script list and click on either the OK or the Apply button. This will bring up the Schedule Scripts dialog box. Refer to *Figure 6.14*, "Schedule Script".

Figure 6.14. Schedule Script

4. Take any of the actions on the script outlined in *Table 6.5, "Script Scheduling Actions"*.
5. To finish the scheduling action click on:
 - OK button to apply the modifications and leave the dialog box.
 - Cancel button to quit and not take any scheduling actions on the named script.
 - Apply button to apply the scheduling options selected. You may take action on another script by entering the name of the script in the Script Name field.

Table 6.5. Script Scheduling Actions

Option	Description
Script Name	Name of script for scheduling action.
Cancel	Cancels the next scheduled execution of the specified script. Normal scheduling resumes thereafter.
Disable	This option disables the script from execution.
Enable	This option enables a previously disabled script. Once enabled, the script operates normally.
Start	Starts execution of the script identified.

Option	Description
Start/After	Starts execution of the script when the specified time is reached. Enter the time in the field below the option.

6.4.6. Abort a Script

Use this option to stop a script that is currently executing.

Use the following procedure to abort an active defragmentation script:

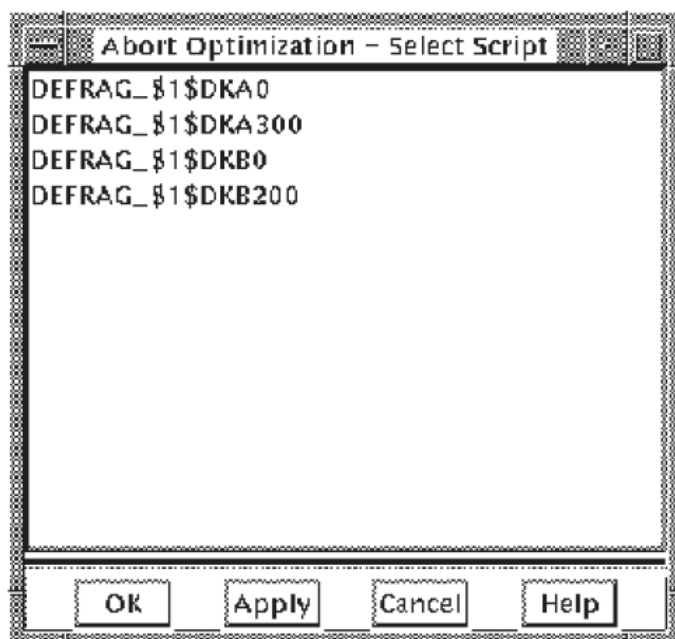
1. Select the Script pull-down menu.
2. Select the Abort option.
3. Select the script to abort from the Script List box. Use caution because a double click on a script entry will abort that script.

Point to any one or more of the scripts listed, clicking once on each one you want to abort.

4. Finish the abort action by selecting:
 - OK to abort the selected scripts and exit the Abort Script option
 - Apply to abort the selected scripts and remain in the Abort Script option
 - Cancel to leave the Abort Script option without taking any action

Figure 6.15, "The Abort Dialog Box" shows a list of scripts in the Abort dialog box.

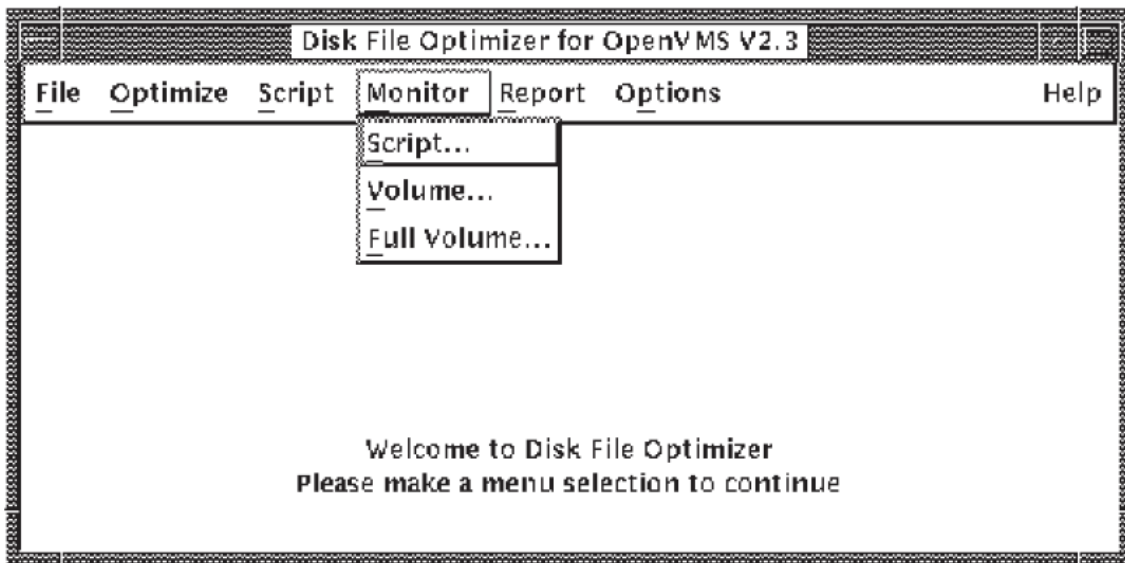
Figure 6.15. The Abort Dialog Box



6.5. Monitor a Script or Volume

The Monitor menu allows you to select a script or volume for monitoring the defragmentation process. You can view how the space is allocated on the full volume or on a partial volume in a sliding window.

Figure 6.16. Monitor Menu



When selecting the Monitor option, DFO displays two windows:

- A process statistics window showing process information for the defragmentation process. Refer to *Figure 6.18, "Defragmentation Process Statistics Window"* and *Table 6.6, "Defragmentation Process Statistics Window Description"*.
- A monitor map window showing free space and allocated space on the volume. Refer to *Figure 6.19, "Defragmentation Full Volume Map Window"* and *Figure 6.20, "Defragmentation Partial Volume Map Window"*.

This procedure describes how to monitor allocated space on a disk volume.

1. Select the Monitor pull-down menu.
2. Select the means by which you want to monitor allocated space. Select the Script, Volume, or Full Volume option of the Monitor pull-down menu. Refer to *Figure 6.16, "Monitor Menu"*.
 - **IF** you want to identify the volume by a script, select the Script option, then select the script name from the Script List. If you choose to monitor the volume by a script, a Defragmentation Process Statistic Window and a Defragmentation Full Volume Map Window will appear. Refer to *Figure 6.18, "Defragmentation Process Statistics Window"*, *Figure 6.19, "Defragmentation Full Volume Map Window"* and *Table 6.6, "Defragmentation Process Statistics Window Description"*.
 - **IF** you want to identify the volume by name, select the Volume or Full Volume option, then select the volume name from the Volume List.

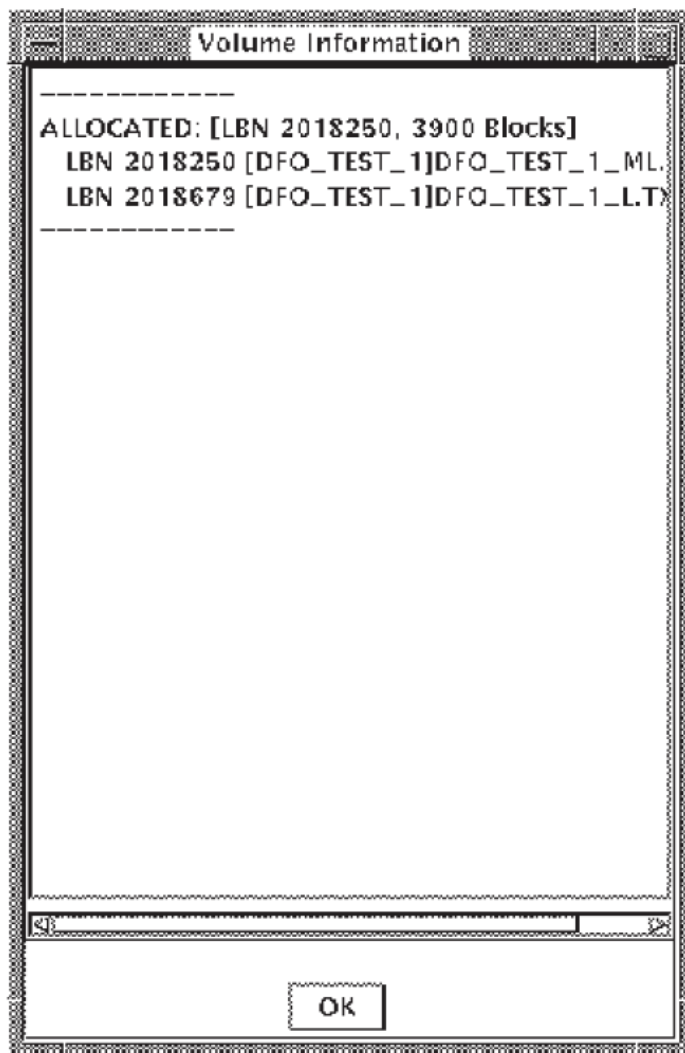
The volume options allows you to view different parts of the entire volume map, use the arrows and scroll bars around the map on the defragmentation volume map window. If you choose to monitor the Volume, a Defragmentation Process Statistic Window and a Defragmentation Partial Volume Map Window will appear. Refer to *Figure 6.18, "Defragmentation Process Statistics Window"*, *Figure 6.20, "Defragmentation Partial Volume Map Window"* and *Table 6.7, "Defragmentation Volume Map Window Description"*.

To view the full volume, select the Full Volume option, then select the volume name from the Volume List. If you choose to monitor the Full Volume, a Defragmentation Process Statistic Window and a Defragmentation Full Volume Map Window will appear. Refer to *Figure 6.18, "Defragmentation Process Statistics Window"*, *Figure 6.19, "Defragmentation Full Volume Map Window"* and *Table 6.6, "Defragmentation Process Statistics Window Description"*.

3. To view more detailed information about the allocated space shown on the Defragmentation Volume Map, press mouse button 1 and drag the cursor over the area of the map you want to examine. A pop-up window identifies the files taking space on that area of the volume.

Figure 6.17, "Volume Map File Information" is an example of the volume map file information.

Figure 6.17. Volume Map File Information



6.5.1. Defragmentation Process Statistics Window

The Defragmentation Process Statistics Window shows the process information and statistics on how files and space on the volume are allocated during defragmentation.

Figure 6.18, "Defragmentation Process Statistics Window" shows the Defragmentation Process Statistics window, and *Table 6.6, "Defragmentation Process Statistics Window Description"* describes the data field information.

Figure 6.18. Defragmentation Process Statistics Window

Disk File Optimizer: Process Statistics			
Process Id:	328002A7	Process:	DFG\$DFO_TEST
Date:	21-FEB-1997 13:32:53.71		
Script Name:	DEFrag_ \$1\$DKB300		
Operating On:	SPIELN	Current Status:	%NORMAL
Phase:	File evaluation	Elapsed Time:	0 00:00:47.25
Buffered I/O	Direct I/O	CPU Utilization	Page Fault
Count: 4054	Count: 6474	Time: 000:00:00:11.59	Count: 968
Rate: 39/second	Rate: 57/second	Usage: 5%	Rate: 0/second
Files Evaluated :	521	Dormant Count:	0
Candidate Count:	521	Exclude Count:	0
Placed Files:	0	Frequent Count:	0
Indexed Files:	0		
Files Partly Defragmented:	0	Dormant Files Placed:	0
Files Defragmented:	0	Frequent Files Placed:	0
Files Processed:	0	Free Space Extent Count:	55067
		Free Space Extent Countdown:	0
Current File			
Exit			

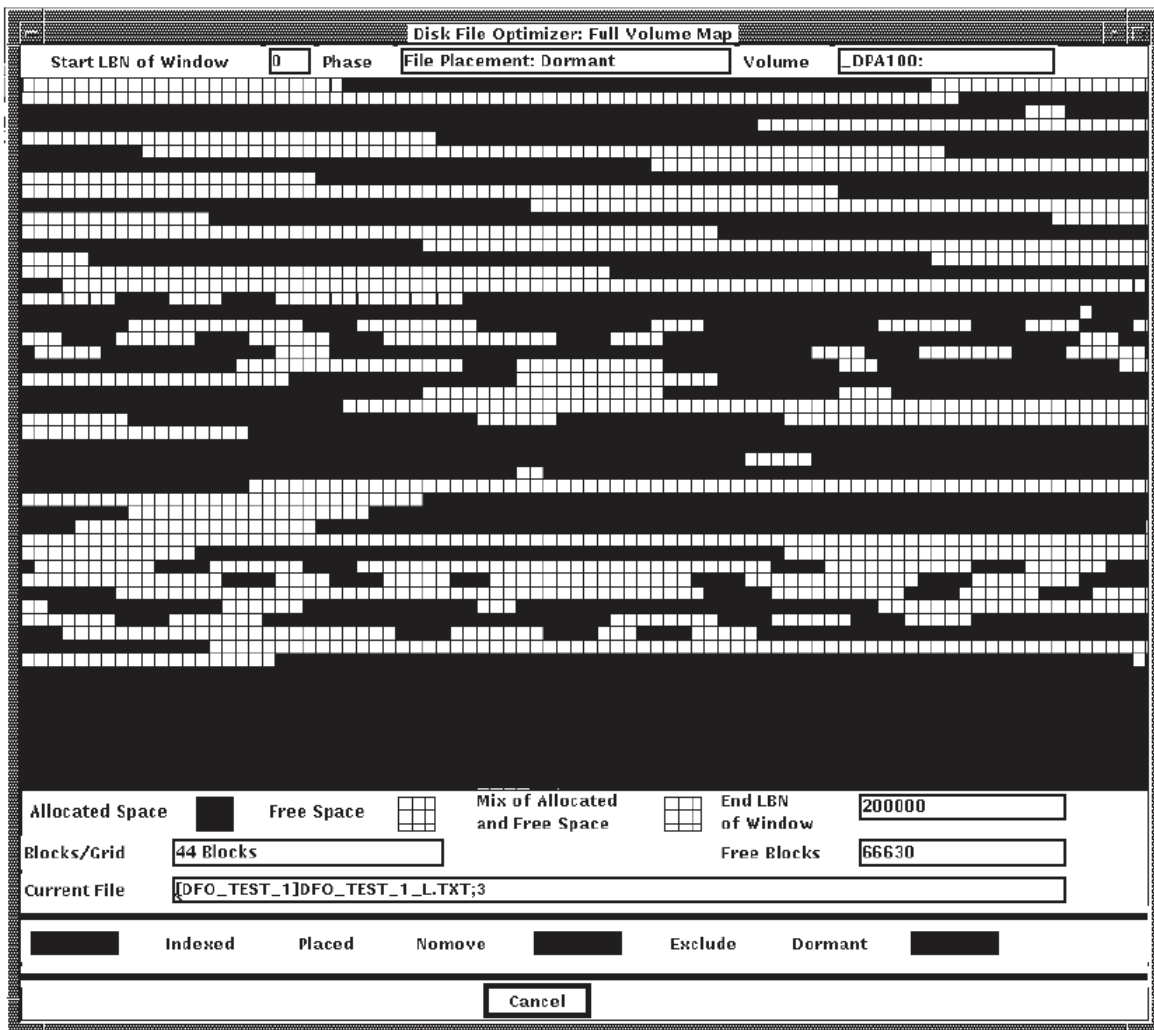
Table 6.6. Defragmentation Process Statistics Window Description

Field	Description
Process Id:	The defragmentation process ID number.
Process:	The defragmentation process name.
Date:	The current date and time.
Script Name:	The name of the script being monitored. (The field applies to the Defrag PLUS variant only.)
Operating On:	The node on which the defragmentation process is operating.
Current Status:	The status of the defragmentation process.
Phase:	The current defragmentation process phase.
Elapsed Time:	The elapsed time since the start of the defragmentation process.
Buffered I/O Count: and Rate:	The buffered I/O count and rate of the defragmentation process.
Direct I/O Count: and Rate:	The direct I/O count and rate of the defragmentation process.

Field	Description
CPU Utilization Time: and Usage	The CPU utilization (time and usage) of the defragmentation process.
Page Fault Count: and Rate:	The page fault count and rate of the defragmentation process.
Files Evaluated:	Number of files evaluated.
Dormant Count:	Number of dormant files.
Candidate Count:	Number of candidate files for defragmentation.
Exclude Count:	Number of excluded files.
Placed Files:	Number of placed files.
Frequent Count:	Number of frequent files.
Indexed Files:	Number of indexed files.
Files Partly Defragmented:	Number of files partly defragmented.
Dormant Files Placed:	Number of dormant files placed.
Files Defragmented:	Number of files defragmented.
Frequent Files Placed:	Number of frequent files placed.
Files Processed:	Number of files processed.
Free Space Extent Count:	Number of extents to be consolidated into free space.
Free Space Extent Countdown:	Number of extents left to be consolidated into free space.
Current File:	Current file being evaluated.

6.5.2. Full Volume Window

The defragmentation full volume map window allows you to see how the storage space and free space on the full volume are allocated. *Figure 6.19, "Defragmentation Full Volume Map Window"* shows the full volume map window and *Table 6.7, "Defragmentation Volume Map Window Description"* describes the window contents and how it functions.

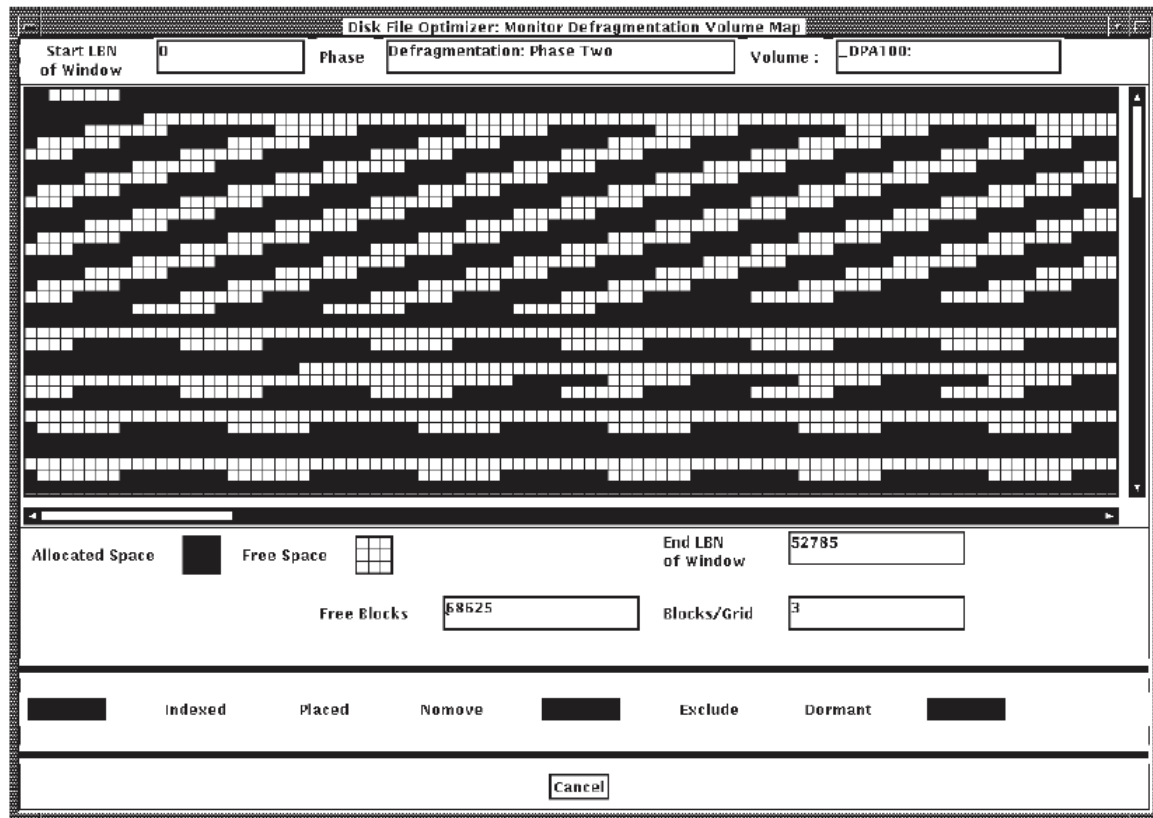
Figure 6.19. Defragmentation Full Volume Map Window**Table 6.7. Defragmentation Volume Map Window Description**

Field	Description
Start LBN of Window	The starting LBN at the upper left corner of the viewport.
Phase	The current phase of the defragmentation process.
Volume	The name of the volume that is being defragmented.
Allocated	Space Shows the color of allocated space on the disk volume.
Free Space	Shows the color of free space on the disk volume.
Mix of Allocated and Free Space	Shows the color of space that is partially allocated and free space. This entry is only on the full volume map.
End LBN of Window	The ending LBN at the lower right corner of the viewport.
Free Blocks	The number of free blocks on the disk volume.
Blocks/Grid	The number of blocks in each grid on the view port.
Current File	The current file the defragmentation process is working on. This entry is only on the full volume map.

6.5.3. Partial Volume Window

The partial volume map window provides a closeup detailed view of how the storage space is allocated on a smaller segment of the whole volume. *Table 6.7, "Defragmentation Volume Map Window Description"* describes the window contents and how it functions.

Figure 6.20. Defragmentation Partial Volume Map Window



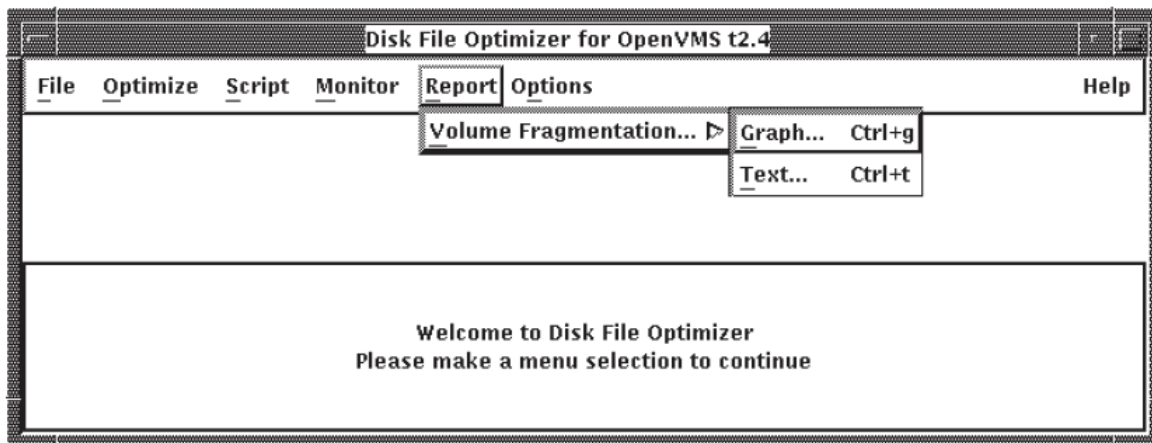
When a Defragmentation Full or Partial Volume Map Window is displayed on the screen, the File Type color bar at the bottom of the dialog helps you identify the different file types on the volume map.

The file type color bar shows the different file types on the volume map in different colors if you have used these file categories in the defragmentation process. The following list shows the color categories for the various files types.

Directories	GREEN
Dormant Files	YELLOW
Exclude Files	ORANGE
Frequent Files	BLUE
General Files	BLACK

6.6. Report on the State of a Volume

The report menu allows you to review the current state of the volumes to determine if defragmentation is needed. Click on the Report menu for the two options it supports. See *Figure 6.21, "Report Menu"*. You can opt for a graphic or a text report of the state of the volumes.

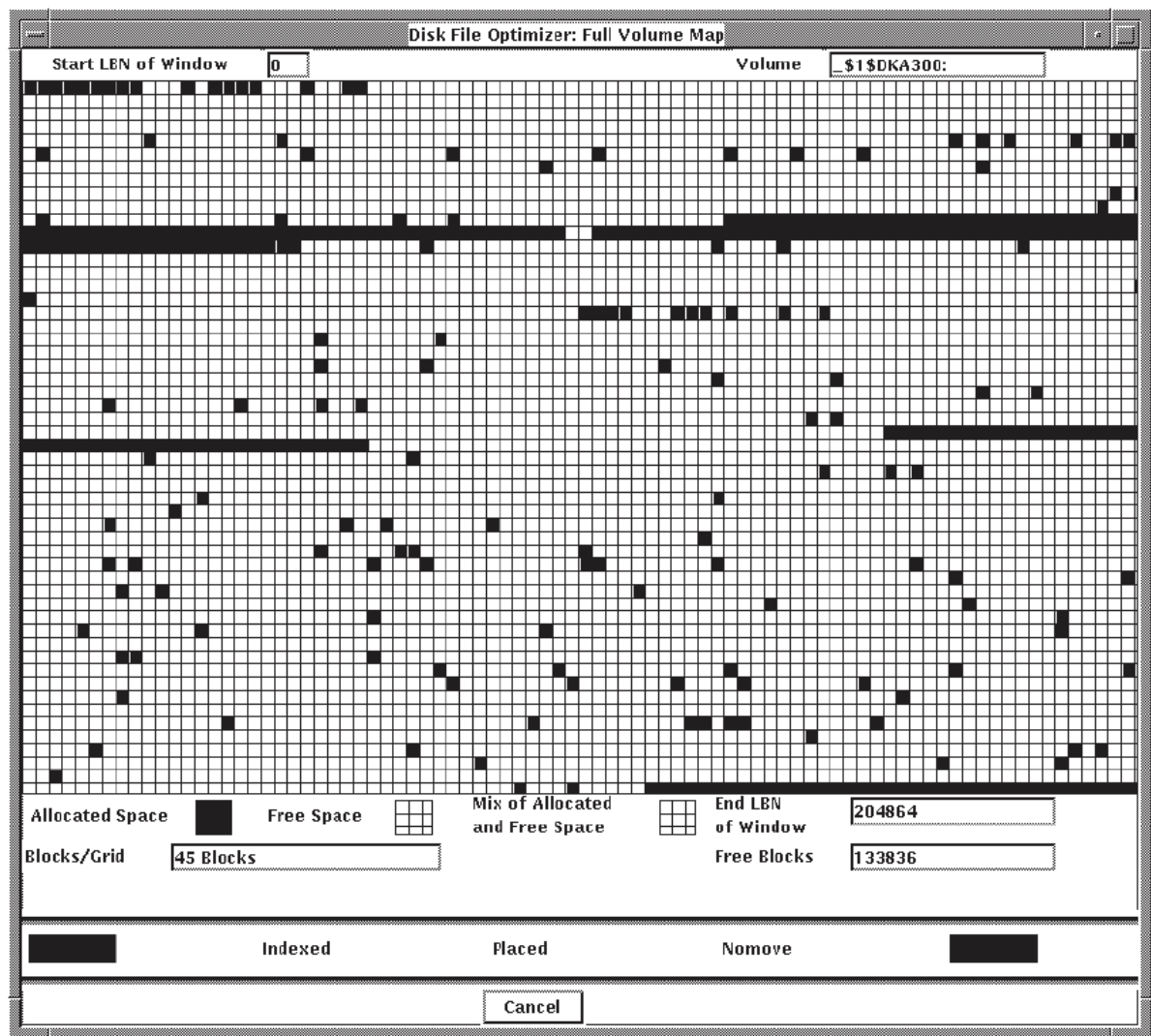
Figure 6.21. Report Menu

6.6.1. Graphics Format

Use the following procedure for a graphics display of the selected volume(s).

1. Click on the Graph... suboption of the Report-Volume Fragmentation menu. See *Figure 6.21, "Report Menu"*. The Report - Select Volume dialog appears. See *Figure 5.17, "The Report Volume Selection Dialog Box"* and *Table 6.8, "Report Volume Option and Field Descriptions"*.
2. Note that the first entry in the list is marked selected by default. You can select any other volume for the respective report. To select another volume, move mouse pointer to it and click.
3. Click on OK.

A graphical report of the state of the selected volume(s) appears in the dialog. See *Figure 6.22, "Report Graphics Full Volume Map"*. Click on Cancel to close this image dialog.

Figure 6.22. Report Graphics Full Volume Map

This dialog is nothing but a color coded window of how the volume is shared by its files and directories. The first line in this dialog shows the Start LBN and Volume id - both of which are self explanatory.

The following may require explanation:

- The multicolored grid below the first line is actually a map of the volume's contents.
- Three gray levels mark different extents. Darkest gray - allocated space, Lightest gray - free space and medium gray - mix of allocated and free space.
- Blocks/grid, End LBN of Window and Free Block, are again self explanatory.
- The colored boxes that follow serve as the key to understanding the volume-map. The colors green, cyan, blue, magenta and red represent directories, indexed files, placed files, no move files and open files respectively.
- Click on a directory (green) or an open file (red) for the Volume Map File Information dialog see - Figure 6.17, "Volume Map File Information". It has details concerning the selected entity with respect to its volume.

- Click on OK to close that dialog.

Note

Click on Apply in the Graphic - Full volume Map dialog to find out if the selected volume is being defragmented currently.

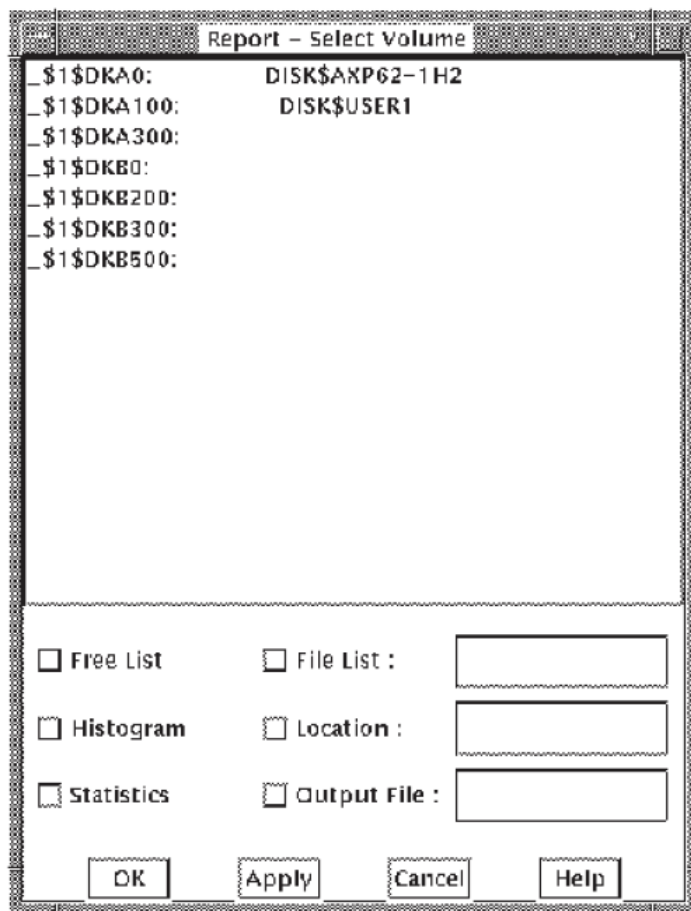
6.6.2. Text Format

Use the following procedure for a text display of the selected volume(s).

1. Click on the Text... suboption of the Report-Volume Fragmentation menu. See *Figure 6.21, "Report Menu"* for the Report - Select Volume dialog.

Use the following procedure to show volume information:

- a. Select the Report pull-down menu. See *Figure 5.15, "Report Menu"*.
- b. Click on the Volume fragmentation option of the Report pull-down menu for the report type list - Graph/Text report. Click on 'Text' to select it. You will see the 'Report - Select Volume' dialog. See *Figure 5.17, "The Report Volume Selection Dialog Box"* and to *Table 5.7, "Report Volume Option and Field Descriptions"*.
- c. Select a volume from the list in the Show Volume options dialog box.
- d. Select any of the options for showing volume information. *Table 5.7, "Report Volume Option and Field Descriptions"* describes the options.
 - If you keep the Statistics option(default), and double click on the volume or click on 'OK' you will see a scrollable page of a Fragmentation Report showing fragmentation index, Freespace Summary, and File Fragmentation Summary, plus more information that you can scroll down. Refer to *Figure 5.18, "Report Volume: Statistics"*.
 - If you select the Histogram option, you will see that portion of the Fragmentation Report that shows the File Fragmentation Histogram for the number of files with a given number of extents and the Volume Free Space Histogram which shows the free space size in LBNs. Refer to *Figure 5.19, "Report Volume: File Fragmentation Histogram"* and *Figure 5.20, "Report Volume: Volume Space Histogram"*.
- e. Finish the Report Volume Defragmentation operation by selecting:
 - OK to show the volume report and optionally create the output file you named in the Output File field. The dialog box disappears.
 - Apply to show the volume report and optionally create the output file you named in the Output File field. The dialog box remains.
 - Cancel to leave the Report option.

Figure 6.23. The Report Volume Selection Dialog Box**Table 6.8. Report Volume Option and Field Descriptions**

Field	Description
Free List	The Free List option is used to show the fragmentation state of the volume. It lists the location and size of all the free space extents on the volume.
Histogram	The Histogram option is used to show the fragmentation state of a volume. It generates two histograms. The first plots the number of files against the number of extents per file and provides information about how many files are fragmented and how badly they are fragmented. The second histogram plots the number of free space extents (holes) against the size of these free space extents. It provides information about how many free space extents are on the volume and their sizes.
Statistics	The Statistics option is used to show the volume fragmentation state. It provides a summary of the file and free space fragmentation for the volume.
File List	The File List option is used to show the fragmentation state of a volume. It lists all files on the volume that have min_extents or more extents (fragments). By default, the number of file window mapping pointers set for the volume (INITIALIZE { volume } /WINDOW=n) plus one is used. This value lists all files that require at least one window turn to completely map.

Field	Description
	Use the field to specify the files for which you want to show the window mapping.
Location	The Location option shows the extents (fragments) associated with all files specified in the field next to this option. If no files are specified, the Location option appends location information to all files listed with File List option. The supplied file-spec may contain wildcards in place of the directory, file name, extension, or version fields.
Output File	The Output File option is used to show the volume fragmentation state or to show information from the scheduling database. It directs the output of the SHOW subcommand to a file.

Figure 6.24. Report Volume: Statistics

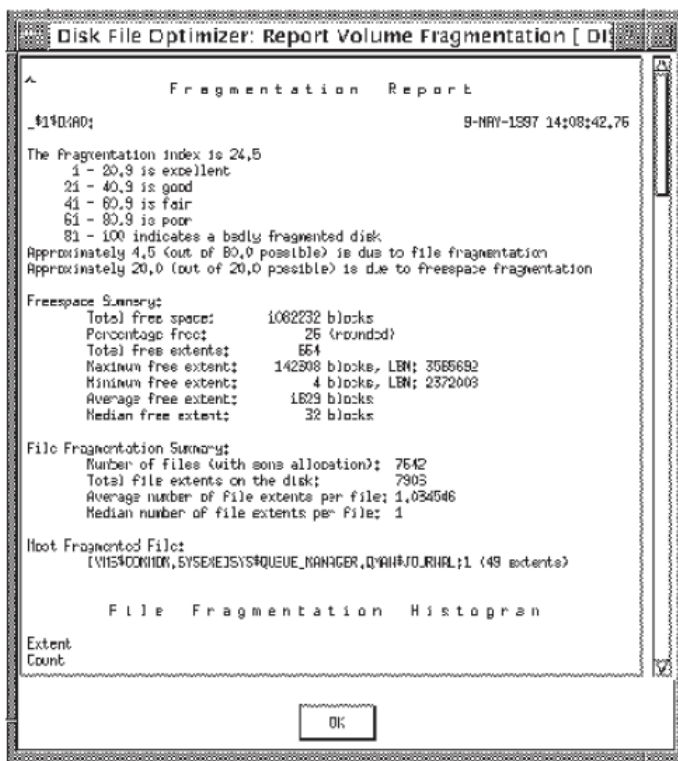
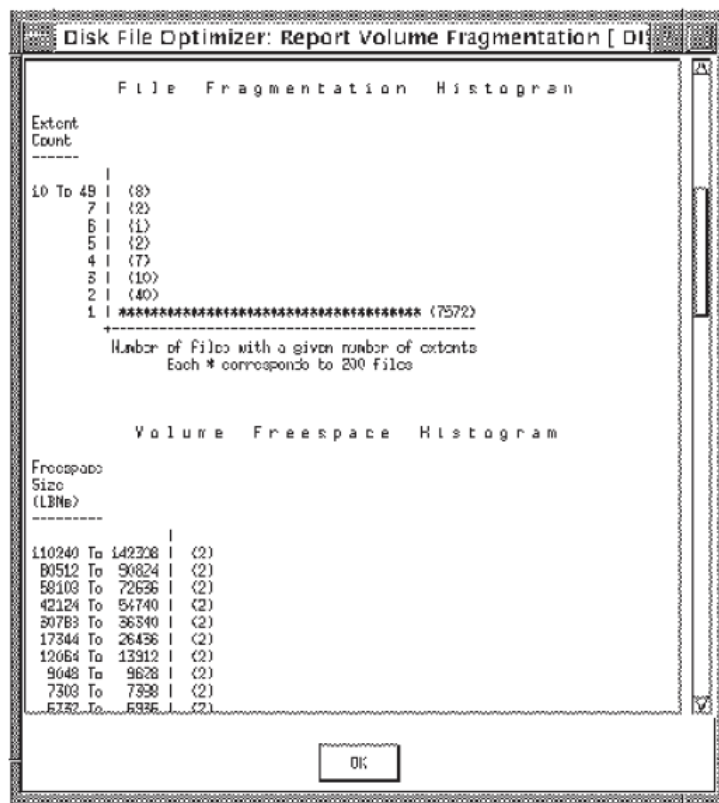
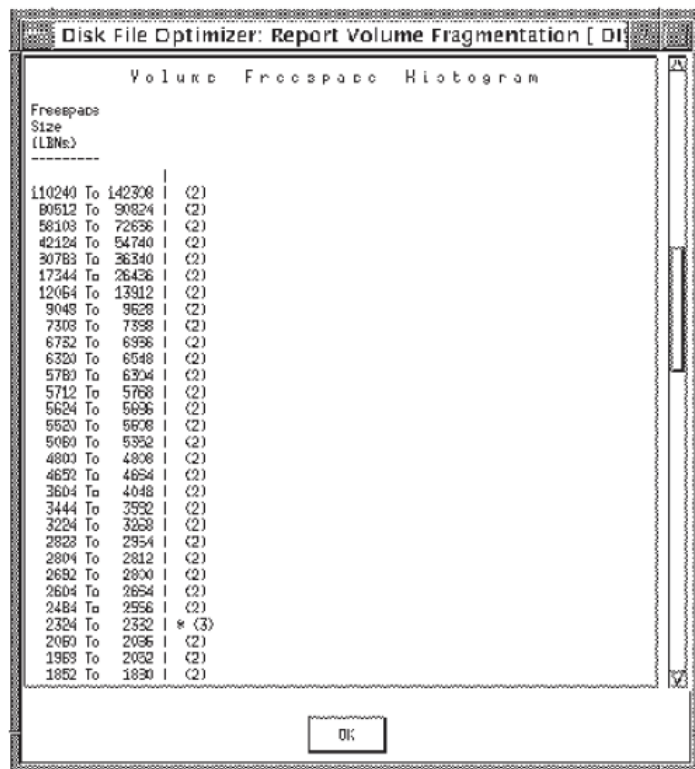


Figure 6.25. Report Volume: File Fragmentation Histogram**Figure 6.26. Report Volume: Volume Space Histogram**

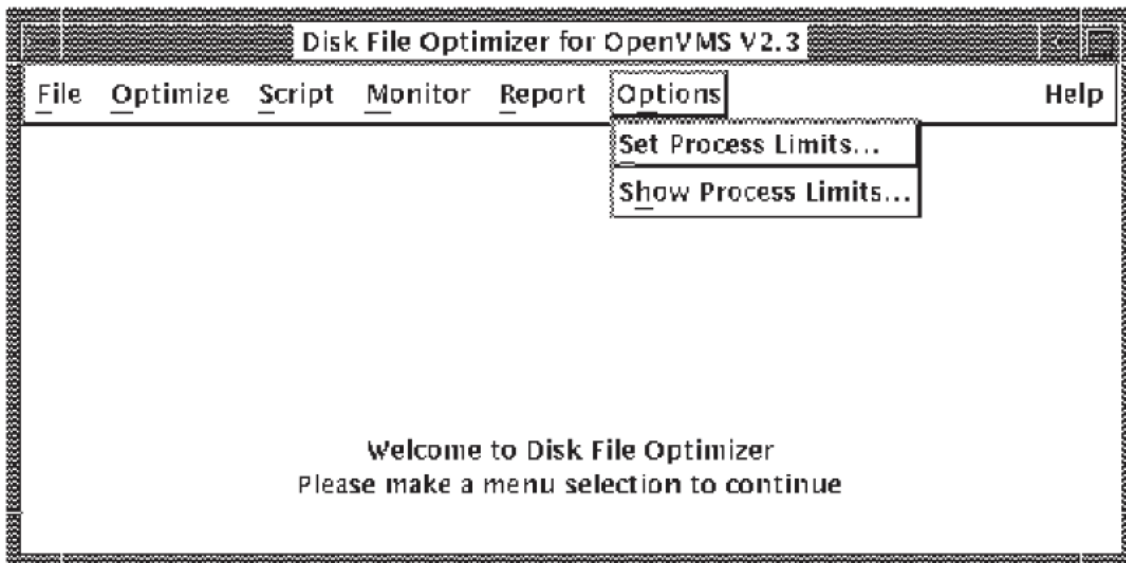
6.7. Options Menu

This pull-down menu option allows you to set and show these Disk File Optimizer process parameters:

- CPU usage
- I/O limit
- Process limit

Figure 6.27, "Options Menu" shows the Options pull-down menu.

Figure 6.27. Options Menu



6.7.1. Setting Process Limits

This option allows you to set the defragmentation process limits for the defragmentation processes on the nodes in your system and allows you to remove unwanted nodes from the database. These process limits define how much of your computing resources you want to dedicate to the defragmentation process. Figure 6.28, "Set Process Dialog Box" shows the dialog box for setting defragmentation process limits.

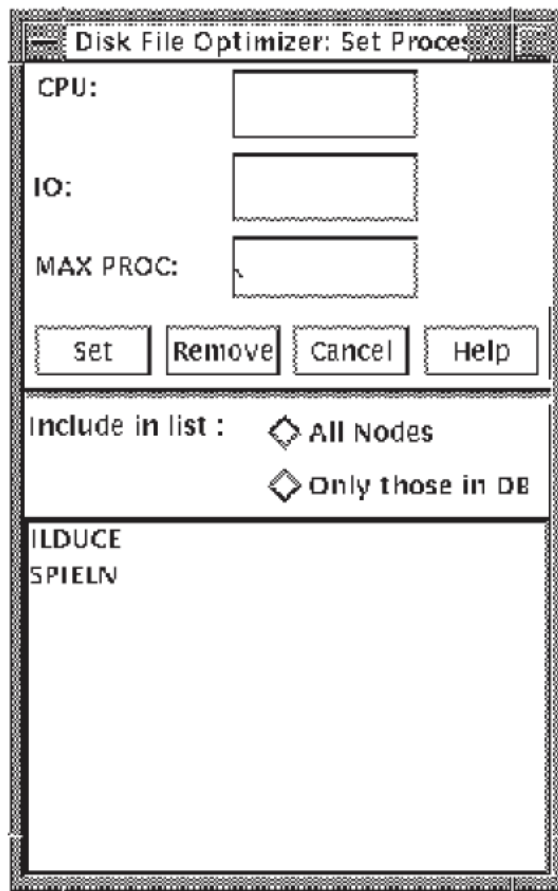
Figure 6.28. Set Process Dialog Box

Table 6.9, "Set Parameters Dialog Box" describes the fields and buttons of the Set Parameters dialog box.

Table 6.9. Set Parameters Dialog Box

Field	Description
CPU:	Enter into this field the percentage of CPU usage for any one defragmentation process. The CPU percent is the percent of a single CPU resources.
IO:	Enter into this field the maximum direct I/O rate (per second) of any one defragmentation process.
MAX PROC:	Enter into this field the maximum number of defragmentation processes per node.
Include in list:	Use the All Nodes button to select all nodes on the VMSccluster. Use the 'Only those in DB' button to select only nodes which have process parameter settings.
Node list	This list presents all nodes in the VMSccluster system from which you select nodes to change defragmentation process parameters.

6.7.1.1. The Set Process Limits Procedure

Use the following procedure to set Defrag PLUS process limits:

1. Select the Options pull-down menu, then select Set Process Limits.
2. Select the node on which the defragmentation process to be changed runs.

IF you are setting process parameters for a node not previously addressed, use the All Nodes option to include nodes on the list.

IF you are setting process parameters for a node in the database, you can use either option to include nodes on the list.

3. Enter the process parameters:
 - CPU
 - IO
 - MAX PROC
4. Exit the Set Process Limits option by selecting the Cancel button.

6.7.1.2. Procedure for Removing Nodes

Before removing nodes, no scripts must have this node in it. Use the following procedure to remove a node from the scheduling and parameter database:

1. Select the Options pull-down menu option, then select Set Process Limits.
2. Set the list to include only those nodes in the database by selecting the Only those in DB option.
3. Select the node which is to be removed from the database.
4. Select Remove button.
5. Exit the Set Process Limits option by selecting the Cancel button.

6.7.2. Showing Defragmentation Process Limits Procedure

This option allows you to show the defragmentation process limits in effect on all nodes in the VMScluster system. You may also produce an output file to report this information. Use the following procedure to show defragmentation process parameters:

1. Select the Options pull-down menu.
2. Select the Show Process Limits option and the Show Process Limits dialogue box appears. Refer to *Figure 6.29, "The Show Process Limits Dialog Box"*.
3. Select to produce an output file.

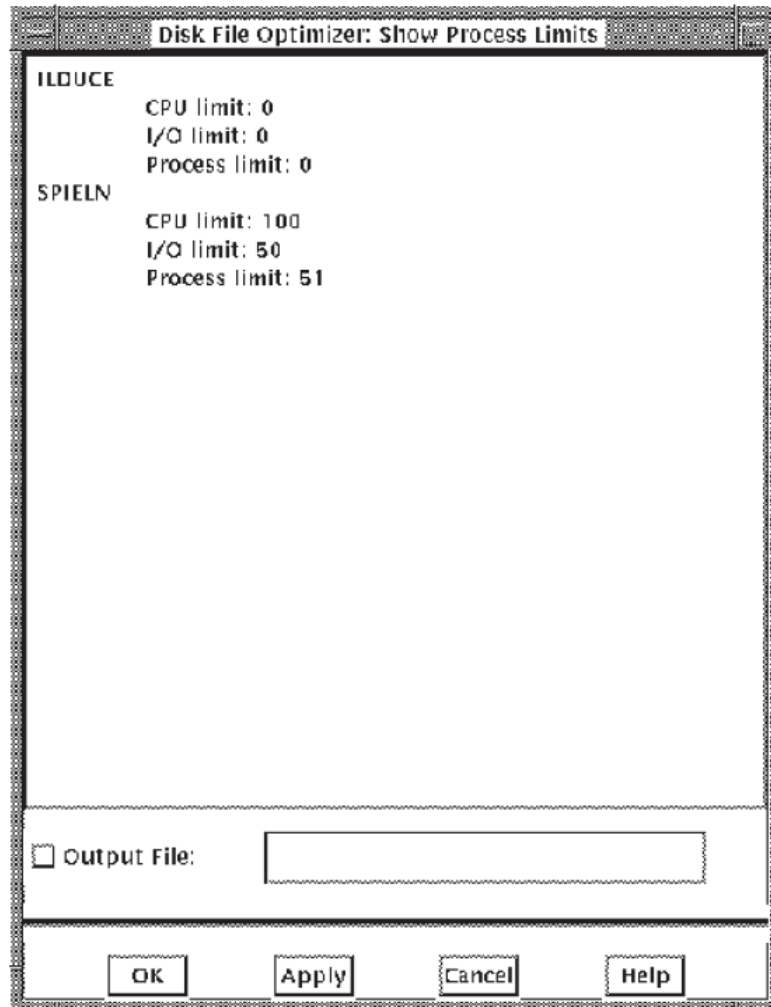
IF you do not want an output file, proceed with Step 4.

IF you want an output file, select the Output File option, and enter the file name.

4. Finish the Show Process Limits option by selecting:
 - OK to leave the Show Process Limits dialog box, and optionally produce the output file.

- Apply to produce and output file. The dialog box remains so you can produce additional output files.
- Cancel to leave the Show Process Limits option.

Figure 6.29. The Show Process Limits Dialog Box



Chapter 7. Callable Interface

7.1. Introduction

This chapter describes the Disk File Optimizer callable interface. It contains three major sections:

- An introduction to using the callable interface
- Details about the DFG\$DEFRAGMENT and DFG\$RELEASE_MEMORY routines and their arguments
- Examples showing how to use the callable interface from various programming environments

7.2. Using the Callable Interface

The callable interface enables you to initiate defragmentation operations from within a program. It is also the interface Defrag PLUS uses to interact with the scheduling database. There is also a call that releases memory consumed as a result of showing the contents of defragmentation process scripts.

The callable interface is available only if you install it with Defrag PLUS software. The Defrag variant does not support the callable interface.

7.2.1. Including Files for DFG\$DEFRAGMENT and DFG\$RELEASE_MEMORY

To compile programs that call the DFG\$DEFRAGMENT and DFG\$RELEASE_MEMORY routines, you need to include the appropriate SYS\$LIBRARY:DFG\$DEF. *xxx* symbol library, where *xxx* stands for the default file extension of various programming languages. For example, if you were using the C programming language, you would include SYS\$LIBRARY:DFG\$DEF.H.

7.2.2. Linking Files with DFG\$DEFRAGMENT and DFG\$RELEASE_MEMORY

When you link a program that calls DFG\$DEFRAGMENT or DFG\$RELEASE_MEMORY, you must specify a link options file containing the following:

```
SYS$SHARE:DFG$DEFRAGMENTSHR/SHARESYS$MESSAGE:DFG$MESSAGE/SHARE
```

Assuming the options file name is LINK_OPTIONS_FILE.OPT, the following command line links MYPROG, which calls DFG\$DEFRAGMENT and DFG\$RELEASE_MEMORY:

```
$ LINK MYPROG.OBJ[,OTHERS.OBJ...],LINK_OPTIONS_FILE.OPT/OPTIONS
```

7.3. DFG\$DEFRAGMENT Routine

This section describes the DFG\$DEFRAGMENT routine.

DFG\$DEFRAGMENT routine

DFG\$DEFRAGMENT routine — The DFG\$DEFRAGMENT routine updates the scheduling database with information from the supplied arguments. After updating the database, DFG\$DEFRAGMENT

returns a completion status in R0. The actual defragmentation operation does not occur until initiated by a scheduler process running on the target nodes.

Format

DFG\$DEFRAGMENT [dfg_list] [sch_list]

Description

DFG\$DEFRAGMENT updates the scheduling database with information provided by the *dfg_list* and *sch_list* parameters. Depending on the specific item descriptors, this information generates one of the following commands:

- **DEFRAGMENT**—Defragments a file, a group of files, or a volume
- **ABORT**—Aborts a defragmentation process
- **CANCEL**—Cancels the next scheduled defragmentation process
- **DISABLE**—Disables a scheduled defragmentation operation
- **ENABLE**—Enables a disabled defragmentation operation
- **REMOVE**—Removes a defragmentation operation from the scheduling database
- **SET**—Sets defragmentation process parameters
- **SHOW**—Shows information about the defragmentation process
- **START**—Starts a defragmentation process

Returns

Longword condition value returned in R0. The condition values that DFG\$DEFRAGMENT returns are listed under **CONDITION VALUES RETURNED**.

Arguments

Table 7.1, "Arguments and Values for DFG\$DEFRAGMENT" lists the arguments and values for the callable interface. Detailed information for each entry in the table follows.

Table 7.1. Arguments and Values for DFG\$DEFRAGMENT

Argument	Item Descriptor Fields	Item Codes
<i>dfg_list</i>	buffer length	DFG\$K_CLASSIFY_DORM
	item code	DFG\$K_CLASSIFY_FREQ
	buffer address	DFG\$K_DFG_FLAGS
	return length address	DFG\$K_EXCLUDE
		DFG\$K_FILE
		DFG\$K_LOG
		DFG\$K_OFFLINE

Argument	Item Descriptor Fields	Item Codes
		DFG\$K_OPTIONS DFG\$K_POST_PROCESS DFG\$K_PRE_PROCESS DFG\$K_VOLUME
<i>sch_list</i>	buffer length item code buffer address return length address	DFG\$K_ABORT DFG\$K_AFTER DFG\$K_CANCEL DFG\$K_DISABLE DFG\$K_ENABLE DFG\$K_INHERIT DFG\$K_INTERVAL DFG\$K_MODIFY DFG\$K_NODE DFG\$K_PREDECESSOR DFG\$K_REMOVE DFG\$K_SCH_FLAGS DFG\$K_SCRIPT DFG\$K_SET_CPU DFG\$K_SET_IO DFG\$K_SET_PROCESSES DFG\$K_SHOW_CPU DFG\$K_SHOW_IO DFG\$K_SHOW_PROCESSES DFG\$K_SHOW_SCRIPT DFG\$K_START DFG\$K_TIME_LIMIT

The *dfg_list* argument is the address of a list of item descriptors, each of which describes one defragmentation-specific option for this command. The list of item descriptors is terminated by a longword of zero.

The *dfg_list* argument is not required; however, you must provide an argument placeholder if it is not used. Specify a longword of zero or the address of along word containing zero as an argument placeholder.

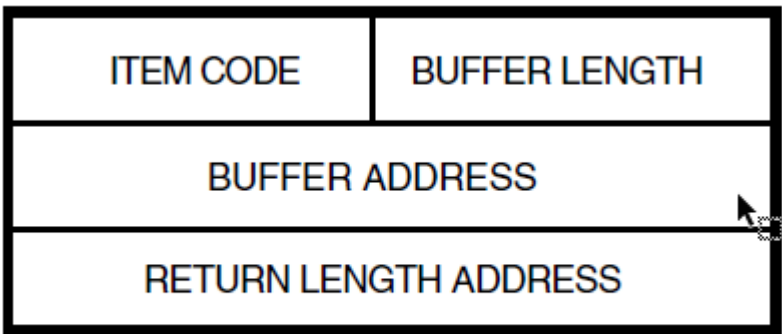
Note

An error occurs if both the *dfg_list* and *sch_list* arguments are zero or are pointers to NULL item lists.

The item descriptors that make up the *dfg_list* have a required order. The first descriptor in the list *must* have either the DFG\$K_FILE or DFG\$K_VOLUME item code. Subsequent item descriptors have no required order.

Figure 7.1, "ITEM_LIST_3 Descriptor" shows a single item_list_3 descriptor.

Figure 7.1. ITEM_LIST_3 Descriptor



CXO-3150A

DFG\$DEFRAGMENT Item Descriptor Fields

buffer length

A word containing a user-supplied integer specifying the length (in bytes) of the buffer or data associated with the buffer address field. The length of the buffer needed depends upon the item code specified in the **item code** field of the item descriptor. If the value of **buffer length** is too small, DFG\$DEFRAGMENT truncates any returned data.

item code

A word containing a user-supplied symbolic code specifying the item of information that the user is supplying to, or receiving from, DFG\$DEFRAGMENT. These symbolic names are described in the DFG\$DEF.XXX files located in SYS\$LIBRARY.

Note

No item code may occur more than once in either the *dfg_list* or *sch_list* item list.

buffer address

A longword containing the user-supplied address of the buffer which contains the information being passed to DFG\$DEFRAGMENT or the address of a buffer into which DFG\$DEFRAGMENT writes information. The use of this field depends upon the item code specified in the item descriptor.

return length address

If used, the **return length address** is a longword containing the user-supplied address of a longword in which DFG\$DEFRAGMENT writes the length, in bytes, of the information it returned. For most DFG\$DEFRAGMENT item codes, this field is not used and must be zero.

dfg_list Item Codes**DFG\$K_CLASSIFY_DORM**

Files associated with the DFG\$K_CLASSIFY_DORM item descriptor are considered dormant when final file placement decisions are made. Dormant files are placed on the volume to allow placing other files in more optimum disk locations. In general, they are placed further away from the ODS-2 data structures than files associated with the DFG\$K_CLASSIFY_FREQ item descriptor and files not classified in any manner.

If a file is indicated in both the dormant and frequent lists, it is considered dormant. If a file is indicated in both the exclude and dormant lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

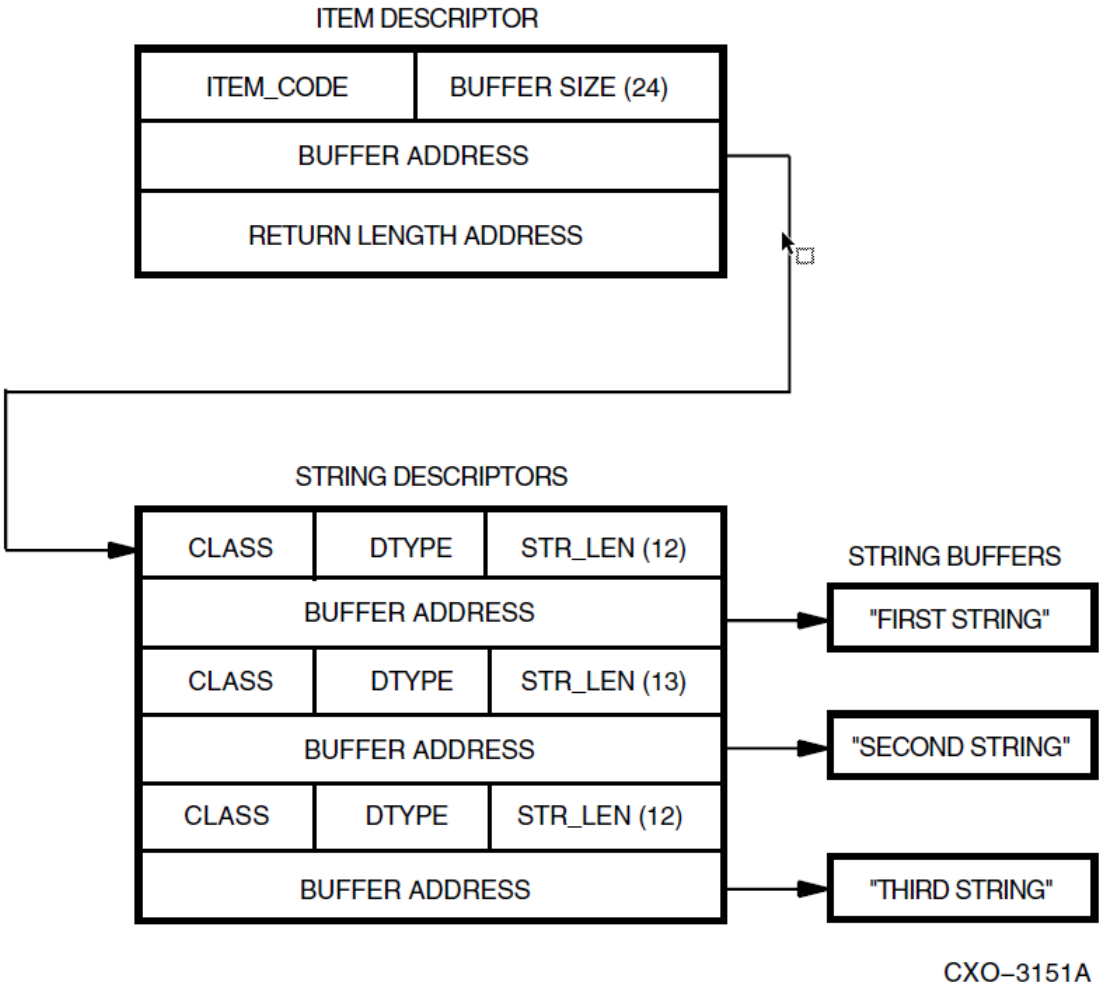
The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*;*. Specifying *.TMP;* selects all *.TMP;* files on the entire volume.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_CLASSIFY_DORM item code are as follows:

- **buffer length**—Contains the length in bytes of a list of string descriptors pointed to by the buffer address. This value must be a multiple of eight.
- **buffer address**—Address of a list of string descriptors (Refer to *Figure 7.2, "Creating a DFG\$K_XXXX Item with Descriptor List"*.)
- **return length address**—Not used; must be zero.

Figure 7.2, "Creating a DFG\$K_XXXX Item with Descriptor List" illustrates the data structures used with the DFG\$K_CLASSIFY_DORM, DFG\$K_CLASSIFY_FREQ, DFG\$K_EXCLUDE, DFG\$K_FILE, and DFG\$K_NODE item codes.

Figure 7.2. Creating a DFG\$K_XXXX Item with Descriptor List



DFG\$K_CLASSIFY_FREQ

The DFG\$K_CLASSIFY_FREQ item descriptor lists files to be considered frequently accessed when final file placement decisions are made. Frequent files are placed to optimize access speed.

If a file is indicated in both the frequent and dormant lists, it is considered dormant. If a file is indicated in both the frequent and exclude lists, it is excluded.

Note

The amount of free space on the volume, the number of files currently being accessed, and other factors influence file placement. The Disk File Optimizer does not guarantee file placement, but makes every effort to place files as described.

The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*;*. Specifying *.TMP;* selects all *.TMP;* files on the entire volume.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_CLASSIFY_FREQ item code are as follows:

- **buffer length**—The length in bytes of a list of string descriptors pointed to by the buffer address. This value must be a multiple of eight.
- **buffer address**—Address of a list of string descriptors. (Refer to *Figure 7.2, "Creating a DFG \$K_XXXX Item with Descriptor List"*.)
- **return length address**—Not used; must be zero.

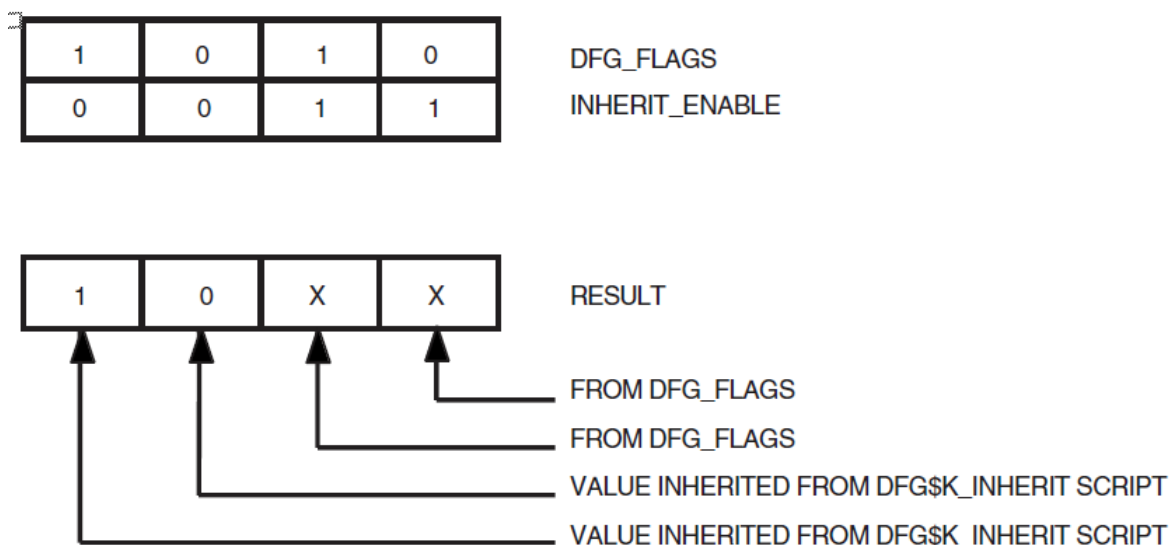
DFG\$K_DFG_FLAGS

The defragment flags provide general information to the defragmentation process about this defragmentation operation. When the DFG\$K_DFG_FLAGS item descriptor is created, the buffer associated with it must be (DFG\$\$_DFG_FLAGS * 2) bytes in size. The first half of the buffer specifies the flag values desired (if any). The second half is nonzero only if the script is inheriting characteristics from another script (that is, the DFG\$K_INHERIT item descriptor is part of this list).

If a DFG\$K_INHERIT item descriptor is part of this item list, the second half of the flags buffer becomes an inherit-enable mask, where each set bit corresponds to a flag with a value that should be inherited from the DFG\$K_INHERIT script. To inherit a flag value from the specified DFG \$K_INHERIT script, set the mask (in the second half of the DFG_FLAGS buffer) corresponding to the field you want to inherit. For example, to inherit the defragmentation level from the DFG \$K_INHERIT script, set the bits associated with DFG\$M_LEVEL mask in the second longword.

Figure 7.3, "Flags and Inherit Enable" illustrates this concept.

Figure 7.3. Flags and Inherit Enable



The following bit masks are defined for DFG_FLAGS:

- **DFG\$M_CONSOLIDATE_FREESPACE**—(1) runs free space consolidation algorithm. (0) does not run free space consolidation algorithm.
- **DFG\$M_DEFAULT_PRIORITY**—Specifies the default priority for the defragmentation process.
- **DFG\$M_FULL**—Provides full (1) or brief (0) error logging.
- **DFG\$M_HONOR**—Honors the placement of files with placement control information. These files become part of the exclude file list.

- **DFG\$M_INDEXED**—(1) defragments and places RMS indexed files. (0) excludes RMS indexed files.
- **DFG\$M_ISSUE_OPCOM**—(1) turns on OPCOM notification. (0) turns off OPCOM notification.
- **DFG\$M_LEVEL**—This three-bit field can have any one of the following values:
 - *DFG\$K_LEVEL_1*—Any file that is not 100 percent contiguous, or that can be selected using levels DFG\$K_LEVEL_2 through DFG\$K_LEVEL_4, is a candidate for defragmentation.
 - *DFG\$K_LEVEL_2*—Any file with extents that cannot be completely mapped by the mapping window, or that can be selected using levels DFG\$K_LEVEL_3 or DFG\$K_LEVEL_4 is a candidate for defragmentation. The default window value is 7 and can be set for a given volume using:


```
$ SET VOLUME {device-spec}/WINDOWS=[n]
```
 - *DFG\$K_LEVEL_3*—Any file with retrieval pointers in more than one file header, or that can be selected using DFG\$K_LEVEL_4 is a candidate for defragmentation.
 - *DFG\$K_LEVEL_4*—This level is reserved for future use. It is functionally equivalent to DFG\$K_LEVEL_3.
 - *DFG\$K_LEVEL_5*—This level requests a volume analysis only; no defragmentation occurs. The statistical results are available in the log file, which is required when DFG\$K_LEVEL_5 is specified.
- **DFG\$M_LOG**—Generates a log file. (See the DFG\$K_LOG item descriptor to specify a log file.)
- **DFG\$M_LOW_PRIORITY**—Specifies the minimum priority for the defragmentation process.
- **DFG\$M_NOTIFY**—Notifies users listed in the DFG\$MAIL_ADDRESS mail distribution list of defragmentation process completion.
- **DFG\$M_RVN_TO_DEFRAG**—Identifies a specific RVN of a bound volume set to defragment.
- **DFG\$M_USE_VPA**—(1) Use the hotfiles database. (0) Do not use the hotfiles database.
- **DFG\$M_WRITE_CHECK**—Performs a read-after-write check of all files moved.

Note

Every DFG\$M_XXXX bit mask value, has a corresponding DFG\$V_XXXX bit field definition.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_DFG_FLAGS item code are as follows:

- **buffer length**—The length of the DFG_FLAGS buffer. This value must be (DFG\$S_DFG_FLAGS * 2).
- **buffer address**—Address of the DFG_FLAGS buffer.

- **return length address**—Not used; must be zero.

DFG\$K_EXCLUDE

Files associated with the DFG\$K_EXCLUDE item descriptor are excluded from processing.

Any file associated with DFG\$K_EXCLUDE and with either the DFG\$K_CLASSIFY_DORM or DFG\$K_CLASSIFY_FREQ item descriptor is excluded from processing.

The file-spec field allows wildcard characters. You can specify wildcard characters in place of the directory name, file name, file type, or file version number field. The default file-spec for all wildcard operations is DISK:[000000...]*.*;*. Specifying *.TMP;* selects all *.TMP;* files on the entire volume.

Note

The file defragmentation utility automatically excludes system (*.SYS) files, open files, files excluded in an options file (DFG\$K_OPTIONS item code), and files excluded using the DFG\$V_HONOR and DFG\$V_INDEXED bits of the DFG\$K_DFG_FLAGS item code.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_EXCLUDE item code are as follows:

- **buffer length**—The length in bytes of a list of string descriptors pointed to by the buffer address. This value must be a multiple of eight.
- **buffer address**—Address of a list of string descriptors. (Refer to *Figure 7.2, "Creating a DFG \$K_XXXX Item with Descriptor List"*.)
- **return length address**—Not used; must be zero.

DFG\$K_FILE

The DEFRAGMENT FILE command, identifies the files to defragment on a specific volume. You must supply either this item code or the DFG\$K_VOLUME item code. The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_FILE item code are as follows:

- **buffer length**—Contains the length in bytes of a list of string descriptors pointed to by the buffer address. This value must be a multiple of eight.
- **buffer address**—Address of a list of string descriptors (Refer to *Figure 7.2, "Creating a DFG \$K_XXXX Item with Descriptor List"*.)
- **return length address**—Not used; must be zero.

DFG\$K_LOG

The defragment process uses the specified file for logging purposes. All pertinent data (determined by the DFG\$M_FULL flag) is written to this file, which is opened for exclusive write.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_LOG item code are as follows:

- **buffer length**—Length of the LOG file specification string, which has a maximum size of 255 bytes.

- **buffer address**—Address of the buffer containing the file specification string for the LOG option. The file specification string may not contain wildcard characters.
- **return length address**—Not used; must be zero.

DFG\$K_OFFLINE

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_OFFLINE item code are as follows:

- **buffer length**—Length of the volume-name string, which has a maximum size of 255 bytes.
- **buffer address**—Address of the buffer containing the volume-name string.
- **return length address**—Not used; must be zero.

DFG\$K_OPTIONS

The options file contains information about files that are classified as frequent, dormant, or excluded.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_OPTIONS item code are as follows:

- **buffer length**—Length of the OPTIONS file specification string, which has a maximum size of 255 bytes.
- **buffer address**—Address of the buffer containing the file specification string for the OPTIONS file. The file specification string may not contain wildcard characters.
- **return length address**—Not used; must be zero.

DFG\$K_POST_PROCESS

The post processing file contains DCL commands that are processed after the defragmentation process completes. The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_POST_PROCESS item code are as follows:

- **buffer length**—Length of the POST_PROCESS file specification string, which has a maximum size of 255 bytes.
- **buffer address**—Address of the buffer containing the file specification string for the POST_PROCESS file. The file specification string may not contain wildcard characters.
- **return length address**—Not used; must be zero.

DFG\$K_PRE_PROCESS

The preprocessing file contains DCL commands that are processed before the defragmentation process begins. The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_PRE_PROCESS item code are as follows:

- **buffer length**—Length of the PRE_PROCESS file specification string, which has a maximum size of 255 bytes.
- **buffer address**—Address of the buffer containing the file specification string for the PRE_PROCESS file. The file specification string may not contain wildcard characters.

- **return length address**—Not used; must be zero.

DFG\$K_VOLUME

If you would use the DEFRAGMENT VOLUME command, rather than the DEFRAGMENT FILE command for this defragmentation process, use this item code to identify the volume to be defragmented. You must supply either this item code or the DFG\$K_FILE item code. The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_VOLUME item code are as follows:

- **buffer length**—Length of the volume-name string, which has a maximum size of 255 bytes.
- **buffer address**—Address of the buffer containing the volume-name string.
- **return length address**—Not used; must be zero.

sch_list	
VMS Usage:	item_list_3
type:	longword (unsigned)
access:	read_only
mechanism:	by reference

The *sch_list* argument is the address of a list of item descriptors, each of which describes one scheduler-specific option for this command. The list of item descriptors is terminated by a longword of zero.

The *sch_list* argument is not required; however, you must provide an argument placeholder if it is not used. Specify a longword of zero or the address of a longword containing zero as an argument placeholder.

Note

An error occurs if both the *dfg_list* and *sch_list* arguments are zero or are pointers to NULL item lists.

Figure 7.1, "ITEM_LIST_3 Descriptor" shows a single item_list_3 descriptor.

If the *dfg_list* argument is not supplied, the item descriptors that make up the *sch_list* have a required order. Otherwise, there is no required order for the item descriptors in the *sch_list*. If the *dfg_list* argument is not supplied, the first descriptor in the *sch_list* must have one of the following item codes:

- DFG\$K_ABORT
- DFG\$K_CANCEL
- DFG\$K_DISABLE
- DFG\$K_ENABLE
- DFG\$K_REMOVE

- DFG\$K_SET_XXX
- DFG\$K_SHOW_XXX
- DFG\$K_START

Most of these item codes are incompatible with each other. For example, it is invalid to have an item list containing a DFG\$K_DISABLE item descriptor and a DFG\$K_ENABLE item descriptor. The exceptions are the DFG\$K_SET_XXX and DFG\$K_SHOW_XXX item codes. It is valid to have an item list containing any combination of SET item descriptors, or an item list containing any combination of SHOW item descriptors. A combination of SET and SHOW item descriptors is not allowed.

Each item code is related directly to an associated DCL command. For example, DFG\$K_ABORT has the same effect as the DCL command, DEFRAGMENT ABORT {script-name}.

***sch_list* Item Codes**

DFG\$K_ABORT

DFG\$DEFRAGMENT queues an abort request for the specified script. The script is aborted when the scheduler processes the abort request.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_ABORT item code are as follows:

- **buffer length**—Length of the script-name in bytes. The maximum size of a script-name is 64 bytes.
- **buffer address**—Address of the script-name buffer.
- **return length address**—Not used; must be zero.

The script-name can be as follows:

- A valid ASCII script-name. In this case, the specified script is aborted.
- An unsigned longword that has been assigned the value DFG\$K_ALL_SCRIPTS. The buffer length field of the item descriptor **must** be four. In this case, DFG\$DEFRAGMENT queues abort requests for all currently executing scripts.

DFG\$K_AFTER

DFG\$K_AFTER indicates the time after which the scheduler is to start the requested operation.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_AFTER item code are as follows:

- **buffer length**—Length of the time specification string in bytes. The buffer must be long enough to hold an absolute date/time specification as described in the *VMS Concepts Manual*.
- **buffer address**—Address of a character string buffer containing the absolute date/time specification.
- **return length address**—Not used; must be zero.

DFG\$K_CANCEL

DFG\$DEFRAGMENT queues a cancel request for the specified script. DFG\$K_CANCEL skips the next execution of the script.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_CANCEL item code are as follows:

- **buffer length**—Length of the script-name in bytes. The maximum size of a script-name is 64 bytes.
- **buffer address**—Address of the script-name buffer.
- **return length address**—Not used; must be zero.

The script-name can be as follows:

- A valid ASCII script-name. In this case, the specified script is canceled.
- An unsigned longword that has been assigned the value DFG\$K_ALL_SCRIPTS. The buffer length field of the item descriptor **must** be four. In this case, DFG\$DEFRAGMENT queues cancel requests for all scripts.

DFG\$K_DISABLE

DFG\$DEFRAGMENT disables the specified script. Once disabled, the script cannot execute until enabled using the DFG\$K_ENABLE item code.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_DISABLE item code are as follows:

- **buffer length**—Length of the script-name in bytes. The maximum size of a script-name is 64 bytes.
- **buffer address**—Address of the script-name buffer.
- **return length address**—Not used; must be zero.

The script-name can be:

- A valid ASCII script-name. In this case, the specified script is disabled.
- An unsigned longword that has been assigned the value DFG\$K_ALL_SCRIPTS. The buffer length field of the item descriptor **must** be four. In this case, DFG\$DEFRAGMENT queues disable requests for all scripts.

DFG\$K_ENABLE

DFG\$DEFRAGMENT enables the specified script. Once enabled, the script operates normally.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_ENABLE item code are as follows:

- **buffer length**—Length of the script-name in bytes. The maximum size of a script-name is 64 bytes.
- **buffer address**—Address of the script-name buffer.

- **return length address**—Not used; must be zero.

The script-name can be:

- A valid ASCII script-name. In this case, the specified script is enabled.
- An unsigned longword that has been assigned the value `DFG$K_ALL_SCRIPTS`. The buffer length field of the item descriptor **must** be four. In this case, `DFG$DEFRAGMENT` queues enable requests for all scripts.

DFG\$K_INHERIT

The `DFG$K_INHERIT` item descriptor specifies the name of a script that is to be inherited as well as a longword bit mask (`INHERIT_MASK`) that indicates which fields in the script may be inherited. The `INHERIT_MASK` structure has the following bit definitions:

- *DFG\$V_INHERIT_AFTER* – When set to (1), the start time of an inherit script can be copied to the new script. When reset to (0), the start time cannot be copied.
- *DFG\$V_INHERIT_DORMANT* – When set to (1), the dormant file list of an inherit script can be copied to the new script. When reset to (0), the dormant file list cannot be copied.
- *DFG\$V_INHERIT_EXCLUDE* – When set to (1), the exclude file list of an inherit script can be copied to the new script. When reset to (0), the exclude file list cannot be copied.
- *DFG\$V_INHERIT_FILE* – When set to (1), the file or list of files to be defragmented can be copied from an inherit script to the new script. When reset to (0), the file list cannot be copied.
- *DFG\$V_INHERIT_FREQUENT* – When set to (1), the frequent file list of an inherit script can be copied to the new script. When reset to (0), the frequent file list cannot be copied.
- *DFG\$V_INHERIT_INTERVAL* – When set to (1), the interval of an inherit script can be copied to the new script. When reset to (0), the interval cannot be copied.
- *DFG\$V_INHERIT_LOG* – When set to (1), the log file specification of an inherit script can be copied to the new script. When reset to (0), the log file specification cannot be copied.
- *DFG\$V_INHERIT_NODE* – When set to (1), the node list associated with the inherit script can be copied to the new script. When reset to (0), the node list cannot be copied.
- *DFG\$V_INHERIT_OPTIONS* – When set to (1), the options file specification of the inherit script can be copied to the new script. When reset to (0), the options file specification cannot be copied.
- *DFG\$V_INHERIT_POST_PROC* – When set to (1), the postprocessing file specification of the inherit script can be copied to the new script. When reset to (0), the postprocessing file specification cannot be copied.
- *DFG\$V_INHERIT_PRE_PROC* – When set to (1), the preprocessing file specification of the inherit script can be copied to the new script. When reset to (0), the preprocessing file specification cannot be copied.
- *DFG\$V_INHERIT_PREDECESSOR* – When set to (1), the predecessor associated with the inherit script can be copied to the new script. When reset to (0), the predecessor cannot be copied.

Note

Inheriting the predecessor of one script to another changes the linked set of scripts. For example, if a set of linked script A,B,C exists and a new script, D is created by inheriting script C, the link changes to A, B, D, C.

- **DFG\$V_INHERIT_TIME_LIMIT** – When set to (1), the time limit associated with the inherit script can be copied to the new script. When reset to (0), the time limit cannot be copied.

DFG\$DEFRAGMENT creates or modifies a database entry for this script using the following method:

- Copies all enabled fields from the script specified by the DFG\$K_INHERIT item code to the new script. Fields are enabled by setting the corresponding bit in the INHERIT_MASK bit mask.
- Overwrites any fields in this copy with those specified by other item descriptors in the item list. (Refer also to the DFG\$K_DFG_FLAGS and DFG\$K_SCH_FLAGS item codes.)
- Names the new script as appropriate. The script name can be specified using the DFG \$K_SCRIPT item descriptor or can be assigned by DFG\$DEFRAGMENT.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_INHERIT item code are as follows:

- **buffer length**—Must be 12.
- **buffer address**—Address of a three longword structure containing a string descriptor for the inherit script name string and the longword INHERIT_MASK. The script name provided indicates the script from which DFG\$DEFRAGMENT inherits all characteristics except those supplied with other item descriptors in this item list.
- **return length address**—Not used; must be zero.

DFG\$K_INTERVAL

Specifies the interval the scheduler is to use for repetitive executions of this script. The interval is specified using a delta-time specification as described in the *VMS Concepts Manual*.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_INTERVAL item code are as follows:

- **buffer length**—Length of the delta-time specification in bytes.
- **buffer address**—Address of a buffer containing the delta-time specification.
- **return length address**—Not used; must be zero.

DFG\$K_MODIFY

The DFG\$K_MODIFY item descriptor specifies a script that is to be modified and a longword bit mask (INHERIT_ENABLE_MASK) that indicates which fields in the script can be modified. Other DFG\$DEFRAGMENT item codes must be used to identify the specific modifications. For example,

to change the time a script is scheduled to execute, the `DFG$V_INHERIT_AFTER` bit must be set to 1 and the `DFG$K_AFTER` item code supplies to value to change to.

The `INHERIT_ENABLE_MASK` structure has the following bit definitions:

- *DFG\$V_INHERIT_AFTER* – When set to (1), the start time of the script can be modified. When reset to (0), the existing start time is removed from the script.
- *DFG\$V_INHERIT_DORMANT* – When set to (1), the dormant file list of the script can be modified. When reset to (0), the dormant file list is removed from the script.
- *DFG\$V_INHERIT_EXCLUDE* – When set to (1), the exclude file list of the script can be modified. When reset to (0), the exclude file list is removed from the script.
- *DFG\$V_INHERIT_FILE* – When set to (1), the file or list of files to be defragmented can be modified. When reset to (0), the file list is removed.
- *DFG\$V_INHERIT_FREQUENT* – When set to (1), the frequent file list of the script can be modified. When reset to (0), the frequent file list is removed from the script.
- *DFG\$V_INHERIT_INTERVAL* – When set to (1), the interval of the script can be modified. When reset to (0), the interval is removed from the script.
- *DFG\$V_INHERIT_LOG* – When set to (1), the log file specification of the script can be modified. When reset to (0), the log file specification is removed from the script.
- *DFG\$V_INHERIT_NODE* – When set to (1), the node list associated with the script can be modified. When reset to (0), the node list is removed from the script.
- *DFG\$V_INHERIT_OPTIONS* – When set to (1), the options file specification of the script can be modified. When reset to (0), the options file specification is removed from the script.
- *DFG\$V_INHERIT_POST_PROC* – When set to (1), the postprocessing file specification can be modified. When reset to (0), the postprocessing file specification is removed.
- *DFG\$V_INHERIT_PRE_PROC* – When set to (1), the preprocessing file specification can be modified. When reset to (0), the preprocessing file specification is removed.
- *DFG\$V_INHERIT_PREDECESSOR* – When set to (1), the predecessor associated with the script can be modified. When reset to (0), the predecessor is removed from the script.

Note

Modifying the predecessor of a script changes the linked set of scripts. For example, if a set of linked scripts A, B, C exists and B is modified to have no predecessor, then the link changes to B, C.

- *DFG\$V_INHERIT_SCRIPT* – When set to (1), the script name associated with the script can be modified. When reset to (0), the script name is removed.
- *DFG\$V_INHERIT_TIME_LIMIT* – When set to (1), the time limit associated with the script can be modified. When reset to (0), the time limit is removed from the script.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the `DFG $K_MODIFY` item code are as follows:

- **buffer length**—Must be 12.
- **buffer address**—Address of a three longword structure containing a string descriptor for the inherit script name string and the longword INHERIT_MASK. The script name provided indicates the script from which DFG\$DEFRAGMENT inherits all characteristics except those supplied with other item descriptors in this item list.
- **return length address**—Not used; must be zero.

DFG\$K_NODE

When an item list contains the DFG\$K_NODE item descriptor, the command is limited to those nodes associated with DFG\$K_NODE. If the DFG\$K_NODE item descriptor is not present, the command applies to all nodes accessible from the local system. *Figure 7.2, "Creating a DFG \$K_XXXX Item with Descriptor List"* illustrates how to construct a DFG\$K_NODE item descriptor.

If DFG\$K_SET_XXX or DFG\$K_SHOW_XXX are present, DFG\$K_NODE can not be the first item descriptor in the item list.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_NODE item code are as follows:

- **buffer length**—Length in bytes of a list of string descriptors pointed to by the buffer address.
- **buffer address**—Address of a list of string descriptors that describe the node name strings. The script (or set-show data) is limited to these nodes.
- **return length address**—Not used; must be zero.

DFG\$K_PREDECESSOR

The DFG\$K_PREDECESSOR item code specifies the script to precede this one in a linked execution. Using the DFG\$K_PREDECESSOR item code, scripts can be chained together to execute sequentially.

The predecessor script name must exist in the defragment database or DFG\$DEFRAGMENT returns an appropriate error status. Further, any attempt to name a continuously executing script as a predecessor results in an error. Linked scripts execute sequentially.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_PREDECESSOR item code are as follows:

- **buffer length**—Length of the predecessor script-name string in bytes. The maximum size of a script-name string is 64 bytes.
- **buffer address**—Address of a buffer containing the script-name string.
- **return length address**—Not used; must be zero.

DFG\$K_REMOVE

DFG\$REMOVE deletes the specified script from the database. If the script is currently executing, it is allowed to complete before it is deleted.

If the specified script is not in the scheduling database, DFG\$DEFRAGMENT returns an error status indicating the script could not be found.

If DFG\$K_REMOVE is present, it must be the first item descriptor in the item list.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_REMOVE item code are as follows:

- **buffer length**—Length of the script-name string in bytes. The maximum size of a script-name string is 64 bytes.
- **buffer address**—Address of a buffer containing the script-name string.
- **return length address**—Not used; must be zero.

The script-name can be:

- A valid ASCII script-name. In this case, the specified script is removed.
- An unsigned longword that has been assigned the value DFG\$K_ALL_SCRIPTS. The buffer length field of the item descriptor **must** be four. In this case, DFG\$DEFRAGMENT queues remove requests for all scripts.

DFG\$K_SCH_FLAGS

The scheduler flags control the scheduler mode (continuous or scheduled) and command type (DEFRAGMENT FILE or DEFRAGMENTVOLUME) of this script. When the DFG \$K_SCH_FLAGS item descriptor is created, the buffer associated with it must be (DFG \$\$SCH_FLAGS * 2) bytes in size. The first half of the buffer specifies the flag values desired (if any). The second half is zero, unless the script is inheriting characteristics from another script (that is, the DFG\$K_INHERIT item descriptor is part of this list).

If a DFG\$K_INHERIT item descriptor is part of this item list, then the second half of the flags buffer becomes an inherit-enable mask, where each set bit corresponds to a flag whose value should be inherited from the DFG\$K_INHERIT script. To inherit a flag value from the specified DFG \$K_INHERIT script, set the mask (in the second half of the SCH_FLAGS buffer) corresponding to the field you want to inherit. To inherit the continuous flag from the DFG\$K_INHERIT script, assign the second longword of the SCH_FLAGS the value of DFG\$M_CONTINUOUS. *Figure 7.3, "Flags and Inherit Enable"* illustrates this concept.

The following mask bits are defined:

- *DFG\$M_CONTINUOUS*—When set (1), the script runs in continuous mode: After completing one defragment job, the scheduler begins a new one immediately with the same script. When reset (0), the script does not run in continuous mode.
- *DFG\$M_FILE_COMMAND*—When set (1), identifies the command as a DEFRAGMENT FILE command. When reset(0), identifies the command as a DEFRAGMENT VOLUME command.
- *DFG\$M_SPARE_SCH*—Reserved for future use by VSI. Should be zero.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_SCH_FLAGS item code are as follows:

- **buffer length**—Must be (DFG\$\$SCH_FLAGS * 2).
- **buffer address**—Address of the SCH_FLAGS buffer.
- **return length address**—Not used; must be zero.

DFG\$K_SCRIPT

The DFG\$K_SCRIPT item code allows you to name the created script. If the script-name is in the database, DFG\$DEFRAGMENT issues a duplicate script name error message and does not enter the new script into the database.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_SCRIPT item code are as follows:

- **buffer length**—Length of the script-name string in bytes. The maximum size of a script-name string is 64 bytes.
- **buffer address**—Address of a buffer containing the script-name string.
- **return length address**—Not used; must be zero.

DFG\$K_SET_CPU

The DFG\$K_SET_CPU item code sets the maximum percentage of a single CPU that is available to a defragmentation process. This percentage must be a number from 0 to 100, inclusive. If a defragmentation process exceeds this limit, the scheduler decrements the process priority of the defragmentation process once every minute until:

- This limit is no longer exceeded.
- The defragmentation process priority is at the SYSGEN parameter, DEFPRI, minus 1.

Momentary violations of this limit are ignored.

Depending on whether the DFG\$K_NODE item descriptor is part of this item list, the CPU limit can affect all nodes or only those listed as part of the DFG\$K_NODE item descriptor. Further, if DFG\$K_NODE is not present, the value provided with DFG\$K_SET_CPU is assigned to every node in the scheduling database and becomes the default value for subsequent nodes not already in the database.

DFG\$K_SET_CPU can be the first item descriptor in an item list that optionally contains other DFG\$K_SET_XXXX item descriptors and an optional DFG\$K_NODE descriptor.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_SET_CPU item code are as follows:

- **buffer length**—Must be four.
- **buffer address**—Address of a longword buffer containing the unsigned maximum, integral, CPU percentage for defragment operations. A value of zero imposes no limit.
- **return length address**—Not used; must be zero.

DFG\$K_SET_IO

The DFG\$K_SET_IO item code sets the maximum direct I/O rate (per second) available to a defragmentation process. If a defragmentation process exceeds this limit, the scheduler decrements the process priority of the defragmentation process once every minute until:

- This limit is no longer exceeded.

- The defragmentation process priority is at the SYSGEN parameter, DEFPRI, minus 1.

Momentary violations of this limit are ignored.

Depending on whether the DFG\$K_NODE item descriptor is part of this item list, the I/O limit may affect all nodes, or only those listed as part of the DFG\$K_NODE item descriptor. Further, if DFG\$K_NODE is not present, the value provided with DFG\$K_SET_IO is assigned to every node in the scheduling database and becomes the default value for subsequent nodes not already in the database.

DFG\$K_SET_IO may be the first item descriptor in an item list that optionally contains other DFG\$K_SET_XXXX item descriptors and an optional DFG\$K_NODE descriptor.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_SET_IO item code are as follows:

- **buffer length**—Must be four.
- **buffer address**—Address of a longword buffer containing the unsigned maximum I/O rate for defragment operations. If zero is specified, no limit is imposed.
- **return length address**—Not used; must be zero.

DFG\$K_SET_PROCESSES

The DFG\$K_SET_PROCESSES item code limits the number of defragmentation processes that the scheduler starts on a given node. If the number of defragmentation processes is greater than or equal to the process limit, the scheduler does not start additional defragmentation processes.

The scheduler does not abort scripts to reduce the number of defragmentation processes. Instead, it stops creating new processes and allows normal process completion to reduce the number of concurrent defragmentation processes.

Depending on whether the DFG\$K_NODE item descriptor is part of this item list, the process limit may affect all nodes or only those listed as part of the DFG\$K_NODE item descriptor. Further, if DFG\$K_NODE is not present, the value provided with DFG\$K_SET_PROCESSES is assigned to every node in the scheduling database and becomes the default value for subsequent nodes not already in the database.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_SET_PROCESSES item code are as follows:

- **buffer length**—Must be four.
- **buffer address**—Address of an unsigned longword buffer containing the maximum number of defragmentation processes allowed on each node of the VAXcluster (DFG\$K_NODE not present) or on the given nodes (DFG\$K_NODE present). Specifying a limit of zero imposes no limit.
- **return length address**—Not used; must be zero.

DFG\$K_SHOW_CPU

When the DFG\$K_SHOW_CPU item descriptor is encountered, DFG\$DEFRAGMENT returns the following data in the user-supplied buffer:

- **If DFG\$K_NODE is present:**

The CPU limit for each node supplied in the DFG\$K_NODE item descriptor.

- **If DFG\$K_NODE is not present:**

The default CPU limit applied to all nodes that do not have an explicit CPU limit.

A DFG\$K_NODE item descriptor may be associated with the DFG\$K_SHOW_CPU item list. DFG\$K_NODE can indicate any positive number ($n > 0$) of node names. If multiple node names are requested, DFG\$DEFRAGMENT returns the CPU percentage in the same order as the node names provided.

If the user buffer is too small for all data, the data is truncated and DFG\$DEFRAGMENT returns an appropriate status.

DFG\$K_SHOW_CPU may be the first item descriptor in an item list that optionally contains other DFG\$K_SHOW_XXXX item descriptors and a DFG\$K_NODE descriptor.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_SHOW_CPU item code are as follows:

- **buffer length**—Size of the user-supplied buffer in bytes. The length of the buffer must be at least ($\text{number_of_nodes} * 4$) bytes and must be an integral multiple of four.
- **buffer address**—Address of the user-supplied buffer into which DFG\$DEFRAGMENT writes the current maximum CPU percentage.
- **return length address**—Address of a longword buffer into which DFG\$DEFRAGMENT writes the number of bytes returned in the buffer pointed to by the buffer address.

DFG\$K_SHOW_IO

When the DFG\$K_SHOW_IO item descriptor is encountered, DFG\$DEFRAGMENT returns the following data in the user-supplied buffer:

- **If DFG\$K_NODE is present:**

The direct I/O limit for each node supplied in the DFG\$K_NODE item descriptor.

- **If DFG\$K_NODE is not present:**

The default I/O limit applied to all nodes that do not have an explicit I/O limit.

A DFG\$K_NODE item descriptor may be associated with the DFG\$K_SHOW_IO item descriptor. DFG\$K_NODE can indicate any positive number of node names. If multiple node names are requested, DFG\$DEFRAGMENT returns the I/O rate for each node in the same order as the node names provided.

If the user buffer is too small for all data, the data is truncated and DFG\$DEFRAGMENT returns an appropriate status.

DFG\$K_SHOW_IO may be the first item descriptor in an item list that optionally contains other DFG\$K_SHOW_XXXX item descriptors and a DFG\$K_NODE descriptor.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_SHOW_IO item code are as follows:

- **buffer length**—Size of the user-supplied buffer in bytes. This buffer must be at least (number_of_nodes * 4) bytes and an integral multiple of four.
- **buffer address**—Address of the user-supplied buffer into which DFG\$DEFRAGMENT writes the current maximum I/O data.
- **return length address**—Address of a longword buffer into which DFG\$DEFRAGMENT writes the number of bytes returned in the buffer pointed to by the buffer address.

DFG\$K_SHOW_PROCESSES

When the DFG\$K_SHOW_PROCESSES item descriptor is encountered, DFG\$DEFRAGMENT returns the following data in the user-supplied buffer:

- **If DFG\$K_NODE is present:**

The process limit for each node supplied in the DFG\$K_NODE item descriptor.

- **If DFG\$K_NODE is not present:**

The default process limit applied to all nodes that do not have an explicit process limit.

A DFG\$K_NODE item descriptor may be associated with the DFG\$K_SHOW_PROCESSES item descriptor. DFG\$K_NODE can indicate any positive number of node names. If multiple node names are requested, DFG\$DEFRAGMENT returns the process limit for each node in the same order as the node names provided.

If the user buffer is too small for all data, the data is truncated and DFG\$DEFRAGMENT returns an appropriate status.

DFG\$K_SHOW_PROCESSES may be the first item descriptor in an item list that optionally contains other DFG\$K_SHOW_XXXX item descriptors and a DFG\$K_NODE descriptor.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_SHOW_PROCESSES item code are as follows:

- **buffer length**—Size of the user-supplied buffer in bytes. This buffer must be at least (number_of_nodes * 4) bytes and an integral multiple of four.
- **buffer address**—Address of the user-supplied buffer into which DFG\$DEFRAGMENT writes the maximum processes information.
- **return length address**—Address of a longword buffer into which DFG\$DEFRAGMENT writes the number of bytes returned in the buffer pointed to by the buffer address.

DFG\$K_SHOW_SCRIPT

The DFG\$K_SHOW_SCRIPT item code requests that DFG\$DEFRAGMENT return information about the script associated with the DFG\$K_SCRIPT item descriptor (which must be part of the item list containing the DFG\$K_SHOW_SCRIPT item code). The information is returned in a buffer allocated by DFG\$DEFRAGMENT. The address and length of the buffer allocated are returned in longword buffers provided by the caller.

DFG\$K_SHOW_SCRIPT may be the first item descriptor in an item list that optionally contains other DFG\$K_SHOW_XXXX item descriptors and a DFG\$K_NODE descriptor.

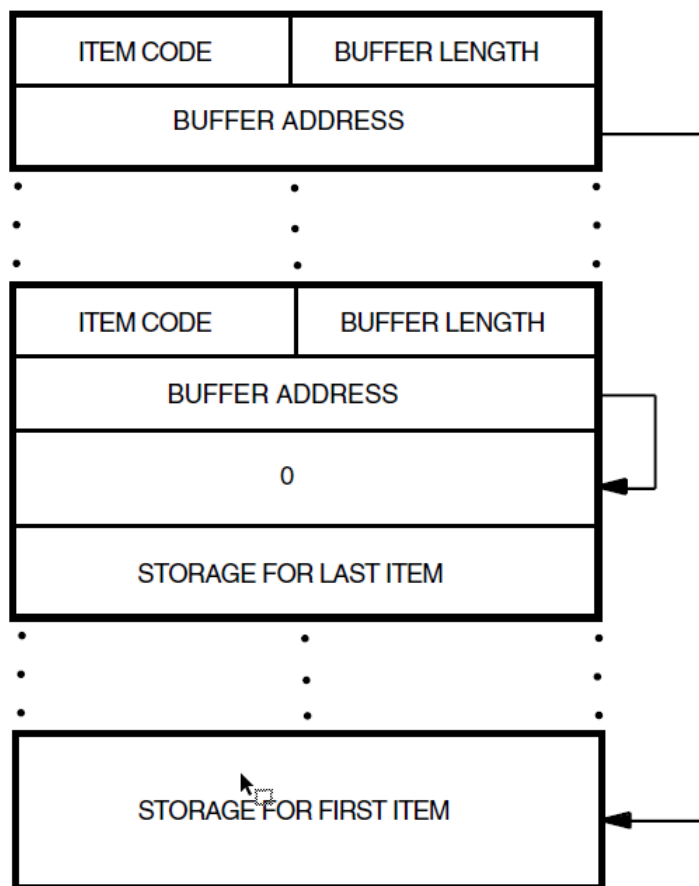
The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG \$K_SHOW_SCRIPT item code are as follows:

- **buffer length**—Must be four, which is the length of the buffer that holds the address of the data block allocated by DFG\$DEFRAGMENT.
- **buffer address**—Address of a longword buffer that DFG\$DEFRAGMENT fills with the address of the buffer it allocates to hold the script data.
- **return length address**—Address of a longword that DFG\$DEFRAGMENT fills with the number of bytes it allocated for the script data.

The buffer created by DFG\$DEFRAGMENT consists of two parts. The first part is a contiguous list of VMS ITEM_LIST_2 descriptors, terminated by a longword of zero. These descriptors are identical to the ITEM_LIST_3 descriptors as shown in *Figure 7.1, "ITEM_LIST_3 Descriptor"*, except they do not have a **return length address** field. The second part of the script data buffer consists of all the data indicated by the descriptors in the first part.

Figure 7.4, "Show Script Data Buffer" illustrates the format of the buffer allocated and written by DFG\$DEFRAGMENT.

Figure 7.4. Show Script Data Buffer



CXO-3153A

The possible item codes associated with the ITEM_LIST_2 descriptors returned by DFG \$DEFRAGMENT are:

DFG\$K_SHOW_SCRIPT_AFTER

DFG\$DEFRAGMENT returns the time that was specified with the DFG\$K_AFTER item descriptor when the script was originally created. The time is a VMS absolute time string. If no DFG\$K_AFTER item descriptor was originally supplied, DFG\$DEFRAGMENT does not return this descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the time string supplied in the buffer pointed to by the **buffer address** field.
- **buffer address**—Address of the time string buffer.

There is only one DFG\$K_SHOW_SCRIPT_AFTER item descriptor per script.

DFG\$K_SHOW_SCRIPT_DFG_FLAGS

DFG\$DEFRAGMENT returns the defragmentation process flags stored in the database for this script. The flag fields are described under the DFG\$K_DFG_FLAGS item descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—DFG\$\$_DFG_FLAGS.
- **buffer address**—Address of the buffer holding the DFG_FLAGS.

There is only one DFG\$K_SHOW_SCRIPT_DFG_FLAGS descriptor per script.

DFG\$K_SHOW_SCRIPT_DORMANT

DFG\$DEFRAGMENT returns a file specification. The file specification indicates a file (or files, if wildcard characters are present) that will be classified dormant by the defragmentation process. This file specification was originally part of the DFG\$K_CLASSIFY_DORM item descriptor. Wildcard characters and logicals are not resolved for this file specification.

Since the DFG\$K_SHOW_SCRIPT_DORMANT item descriptor describes only one file specification, DFG\$DEFRAGMENT returns multiple DFG\$K_SHOW_SCRIPT_DORMANT item descriptors if there are multiple dormant file specifications. If there are no dormant files associated with this script, DFG\$DEFRAGMENT does not return a DFG\$K_SHOW_SCRIPT_DORMANT item descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the file specification in bytes.
- **buffer address**—Address of the file specification.

DFG\$K_SHOW_SCRIPT_ENABLE

DFG\$DEFRAGMENT returns a word indicating whether the script is currently enabled or disabled as follows:

- DFG\$K_ENABLE—Script is enabled.

- **DFG\$K_DISABLE**—Script is disabled.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Must be two.
- **buffer address**—Address of a two-byte buffer containing a value of DFG\$K_ENABLE or DFG\$K_DISABLE.

There is only one DFG\$K_SHOW_SCRIPT_ENABLE item descriptor per script.

DFG\$K_SHOW_SCRIPT_EXCLUDE

DFG\$DEFRAGMENT returns a file specification. The file specification indicates a file that will be excluded by the defragmentation process. This file specification was originally part of the DFG\$K_EXCLUDE item descriptor. Wildcard characters and logicals are not resolved for this file specification.

Since the DFG\$K_SHOW_SCRIPT_EXCLUDE item descriptor describes only one file specification, DFG\$DEFRAGMENT returns multiple DFG\$K_SHOW_SCRIPT_EXCLUDE item descriptors if there are multiple exclude file specifications. If no excluded files are associated with this script, DFG\$DEFRAGMENT does not return a DFG\$K_SHOW_SCRIPT_EXCLUDE item descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the file specification in bytes.
- **buffer address**—Address of the file specification.

DFG\$K_SHOW_SCRIPT_FILE

DFG\$DEFRAGMENT returns the name of the target file or files for this script.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the target file-name in bytes.
- **buffer address**—Address of the target filename.

DFG\$K_SHOW_SCRIPT_FREQUENT

DFG\$DEFRAGMENT returns a file specification. The file specification indicates a file that will be classified frequent by the defragmentation process. This file specification was originally part of the DFG\$K_CLASSIFY_FREQ item descriptor. Wildcard characters and logicals are not resolved for this file specification.

Since the DFG\$K_SHOW_SCRIPT_FREQUENT item descriptor describes only one file specification, DFG\$DEFRAGMENT returns multiple DFG\$K_SHOW_SCRIPT_FREQUENT item descriptors if there are multiple frequent file specifications. If there are no frequent files associated with this script, DFG\$DEFRAGMENT does not return a DFG\$K_SHOW_SCRIPT_FREQUENT item descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the file specification in bytes.
- **buffer address**—Address of the file specification.

DFG\$K_SHOW_SCRIPT_INTERVAL

DFG\$DEFRAGMENT returns a VMS delta-time specification indicating the repeat interval for scripts that operate on a repeating schedule. If no interval was specified, DFG\$DEFRAGMENT does not return this descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the delta-time string.
- **buffer address**—Address of the delta-time string.

The DFG\$K_SHOW_SCRIPT_INTERVAL item descriptor can occur only once per script.

DFG\$K_SHOW_SCRIPT_LOG

DFG\$DEFRAGMENT returns the file specification of the log file used for this script. If the file specification contains logical names, they are not resolved. If no log file was specified with the script, DFG\$DEFRAGMENT does not return this item descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the log file string.
- **buffer address**—Address of the log file string.

The DFG\$K_SHOW_SCRIPT_LOG item descriptor can occur only once per script.

DFG\$K_SHOW_SCRIPT_NODE

DFG\$DEFRAGMENT returns a node name on which a defragmentation process associated with this script can execute. If DFG\$DEFRAGMENT does not return a DFG\$K_SHOW_SCRIPT_NODE item descriptor, the script may execute on any local node.

Since the DFG\$K_SHOW_SCRIPT_NODE descriptor describes only one node, DFG\$DEFRAGMENT returns multiple DFG\$K_SHOW_SCRIPT_NODE descriptors if execution of the script can occur on more than one node.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the node name string in bytes.
- **buffer address**—Address of the node name string.

DFG\$K_SHOW_SCRIPT_OPTIONS

DFG\$DEFRAGMENT returns the file specification of the options file, if any, associated with this script. If no options file is associated with this script, DFG\$DEFRAGMENT does not return a DFG\$K_SHOW_SCRIPT_OPTIONS item descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the options file name string in bytes.
- **buffer address**—Address of the options file name string.

DFG\$K_SHOW_SCRIPT_POST_PROC

DFG\$DEFRAGMENT returns the file specification of the postprocessing file, if any, associated with this script. If no postprocessing file is associated with this script, DFG\$DEFRAGMENT does not return a DFG\$K_SHOW_SCRIPT_POST_PROC item descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the postprocessing file name string in bytes.
- **buffer address**—Address of the postprocessing file name string.

DFG\$K_SHOW_SCRIPT_PRE_PROC

DFG\$DEFRAGMENT returns the file specification of the preprocessing file, if any, associated with this script. If no preprocessing file is associated with this script, DFG\$DEFRAGMENT does not return a DFG\$K_SHOW_SCRIPT_PRE_PROC item descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the preprocessing file name string in bytes.
- **buffer address**—Address of the preprocessing file name string.

DFG\$K_SHOW_SCRIPT_PREDECESSOR

DFG\$DEFRAGMENT returns the name of the predecessor to this script. If no predecessor exists, DFG\$DEFRAGMENT does not return a DFG\$K_SHOW_SCRIPT_PREDECESSOR item descriptor.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the predecessor's script name.
- **buffer address**—Address of the predecessor's script name.

The DFG\$K_SHOW_SCRIPT_PREDECESSOR item descriptor may occur only once per script shown.

DFG\$K_SHOW_SCRIPT_Q_ACTION

DFG\$DEFRAGMENT returns a word indicating what action is pending for this script. The following symbolic constants are defined:

- DFG\$K_ABORT—An abort operation is scheduled for this script.

- DFG\$K_CANCEL—A cancel operation is scheduled for this script.
- DFG\$K_REMOVE—A remove operation is scheduled for this script.
- DFG\$K_SCRIPT—This script is scheduled to execute.
- DFG\$K_SHOW_SCRIPT_Q_CANCEL— The script is currently canceled.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Must be two.
- **buffer address**—Address of the two-byte buffer containing the Q_ACTION code.

The DFG\$K_SHOW_SCRIPT_Q_ACTION item descriptor occurs once for every pending action. Thus, it may appear several times for a single script. The pending actions are returned in the order in which they will be executed.

DFG\$K_SHOW_SCRIPT_Q_REMAINING

DFG\$DEFRAGMENT returns a VMS delta-time string indicating how much time is left for the current execution of a script that was given a time limit (with the DFG\$K_TIME_LIMIT descriptor). If DFG\$DEFRAGMENT does not return a DFG\$K_SHOW_SCRIPT_Q_REMAINING item descriptor, no time limit is imposed on the script.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the delta-time specification in bytes.
- **buffer address**—Address of the delta-time specification.

DFG\$K_SHOW_SCRIPT_Q_START_TIME

DFG\$DEFRAGMENT returns a VMS absolute time indicating when the script will begin (1-minute intervals) if the script is scheduled to begin in the future, or it may indicate when the script started.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the time specification in bytes.
- **buffer address**—Address of the time specification.

The DFG\$K_SHOW_SCRIPT_Q_START_TIME item descriptor can occur only once per script.

DFG\$K_SHOW_SCRIPT_Q_STATUS

DFG\$DEFRAGMENT returns a word indicating the current abort status of the defragmentation process associated with the script. The following values may be returned:

- DFG\$K_ABORT_NOT_ISSUED—An abort operation is pending, but has not yet been issued.

- DFG\$K_ABORT_ISSUED—An abort command has been issued but the defragmentation process has not yet terminated.
- DFG\$K_ENGINE_ABORTED—The defragmentation process has aborted.
- DFG\$K_ENGINE_TERMINATED—The defragmentation process terminated on its own.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Must be two.
- **buffer address**—Address of the word containing the value of the abort status.

The DFG\$K_SHOW_SCRIPT_Q_STATUS item descriptor may occur once per queue entry.

DFG\$K_SHOW_SCRIPT_SCRIPT

DFG\$DEFRAGMENT returns the script name associated with this data. This item descriptor is the first one in the item list returned by DFG\$DEFRAGMENT. There is one DFG\$K_SHOW_SCRIPT_SCRIPT item descriptor for each script shown.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the script name string in bytes.
- **buffer address**—Address of the script name string.

The DFG\$K_SHOW_SCRIPT_SCRIPT item code descriptor may occur only once per script shown.

DFG\$K_SHOW_SCRIPT_SUCCESSOR

DFG\$DEFRAGMENT returns the name of the script that is the successor to this script. If no successor exists, DFG\$DEFRAGMENT does not return the DFG\$K_SHOW_SCRIPT_SUCCESSOR item code.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the successor script name in bytes.
- **buffer address**—Address of the successor script name.

DFG\$K_SHOW_SCRIPT_TIME_LIMIT

DFG\$DEFRAGMENT returns a VMS delta-time specification indicating the time limit imposed on this script, if any. If the DFG\$K_SHOW_SCRIPT_TIME_LIMIT item descriptor is absent, no time limit is established for this script.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the delta-time specification in bytes.
- **buffer address**—Address of the delta-time specification.

DFG\$K_SHOW_SCRIPT_VOLUME

DFG\$DEFRAGMENT returns the name of the target volume for this script.

The uses of the **buffer length** and **buffer address** fields of the ITEM_LIST_2 descriptor are as follows:

- **buffer length**—Length of the target volume name in bytes.
- **buffer address**—Address of the target volume name.

DFG\$K_START

The DFG\$K_START item descriptor starts the specified script.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_START item code are as follows:

- **buffer length**—Length of the script name in bytes.
- **buffer address**—Address of the script name string.
- **return length address**—Not used; must be zero.

DFG\$K_TIME_LIMIT

Indicates the duration of a given defragment operation. Specifying unreasonably small values (15-20 minutes or less) results in no useful defragmentation work.

The uses of the **buffer length**, **buffer address**, and **return length address** fields for the DFG\$K_TIME_LIMIT item code are as follows:

- **buffer length**—Length of the delta-time specification string in bytes.
- **buffer address**—Address of a character string buffer containing a delta-time specification. This buffer must be large enough to contain a VMS delta-time string as described in the *VMS Concepts Manual*.
- **return length address**—Not used; must be zero.

Return Values

DFG\$_ABOSEN

An ABORT command issued for this script had not yet completed when another ABORT command for the same script was issued.

DFG\$_DBUPDATERR

The command did not complete successfully.

DFG\$_EMPDFGSCH

Both the *sch_list* and the *dfg_list* parameters to DFG\$DEFRAGMENT were not present or referenced NULL item lists.

DFG\$_INTERR

DFG\$DEFRAGMENT internal error.

DFG\$_INVDFGLST

The *dfg_list* passed to DFG\$DEFRAGMENT was invalid, because of one of the following:

- A call to DFG\$DEFRAGMENT using an *sch_list* argument containing items that indicate the *dfg_list* should be empty; for example, a *sch_list* argument containing the DFG \$K_SHOW_SCRIPT item code may not be used in a call that has a *dfg_list* argument.
- Something was wrong with the semantics of the *dfg_list*, or a particular item descriptor was formatted improperly.

DFG\$_INVDSLST

Both the *sch_list* and the *dfg_list* parameters are invalid.

DFG\$_INVITEMCOMB

The DFG\$K_INTERVAL item descriptor is not allowed with the DFG\$K_PREDECESSOR item descriptor in a call to DFG\$DEFRAGMENT.

DFG\$_INVSCHLST

The scheduler list passed to DFG\$DEFRAGMENT is invalid.

DFG\$_MISSING_LOGICAL

A required logical name is not defined.

DFG\$_NOPRIV

The user had insufficient privileges to run the file defragmentation utility.

DFG\$_REMOVSEEN

A REMOVE subcommand issued for this script had not yet completed when another REMOVE subcommand for the same script was issued.

DFG\$_SCRDIS

Attempt to disable a script that was already disabled.

DFG\$_SCRENA

Attempt to enable a script that was already enabled.

DFG\$_STARTSEEN

A START command issued for this script had not yet completed when another START command for the same script was issued.

7.4. DFG\$RELEASE_MEMORY Routine

This section describes the DFG\$RELEASE_MEMORY routine.

DFG\$RELEASE_MEMORY routine

DFG\$RELEASE_MEMORY routine — The DFG\$RELEASE_MEMORY routine allows you to free memory allocated to you in response to a DFG\$DEFRAGMENT call to show a script (refer to the **DFG\$K_SHOW_SCRIPT** argument to the DFG\$DEFRAGMENT call). This call returns the address and length of the buffer created.

Format

DFG\$RELEASE_MEMORY **item_list**

Description

DFG\$RELEASE_MEMORY frees memory allocated to satisfy the call to show a script. This function has an item list interface with the structure shown in *Figure 7.5, "Function Item List Interface"*

Figure 7.5. Function Item List Interface

```
+-----+-----+-----+-----+
|      Item Code | Buffer Length      |
+-----+-----+-----+-----+
|      Buffer Address      |
+-----+-----+-----+-----+
|      Return Length Address      |
+-----+-----+-----+-----+
```

Returns

OpenVMS usage: cond_value
type: item_list_3_t
access: read only
mechanism: by reference

Arguments

Buffer Length

Must be zero.

Item Code

Indicates the code that was used when DFG\$\$UPDATE_DATABASE allocated memory. Disk File Optimizer software allocates memory only on DFG\$K_SHOW_SCRIPT item codes.

Buffer Length Address

Address of the longword that was written by DFG\$DEFRAGMENT when the memory was allocated. It contains the size of the buffer in bytes.

Appendix A. Answers to Frequently Asked Questions

This appendix contains frequently asked questions about the Disk File Optimizer(DFO).

If the system crashes, will DFO corrupt files?

No. DFO uses only published OpenVMS interfaces to the file system to accomplish its tasks. Data integrity is guaranteed by these interfaces. Further, the file optimizer never modifies nor accesses user data. If an on-going defragmentation operation is interrupted by a system crash, the volume may be left in a non-optimal state; that is, files may not be as defragmented as they could be and they may not be placed where they would have been placed had the file optimizer completed. However, when the system is brought back up, files on that volume are accessible and will not have suffered any integrity problems.

Does DFO change the modification date on the file?

No. No dates in the file header are changed.

How do I create a scheduling script?

Refer to *Section 2.6.2, "Creating Scheduling Scripts for Defrag PLUS"* for information on how to create scheduling scripts.

Will DFO run on my database disk (Rdb for example)?

Yes. If the database is open, DFO does not defragment the database file. If the database is closed, DFO treats the database file as any other file.

How do I check if the scheduler is running?

Use the following command to check if the scheduler is running on a node:

```
$ SHOW SYSTEM/OUTPUT=X.X
$ SEARCH X.X DFG
```

If the scheduler is running, you will see DFG\$node_name as shown below:

```
218002A8 DFG$LUNTIC          LEF 6 4091 0 00:01:03.21 1263    1781
```

How can I tell if a defragmentation process is running?

If you are using Defrag PLUS, you can use the command shown in *Example A.1, "Defrag PLUS Method"* to see if the defragmentation process is running:

Example A.1. Defrag PLUS Method

```
$ DEFRAGMENT SHOW */NAME/RUNNING
Disk File Optimizer for OpenVMS DFG V2.9
2005, VSI OpenVMS
Volume script DEFRAG_$1$DKB0 is ENABLED, LEVEL=2, device: $1$DKB0 ❶
Defragmentation process PID 2260153D is executing on NODE LUNTIC ❷
Scheduler queue contains the following operations:
    DEFRAGMENT VOLUME $1$DKB0 on 28-MAY-2003 17:00:00.00
Consolidate freespace
Script repeat interval: 1 00:00:00.00
```

```
Minimum Priority: 3
Default Priority: 4
Script may operate on the following nodes:
    LUNTIC
Log file: DISK$USER1:[DFG$DATABASE]DEFRAG_$1$DKB0.LOG;/FULL
No prologue file specified
No epilogue file specified
Mail notification is ONOPCOM notification is OFF
Special file handling:
    Exclude all specifically placed files
```

- ❶ This line shows the device being defragmented, \$1\$DKB0:
- ❷ This line shows the process ID, PID 2260153D, and the node LUNATIC

If you are using Defrag or Defrag PLUS, you can use the command shown in *Example A.2, "Defrag or Defrag PLUS Method"* to see if a defragmentation process is running:

Example A.2. Defrag or Defrag PLUS Method

```
$ SHOW SYSTEM/OUTPUT=X.X
$ SEARCH X.X DFG
226000AA DFG$LUNTIC      LEF  4 3550    0 00:00:12.75  924  271 ❶
2260153E DFG$ALPHA071   LEF  5  142    0 00:00:00.22  137  148 ❷
```

- ❶ This line shows that the scheduler is running.
- ❷ This line shows that a defragmentation process is running on DFG\$ALPHA071

How do I monitor the progress of a defragmentation process?

You can monitor the progress of a defragmentation job using the following command:

```
$ DEFRAGMENT MONITOR script_name or volume_name:
```

Refer to the MONITOR command in *Chapter 3, "Using Defrag"* or *Chapter 4, "Using Defrag PLUS"* for more information.

Can I defragment a RAID array?

Yes. A controller-created RAID array appears as a normal, but possible very large, logical volume and is defragmented as such. The logical volumes presented by the StorageWorks RAID Software for OpenVMS or POLYCENTER Disk Striping from RAID 5 arrays or stripesets may also be defragmented. When you create a script, specify the RAID array. Do not defragment individual members of the RAID array as they consist of one large file that is not fragmented and should not be moved.

Can I defragment a shadow set?

Yes. When you create a script, specify the DSAX: volume. Do not defragment individual members of the shadow set.

How do I know when it is time to defragment INDEXF.SYS.

There are two indications that INDEXF.SYS needs defragmenting:

- If you receive a "file header full" error, you should defragment INDEXF.SYS. This error means that INDEXF.SYS cannot extend any more without the file header extending into another header. The file header cannot have more than one header.

- If INDEXF.SYS has more than five extents, it is a candidate for defragmentation. By OpenVMS file system design, INDEXF.SYS has a minimum of four extents. By DFO product design, DFO will only defragment INDEXF.SYS if it has more than five extents.

Use the following command to check the number of extents in INDEXF.SYS:

```
$ DEFRAGMENT SHOW volume_name:/VOLUME/LOCATION=[000000] INDEXF.SYS
```

How do I do an offline defragmentation of INDEXF.SYS?

Refer to OFFLINE_VOLUME command in *Chapter 3, "Using Defrag"* or *Chapter 4, "Using Defrag PLUS"* for information on how to defragment INDEXF.SYS.

Can you defragment INDEXF.SYS on a system boot volume?

No. You have to dismount the disk before you can defragment INDEXF.SYS.

How long should I expect a defragmentation operation to take?

The time for a defragmentation operation to run depends on the following factors:

- How fragmented the volume is
- How much free space is available
- How busy the volume is with file creation and deletion

Many volumes can be defragmented in less than an hour. Some volumes can take more than 24 hours. The more the volume is fragmented, the more work Disk File Optimizer has to do. If this the first time the volume is defragmented, the defragmentation operation could take a long time. If the defragment processes are scheduled regularly, they will take less time.

How can I automatically cancel a script from happening on Sunday night when I do backups?

The following prolog file can be used to cancel the execution of a script on Sunday or modified to cancel the execution on any day of the week. The prolog file gets the script name from DFO so the same prolog file can be called from multiple scripts that need to be canceled on Sunday. You can place the file (CANCEL_SUNDAY.COM) into DFG\$DATABASE:CANCEL_SUNDAY.COM and then modify scripts so that they call the command procedure using the following command:

```
$ DEFRAGMENT MODIFY script_name/PROLOGUE=DFG$DATABASE:CANCEL_SUNDAY.COM
```

The comments in the command procedure describe how to change the cancel day from Sunday to another day. This script must be run on Saturday for it to cancel a Sunday run of the script.

```
$! command procedure to cancel the execution of a script on Sunday
$! this script must be run on Saturday so it can be cancelled on Sunday
$! The command procedure must be executed as a /PROLOGUE file from a
$! script
$!
$! get today
$!
$ today = f$cvtime("TODAY",, "WEEKDAY")
$!
$! if today is not Saturday, then get out
$!
$! to change the day, change Saturday to day of week before the
```

```
$! day that you want to cancel execution of the defragmentor
$!
$ if today .nes. "Saturday" then exit
$!
$! we want to cancel the next execution of DFG$SCRIPT
$! get script name
$!
$ script_name = f$trnln("DFG$SCRIPT")
$!
$! CANCEL the script so it will not execute tomorrow
$!
$ defragment cancel 'script_name'
$ exit
```

How can I automatically keep a script from executing on Sunday and Monday?

The following prolog file can be used to keep a script from execution on Sunday and Monday. The prolog file gets the script name from DFO so the same prolog file can be called from multiple scripts. You can place the file (ABORT_SUN_MON.COM) into DFG\$DATABASE:ABORT_SUN_MON.COM and then modify scripts so that they call the command procedure using the following command:

```
$ DEFRAGMENT MODIFY script_name/PROLOGUE=DFG$DATABASE:ABORT_SUN_MON.COM

$! command procedure to abort the execution of a script on Sunday
$! and Monday.
$!
$! The command procedure must be executed as a /PROLOGUE file from a
$! script
$!
$! get today
$!
$ today = f$cvtime("TODAY",, "WEEKDAY")
$!
$! if today is Sunday or Monday then abort the script
$!
$ if (today .eqs. "Sunday") .or. (today .eqs. "Monday")
$ then
$   script_name = f$trnln("DFG$SCRIPT")
$   defragment abort 'script_name'
$ endif
```

How can I truncate directories

Some users have directories with large allocated unused blocks after files have been deleted. This results in the need to truncate these directories. DFO does not provide this feature because it is already available using the OpenVMS DCL command SET FILE/TRUNCATE file_name.DIR.

The following shows an example using the SET FILE/TRUNCATE command to truncate a directory by deleting the unused blocks:

```
$ DIRECTORY/SIZE=ALL DIR/SIZE=ALL [SLICK]MAIL.DIR
Directory DSA422:[SLICK]
MAIL.DIR;1          115/140
$ SET FILE/TRUNCATE [SLICK]MAIL.DIR
$ DIR/SIZE=ALL [SLICK]MAIL.DIR
Directory DSA422:[SLICK]
MAIL.DIR;1          115/116
Total of 1 file, 115/116 blocks.
```

Notice in the above example that the directory is 140 blocks in size but the end of file is at block 115. The directory is then truncated to 116 blocks with the SET FILE/TRUNCATE command.

Appendix B. Disk File Optimizer

Logicals and Files

B.1. Files Added to System by the Disk File Optimizer

The following files are permanently added to the system when you install the Defrag PLUS variant. Files marked with an asterisk (*) are not added if you install the Defrag variant:

- DFG\$DATABASE (directory, typically SYS\$COMMON:[000000]DFG\$DATABASE.DIR) *
- DFG\$DATABASE:DFG\$DB.DAT *
- DFG\$DATABASE:DFG\$FILE.DAT *
- DFG\$DATABASE:DFG\$GENERAL.DAT *
- DFG\$DATABASE:DFG\$MAIL.DIS *
- DFG\$DATABASE:DFG\$NODE.DAT *
- DFG\$DATABASE:DFG\$QUEUE.DAT *
- DFG\$DATABASE:DFG\$SCRIPT.DAT *
- DFG\$DATABASE:DFG\$VOLUME.DAT *
- SYS\$COMMON:[SYS\$STARTUP]DFG\$STARTUP.COM
- SYS\$COMMON:[SYS\$STARTUP]DFG\$START_SCHEDULER.COM *
- SYS\$COMMON:[SYSEXEC]DFG\$CREATE_DATABASE.EXE *
- SYS\$COMMON:[SYSEXEC]DFG\$DCL.EXE
- SYS\$COMMON:[SYSEXEC]DFG\$ENGINE_MAIN.EXE
- SYS\$COMMON:[SYSEXEC]DFG\$INIT_DATABASE.EXE *
- SYS\$COMMON:[SYSEXEC]DFG\$SCHEDULER.EXE *
- SYS\$COMMON:[SYSEXEC]DFG\$STOP_SCHEDULER.EXE *
- SYS\$COMMON:[SYSEXEC]DFG\$REFORMAT_DB_RMS.EXE *
- SYS\$COMMON:[SYSEXEC]DFG\$REFORMAT_DB.EXE *
- SYS\$COMMON:[SYSHLP]DEFRAGMENT.HLB
- SYS\$COMMON:[SYSHLP]DFGxxx.RELEASE_NOTES
- SYS\$COMMON:[SYSHLP]DFGxxx_OPR_GUIDE
- SYS\$COMMON:[SYSHLP]DFGxxx_INSTALL_GUIDE
- SYS\$COMMON:[SYSHLP]DFGxxx_SPD

- SYS\$COMMON:[SYSHLP]DFGxxx_COVER_LETTER
- SYS\$COMMON:[SYSLIB]DFG\$DEFRAGMENTSHR.EXE *
- SYS\$COMMON:[SYSLIB]DFG\$RMSSHR.EXE *
- SYS\$COMMON:[SYSLIB]DFG\$UTILITYSHR.EXE
- SYS\$COMMON:[SYSMGR]DFG\$DEINSTALL.COM
- SYS\$COMMON:[SYSMGR]DFG\$SHUTDOWN.COM
- SYS\$COMMON:[SYSMGR]DFG\$STOP_SCHEDULER.COM *
- SYS\$COMMON:[SYSMSG]DFG\$MESSAGE.EXE
- SYS\$COMMON:[SYSTEST]DFG\$IVP.COM

If you install the GUI interface, the following additional files are created:

- SYS\$COMMON:[DECW\$DEFAULTS.USER]DFG\$DW.DAT
- SYS\$COMMON:[DECW\$DEFAULTS.USER]DFG\$DW.UID
- SYS\$COMMON:[SYSEXEC]DFG\$DW.EXE
- SYS\$COMMON:[SYSHLP]DFG\$DW.HLB

If you install the callable interface, the following additional files are created:

- SYS\$COMMON:[SYSLIB]DFG\$DEF.FOR
- SYS\$COMMON:[SYSLIB]DFG\$DEF.H
- SYS\$COMMON:[SYSLIB]DFG\$DEF.MAR
- SYS\$COMMON:[SYSLIB]DFG\$DEF.PAS

B.2. Logical Names

Logical names for the Disk File Optimizer begin with the prefix **DFG\$**. *Table B.1, "Logical Names"* describes each logical name.

Table B.1. Logical Names

Logical Name	Installation Variant	Purpose
DFG\$DATABASE	Defrag PLUS	Equates to the device:[dir] specification where the database resides. Also located in this directory are the scheduler log files, which have a name of DFG\$_xxxx.LOG, where xxxx is the name of the node on which the scheduler is running.
DFG\$DISK	All	Provided to the EPILOGUE and PROLOGUE command files. It translates to the name of the

Logical Name	Installation Variant	Purpose
		disk on which DFO currently operates.
DFG\$MAIL_ADDRESS	Defrag PLUS	Equates to a file name containing a mail distribution list of those who should receive notification when defragmentation processes complete.
DFG\$MESSAGE	All	Equates to SYS \$MESSAGE:DFG \$MESSAGE.EXE, which is the shareable image containing all the defragmentation-specific messages. This logical is used only if you install the callable interface.
DFG\$SCRIPT	All	Provided to the EPILOGUE and PROLOGUE command files. It translates to the name of the script associated with the DFO process.
DFG\$STATE	All	Provided to the EPILOGUE and PROLOGUE command files. As appropriate to indicate the state of DFO to the command procedure, it translates to either of these strings PROLOGUE EPILOGUE
DFG\$STATUS	All	Provided to the EPILOGUE and PROLOGUE command files. It translates to the current status of the DFO process. For the EPILOGUE command file, this status is the final status of the DFO process.
DFG\$ASTLM DFG\$BIOLM DFG\$BYTLM DFG\$DIOLM DFG\$ENQLM DFG\$FILLM	All	These logicals affect the associated process quota of any DFO process running on the associated node. Refer to <i>Section 4.3.5, "Defining Quotas for the Defragmentation Process"</i> for more information.

Logical Name	Installation Variant	Purpose
DFG\$JTQUOTA DFG\$PGFLQUOTA DFG\$PRCLM DFG\$TQELM DFG\$WSDEFAULT DFG\$WSEXTENT DFG\$WSQUOTA		
DFG\$TRANSFERSIZE	All	The logical, DFG \$TRANSFERSIZE, helps you tune the Disk File Optimizer. The definition of this logical allows you to change the amount of data that is transferred at one time. On a volume that has files with large extents, you could wait up to 20 minutes while the system transfers these large extents before you could access the file. By defining this logical, you could access this file in less than a minute. This logical changes the impact that the Disk File Optimizer has on applications that are simultaneously accessing the same files. This logical is set to 10,000 blocks by default. You can change it by editing SYS\$STARTUP:DFG \$STARTUP.COM and then executing the command procedure.

Appendix C. DEFRAGMENT SHOW Command Report

C.1. Introduction

This appendix describes the report you receive from a SHOW command of a volume for the Disk File Optimizer (DFO). This appendix describes the meaning of each line in the report.

C.2. Output from the SHOW Command

The following shows the output of the SHOW command with explanation notes:

```
$ DEFRAGMENT SHOW DISK$ALPHA070:/VOLUME
Disk File Optimizer for OpenVMS DFG V2.9
2005, VSI OpenVMS

                                F r a g m e n t a t i o n       R e p o r t
DISK$ALPHA070                                05-JAN-2005 16:14:10.93
The fragmentation index is 17.2
    1 - 20.9 is excellent
    21 - 40.9 is good
    41 - 60.9 is fair
    61 - 80.9 is poor
    81 - 100 indicates a badly fragmented disk
Approximately 12.6 (out of 80.0 possible) is due to file fragmentation
Approximately 4.6 (out of 20.0 possible) is due to freespace fragmentation
Freespace Summary:
    Total free space:                211072 blocks
    Percentage free:                  10 (rounded)
    Total free extents:               23
    Maximum free extent:              53316 blocks, LBN: 671852
    Minimum free extent:              4 blocks, LBN: 730360
    Average free extent:              9177 blocks
    Median free extent:               584 blocks
File Fragmentation Summary:
    Number of files (with some allocation): 5880
    Total file extents on the disk:      6366
    Average number of file extents per file: 1.082653
    Median number of file extents per file: 1
Most Fragmented File:
    [VMS$COMMON.SYSEXEC]SYS$QUEUE_MANAGER.QMAN$JOURNAL;1 (171 extents)
```

The following explains the report above:

- ❶ The fragmentation index is the relative index of the fragmentation state of the volume. The index is made up of file fragmentation, 80%, and free space fragmentation, 20%. The value of 17.2 is an excellent fragmentation index.

The file fragmentation portion is a number based on how fragmented the files are. If all files were contiguous, this number would be 0.0.

The free space fragmentation is a number based on how fragmented the free space is. If all free space could be consolidated into one area, this number would be 0.0. As files are deleted or moved, the free space becomes fragmented.

Refer to *Section C.4, "Calculation of Fragmentation Index"* for an explanation of how the fragmentation index is calculated.

- ❷ The total free space value in 512 byte blocks on the volume. This is the amount of free space that you see in response to the DCL command SHOW DEVICE.
- ❸ The percentage of free space compared to the total size of the volume. The 10% free space on this volume means it needs to be cleaned up or some files moved off. With only 10% free space, the volume will tend to fragment faster and defragmentation will take longer.
- ❹ The total free extents is the number of areas the free space is fragmented into. The ideal number would be one. However, this number usually cannot be achieved because of files that cannot be moved. These files create areas on the volume. If no files are small enough to fit, the volume is left with some free space.
- ❺ The maximum free extent is the number of blocks in the biggest free space area. The ideal situation would be for this number to be the same as total free space.
- ❻ The minimum free extent is the number of blocks in the smallest free space area.
- ❼ The average free extent is the average number of blocks of free space extents.
- ❽ The median free extent is the median number of blocks of the free space extent. DFO calculates the median by taking the size of all free extents, sorting them, and then selecting the size of free extent in the middle.
- ❾ The number of files on the volume with some allocation. Files with zero allocation are not counted in this number.
- ❿ The total file extents on the volume.
- ⓫ The average number of files extents per file is the total file extents on the disk divided by the number of files (with some allocation). The lower this number, the less fragmented the volume.
- ⓬ The median number of file extents per file is the median number of file extents on the volume. DFO calculates the median by taking the number of extents for each file, sorting them, and then selecting the number in the middle.
- ⓭ The most fragmented file name and the number of extents it contains.

C.3. Examples of the Output from the SHOW Command

This section has three example outputs of the SHOW command after each volume has been defragmented. The examples are on the following volumes:

System volume
User volume with system log files on it
User volume

C.3.1. System Volume

Example C.1, "System Volume" shows an output from the SHOW command for a system volume.

Example C.1. System Volume

```

$ DEFRAGMENT SHOW $1$DKB0:/VOLUME/HISTOGRAM
Disk File Optimizer for OpenVMS DFG V2.9
2005, VSI OpenVMS

                        F r a g m e n t a t i o n       R e p o r t
DISK$ALPHA071                                05-JAN-2005 11:16:42.10
The fragmentation index is 13.6
    1 - 20.9 is excellent
    21 - 40.9 is good
    41 - 60.9 is fair
    61 - 80.9 is poor
    81 - 100 indicates a badly fragmented disk
Approximately 9.2 (out of 80.0 possible) is due to file fragmentation
Approximately 4.4 (out of 20.0 possible) is due to freespace
fragmentation
Freespace Summary:
    Total free space:          148076 blocks
    Percentage free:           7 (rounded)
    Total free extents:        22
    Maximum free extent:       30312 blocks, LBN: 1799744
    Minimum free extent:       4 blocks, LBN: 1022776
    Average free extent:       6730 blocks
    Median free extent:        1556 blocks
File Fragmentation Summary:
    Number of files (with some allocation): 8398
    Total file extents on the disk: 8947
    Average number of file extents per file: 1.065373
    Median number of file extents per file: 1
Most Fragmented File:
    [SYSTEM_FILES.SPIELN]ACCOUNTNG.DAT;7 (110 extents)

```

Example C.2. System Volume: File Fragmentation Histogram

```

                        F i l e       F r a g m e n t a t i o n       H i s t o g r a m
ExtentCount
-----
    |  9 To 110 |  (5)
7 |  (2)
6 |  (3)
5 |  (6)
4 |  (12)
3 |  (28)
2 |  * (158)
1 |  ***** (8184)
+-----+
    Number of files with a given number of extents
    Each * corresponds to 200 files

```

Example C.3. System Volume: Volume Free Space Histogram

```

                        V o l u m e       F r e e s p a c e       H i s t o g r a m
FreespaceSize (LBNs)
-----
    |
30312 | * (1)
26724 | * (1)
21764 | * (1)

```

```
16536 | * (1)
15784 | * (1)
15744 | * (1)
 7048 | * (1)
 4044 | * (1)
 3808 | * (1)
 3672 | * (1)
 1556 | * (1)
  584 | * (1)
  308 | * (1)
   48 | * (1)
   40 | ** (2)
   32 | * (1)
    8 | *** (3)
    4 | ** (2)
+-----
Number of freespace extents of a given size
Each * corresponds to 1 freespace extent
```

The following are numbers to note:

- ❶ Fragmentation index – the 13.6 is an excellent number considering all of the non-movable files on a system disk. Non-movable files are those installed files, those files marked NOMOVE, and those files that are open.
- ❷ File fragmentation – the 9.2 number is caused by files like the most fragmented file with 110 extents. You cannot tell by this report, but this file is open and DFO cannot defragment it.
- ❸ Freespace fragmentation – the 4.4 number is not bad for the system volume. Notice that the total free extents are 22. These 22 extents are caused by all of the non-movable files on the system volume.
- ❹ Percentage free – the 7% free space is low. With only 7% free space, the volume will tend to fragment faster and defragmentation will take longer. It is time to clean up files or add another volume.
- ❺ Total free extents – the 22 free extents is a fairly good number. This small number makes the free space fragmentation number small also.
- ❻ Average free and median free extents – the 6730 average and 1556 median shows the free space extents are not clustered around 6730. They are spread out and tend to be more extents around the low end. This is shown in *Example C.3, "System Volume: Volume Free Space Histogram"*.
- ❼ Average number of file extents per file – the 1.065 number is excellent.

In conclusion, this volume is in great shape except it is very full.

C.3.2. User Volume With System Log Files

Example C.4, "User Volume With System Log Files" shows an output from the SHOW command for a user volume with system log files on it.

Example C.4. User Volume With System Log Files

```
$ DEFRAGMENT SHOW $3$DUA1:/VOLUME/HISTOGRAM
```

```

Disk File Optimizer for OpenVMS DFG V2.9
2005, VSI OpenVMS

                        F r a g m e n t a t i o n       R e p o r t
DISK$USER_DISK                                05-JAN-2005 11:42:51.80
The fragmentation index is 83.4
    1 - 20.9 is excellent
    21 - 40.9 is good
    41 - 60.9 is fair
    61 - 80.9 is poor
    81 - 100 indicates a badly fragmented disk
Approximately 63.4 (out of 80.0 possible) is due to file fragmentation
Approximately 20.0 (out of 20.0 possible) is due to freespace
fragmentation
Freespace Summary:
    Total free space:          104613 blocks
    Percentage free:           4 (rounded)
    Total free extents:        303
    Maximum free extent:       13980 blocks, LBN: 2057343
    Minimum free extent:       3 blocks, LBN: 541440
    Average free extent:       345 blocks
    Median free extent:        27 blocks
File Fragmentation Summary:
    Number of files (with some allocation): 11145
    Total file extents on the disk: 21649
    Average number of file extents per file: 1.942485
    Median number of file extents per file: 1
Most Fragmented File:
    [SYSEXEC]$SYS$QUEUE_MANAGER.QMAN$JOURNAL;1 (8179 extents)

```

Example C.5. User Volume With System Log Files: File Fragmentation Histogram

```

                        F i l e       F r a g m e n t a t i o n       H i s t o g r a m
ExtentCount
-----
    23 To 8179 | (3)
              7 | (7)
              6 | (31)
              5 | (77)
              4 | * (152)
              3 | ** (315)
              2 | *** (658)
              1 | ***** (9902)
              +-----
                  Number of files with a given number of extents
                  Each * corresponds to 200 files

```

Example C.6. User Volume With System Log Files: Volume Free Space Histogram

```

                        V o l u m e       F r e e s p a c e       H i s t o g r a m
FreespaceSize (LBNs)
-----
    171 | * (1)
    165 | ** (2)
    162 | * (1)
    159 | * (1)
    156 | * (1)
    153 | ** (2)

```

```

144 | * (1)
141 | ** (2)
138 | ** (2)
135 | ** (2)
132 | * (1)
126 | *** (3)
123 | * (1)
120 | * (1)
117 | * (1)
114 | *** (3)
111 | **** (4)
108 | * (1)
105 | ** (2)
102 | * (1)
 99 | **** (4)
 96 | *** (3)
 90 | * (1)
 87 | * (1)
 84 | *** (3)
 81 | * (1)
 78 | ** (2)
 75 | * (1)
 72 | *** (3)
 69 | ** (2)
 66 | *** (3)
 63 | ** (2)
 60 | * (1)
 57 | ***** (6)
 54 | *** (3)
 51 | * (1)
 48 | *** (3)
 45 | ***** (5)
 42 | ** (2)
 39 | ***** (8)
 36 | ***** (7)
 33 | **** (4)
 30 | ***** (7)
 27 | ***** (8)
 24 | ***** (11)
 21 | ***** (9)
 18 | ***** (11)
 15 | ***** (15)
 12 | ***** (28)
  9 | ***** (15)
  6 | ***** (27)
  3 | ***** (35)
-----+-----
      Number of freespace extents of a given size
      Each * corresponds to 1 freespace extent

```

The following are numbers to note:

- ❶ Fragmentation index – the 83.4 indicates this volume is badly fragmented. But this is to be expected with this volume with large open fragmented log files and very little free space.
- ❷ File fragmentation – the 63.4 number appears to be bad, but for this volume, it cannot be improved. Not shown by this report, the volume has large open fragmented files. These log files were moved to this volume to get them off the system disk. For instance, the most fragmented file

has 8179 extends. This file is also open and very large (not shown by this report). These factors make it impossible to get the fragmentation index down.

- ③ Freespace fragmentation – the 20.0 number cannot be larger. Notice the 303 total free extents. These numbers are caused by these log files as explained for file fragmentation.
- ④ Percentage free – the 4% free space is very low. With only 4% free space, the volume will tend to fragment faster and defragmentation will take longer.
- ⑤ Total free extents – the 303 free extents is a large number. This large number makes the free space fragmentation large also.
- ⑥ Average free and median free extents – the 345 average and 27 median shows the free space extents are not clustered around 345. They are spread out and tend to be more extents around the low end. This is shown in the *Example C.6, "User Volume With System Log Files: Volume Free Space Histogram"*.
- ⑦ Average number of file extents per file – the 1.942 number reflects what you see in the file fragmentation index.

In conclusion, this volume is defragmented as well as it can be given the type of files on this volume. The fragmentation index of this volume could be improved if the open files could be closed and defragmented. Also, the free space could be improved by removing some files.

C.3.3. User Volume

Example C.7, "User Volume" shows an output from the SHOW command for a user volume.

Example C.7. User Volume

```
$ DEFRAGMENT SHOW DISK$USR5:/VOLUME/HISTOGRAM
Disk File Optimizer for OpenVMS DFG V2.9
2005, VSI OpenVMS
```

F r a g m e n t a t i o n R e p o r t

```
DISK$USR5                                05-JAN-2005 14:56:54.02
The fragmentation index is 1.5
```

①

```
    1 - 20.9 is excellent
    21 - 40.9 is good
    41 - 60.9 is fair
    61 - 80.9 is poor
    81 - 100 indicates a badly fragmented disk
Approximately 0.8 (out of 80.0 possible) is due to file fragmentation ②
Approximately 0.7 (out of 20.0 possible) is due to freespace
fragmentation ③
```

Freespace Summary:

```
    Total free space:      4744809 blocks
    Percentage free:       56 (rounded)
    Total free extents:    10
    Maximum free extent:   2522745 blocks, LBN: 1666278
    Minimum free extent:   9 blocks, LBN: 2655
    Average free extent:   474480 blocks
    Median free extent:    18 blocks
```

File Fragmentation Summary:

```
    Number of files (with some allocation): 17835
    Total file extents on the disk:         17838
    Average number of file extents per file: 1.000168
```

⑦

```

    Median number of file extents per file: 1
Most Fragmented File:
    [000000]INDEXF.SYS;1 (4 extents)

```

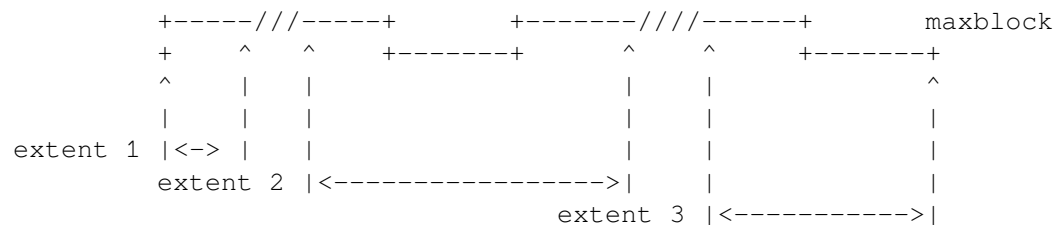
In conclusion, this volume is in great shape.

C.4. Calculation of Fragmentation Index

The fragmentation index is a value that DFO assigns to a volume. The fragment index is made up of two components:

- 80% of the index is file fragmentation
- 20% of the index is free space fragmentation

To explain how the index for free space fragmentation is calculated, you must understand how move area extents are calculated. The following diagram has three move area extents. These are areas between files that cannot be moved. In this example, let each dash (-) represent a block. You can see that the 2 free spaces of 7 blocks each (14 blocks) can fit in the extent 2. However, right now the free space is in 2 extents. Using the algorithm below, you will find that this diagram has 2 extents on the disk, and the best it can do is 1. Therefore, the free space index is $2/1 * .2 = .4$.



where: the upper line is allocated space
the lower line is free space
/// indicates a file that cannot be moved
The following types of files cannot be moved:

1. Reserved system file
2. File is set NOMOVE
3. File is a directory
4. Indexed file and indexed files cannot be moved
5. Placed file and placed files cannot be moved
6. Files excluded by the user

The following algorithm explains how the file portion and freespace portions are calculated:

```

files_larger_than_a_cluster = 0
weighted_fragged_files = 0
do for each file that is larger than a cluster
  begin
    files_larger_than_a_cluster = files_larger_than_a_cluster + 1
    weighted_fragged_files = weighted_fragged_files +
      extents_for_this_file + MAX(0, (extents_for_this_file - window_size))
  end
file_portion = (100 - (100 *
  (files_larger_than_a_cluster /
    weighted_fragged_files))) * .8
calculate each move_area_extent
temp = total_freespace
i = 0
do for each move_area_extent entry or temp goes to zero
  i = i + 1
  temp = temp - minimum(move_area_extent[i], temp)
  ! i represents the best we can do given the files

```

```
! we can not move
freespace_portion = (freespace_extents / i) * .2
defragment_index = file_portion + freespace_portion
```

Appendix D. Brief Log File Example

Example D.1, "Brief Log File" shows an example of a brief log file.

Example D.1. Brief Log File

```
Disk File Optimizer for OpenVMS DFG V2.9
Time: 05-JAN-2005
12:57:10.82
Process ID: 2100013D
Node: SPIELN
Run time parameters:
  Defragmentation level: 2
  Defragment placed files: No
  Defragment indexed files: Yes
  Log file report: Brief
  Notify user by mail: No
  Notify user by opcom: Yes
  Default priority: 4
  Minimum priority: 3
  Query hotfile database: No
  RVN to defragment: 0
  Consolidate freespace: Yes
  Perform write check: No

Defragmentation process start
Process ID: 2100013D
Device: _$1$DKA0:
Time: 05-JAN-2005 12:57:11.38
  Volume name: _$1$DKA0:
  Media name: DKX00
  Cluster factor: 4
  Number of blocks: 4110480
Begin file evaluation phase: 05-JAN-2005
12:57:11.91
  Candidates for defragmentation (based on the defragmentation level):
Begin free space consolidation: 05-JAN-2005
12:57:14.86
There are 436 consolidation areas
  IN USE DISK$AXP62-1H2:[VMS$COMMON.SYSLIB]LIB.MLB;2 (1 extent)
  OPEN DISK$AXP62-1H2:[AMEND]DECW$SM.LOG;5
End free space consolidation: 05-JAN-2005
13:12:08.95
Completion data for volume:
Time: 05-JAN-2005
13:12:09.05
  Statistics:
    Total number of files evaluated: 0
    Total number of candidates found: 0
    Total number of files defragmented: 0
    Total number of files partly defragmented: 0
    Total number of files placed frequent: 0
    Total number of files placed dormant: 0
```

Accounting information

Buffered I/O count:	6354
Direct I/O count:	50932
Page faults:	260
Peak working set:	5072
Charged CPU time:	00:00:46.78
Elapsed time:	0 00:14:57.69

Defragmentation process normal completion

Process ID: 2100013D

Device: _\$1\$DKA0:

Time: 05-JAN-2005 13:12:09.68

Appendix E. Full Log File Example

Example E.1, "Full Log File" shows an example of a full log file.

Example E.1. Full Log File

```
Disk File Optimizer for OpenVMS DFG V2.9
  Time:                                05-JAN-2005
14:44:08.70
  Process ID:                          20E0059A
  Node:                                SPIELN
  Run time parameters:
    Defragmentation level:              2
    Defragment placed files:            No
    Defragment indexed files:           Yes
    Log file report:                    Full
    Notify user by mail:                 No
    Notify user by opcom:                Yes
    Default priority:                    4
    Minimum priority:                    3
    Query hotfile database:              No
    RVN to defragment:                   0
    Consolidate freespace:                Yes
    Perform write check:                 No
  Defragmentation process start
Process ID: 20E0059A
Device: _$1$DKB0:
Time: 05-JAN-2005 14:44:09.24
  Volume name:        _$1$DKB0:
  Media name:          DKX00
  Cluster factor:      3
  Number of blocks:    2050860
Begin file evaluation phase:                                05-JAN-2005
14:44:12.71
  Files specified for dormant placement:
    DISK$JIM: [DFO_TEST_1]DFO_TEST_1_S.TXT;42
    DISK$JIM: [DFO_TEST_1]DFO_TEST_1_S.TXT;4
    DISK$JIM: [DFO_TEST_1]DFO_TEST_1_MS.TXT;41
    DISK$JIM: [DFO_TEST_1]DFO_TEST_1_S.TXT;41
    DISK$JIM: [DFO_TEST_1]DFO_TEST_1_MS.TXT;3
  Files specified for frequent placement:
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_MS.TXT;15
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_S.TXT;7
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_S.TXT;15
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_MS.TXT;2
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_S.TXT;2
  Candidates for defragmentation (based on the defragmentation level):
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_MS.TXT;2 (34 extents)
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_MS.TXT;6 (34 extents)
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_L.TXT;4 (1107 extents)
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_MS.TXT;1 (34 extents)
    DISK$JIM: [DFO_TEST_10]DFO_TEST_10_L.TXT;1 (3321 extents)
End file evaluation phase:                                05-JAN-2005
14:45:06.04
  Statistics:
    Number of files evaluated:            886
```

```

Number of candidates for defragmentation:      668
(Does not include dormant and frequent files)
Number of frequent files in list:              108
Number of frequent-file candidates:           59
Number of dormant files in list:              110
Number of dormant-file candidates:           60
Candidates evaluated by category:
    Indexed files:                             0
    Files with fixed placement:                 0
    Contiguous files:                          99
    Less than or equal to window turn:         0
    Less than or equal to one header:          443
    Greater than one header:                   225
Begin file defragmentation phase one:          05-JAN-2005
14:45:06.68
Files processed during defragmentation phase one:
Symbols preceding a file name indicate the following:
    Blank   - file has been preprocessed for defragmentation
    d       - file has been defragmented
    *       - file has been partly defragmented
    D       - file has been placed for dormant access
    Dd      - file has been defragmented and placed dormant
    D*      - file has been placed dormant and been partly defragmented
    -D      - file could not be placed dormant
    F       - file has been placed for frequent access
    Fd      - file has been defragmented and placed frequent
    F*      - file has been placed frequent and been partly
defragmented
    -F      - file could not be placed frequent
    Moved   - file has been moved (number of extents unchanged)
    EMPTY   - file was empty; no action taken
    FAIL    - file's fragmentation cannot be reduced
    IN USE  - new disk space for file was in use; no action taken
    OPEN    - file was open; no action taken
    NOMOVE  - file may not be moved; no action taken
    UNAVL   - file was unavailable; no action taken
d    DISK$JIM: [DFO_TEST_6]DFO_TEST_6_MS.TXT;9 (1 extent)
d    DISK$JIM: [DFO_TEST_5]DFO_TEST_5_MS.TXT;9 (1 extent)
d    DISK$JIM: [DFO_TEST_3]DFO_TEST_3_MS.TXT;11 (1 extent)
d    DISK$JIM: [DFO_TEST_3]DFO_TEST_3_MS.TXT;35 (1 extent)
d    DISK$JIM: [DFO_TEST_9]DFO_TEST_9_MS.TXT;9 (1 extent)
d    DISK$JIM: [DFO_TEST_4]DFO_TEST_4_MS.TXT;11 (1 extent)
End file defragmentation phase one:          05-JAN-2005 14:47:30.23
Statistics:
    Number of files processed:                  116
    Number of files defragmented:               116
    Number of files partly defragmented:        0
    Number of frequent files placed:            0
    Number of dormant files placed:            0
Begin file defragmentation phase two:          05-JAN-2005
14:47:30.53
Files processed during defragmentation phase two:
*    DISK$JIM: [DFO_TEST_5]DFO_TEST_5_L.TXT;3 (6 extents)
*    DISK$JIM: [000000]L.TXT;1 (6 extents)
*    DISK$JIM: [DFO_TEST_3]DFO_TEST_3_L.TXT;3 (3 extents)
d    DISK$JIM: [DFO_TEST_4]DFO_TEST_4_L.TXT;3 (1 extent)
d    DISK$JIM: [DFO_TEST_7]DFO_TEST_7_L.TXT;3 (1 extent)
d    DISK$JIM: [DFO_TEST_2]DFO_TEST_2_L.TXT;10 (1 extent)

```

```

End file defragmentation phase two:                                05-JAN-2005
18:33:36.08
  Statistics:
    Number of files processed:                                     433
    Number of files defragmented:                                 368
    Number of files partly defragmented:                         65
    Number of frequent files placed:                             0
    Number of dormant files placed:                              0
Begin file defragmentation phase three:                           05-JAN-2005
18:33:36.34
  Files processed during defragmentation phase three:
    d    DISK$JIM:[DFO_TEST_5]DFO_TEST_5_L.TXT;3 (1 extent)
    d    DISK$JIM:[000000]L.TXT;1 (1 extent)
  FAIL   DISK$JIM:[DFO_TEST_3]DFO_TEST_3_L.TXT;3 (2 extents)
    d    DISK$JIM:[DFO_TEST_5]DFO_TEST_5_L.TXT;10 (1 extent)
    d    DISK$JIM:[DFO_TEST_5]DFO_TEST_5_L.TXT;9 (1 extent)
End file defragmentation phase three:                             05-JAN-2005 10:37:05.78
  Statistics:
    Number of files processed:                                     65
    Number of files defragmented:                                 62
    Number of files partly defragmented:                         0
    Number of frequent files placed:                             0
    Number of dormant files placed:                              0
Begin frequent file placement:                                    05-JAN-2005
10:37:06.10
  Files processed during frequent file placement:
    Fd    DISK$JIM:[DFO_TEST_10]DFO_TEST_10_L.TXT;3 (1 extent)
    Fd    DISK$JIM:[DFO_TEST_10]DFO_TEST_10_L.TXT;2 (1 extent)
    F     DISK$JIM:[DFO_TEST_10]DFO_TEST_10_L.TXT;9 (1 extent)
    Fd    DISK$JIM:[DFO_TEST_10]DFO_TEST_10_L.TXT;8 (1 extent)
End frequent file placement:                                     05-JAN-2005 10:52:44.66
  Statistics:
    Number of files processed:                                     100
    Number of files defragmented:                                 0
    Number of files partly defragmented:                         0
    Number of frequent files placed:                             100
    Number of dormant files placed:                              0
Begin dormant file placement:                                    05-JAN-2005
10:52:44.90
  Files processed during dormant file placement:
    Dd    DISK$JIM:[DFO_TEST_1]DFO_TEST_1_L.TXT;10 (1 extent)
    Dd    DISK$JIM:[DFO_TEST_1]DFO_TEST_1_L.TXT;3 (1 extent)
    Dd    DISK$JIM:[DFO_TEST_1]DFO_TEST_1_L.TXT;6 (1 extent)
    Dd    DISK$JIM:[DFO_TEST_1]DFO_TEST_1_L.TXT;9 (1 extent)
End dormant file placement:                                     05-JAN-2005
11:06:57.49
  Statistics:
    Number of files processed:                                     100
    Number of files defragmented:                                 0
    Number of files partly defragmented:                         0
    Number of frequent files placed:                             0
    Number of dormant files placed:                              100
  Begin free space consolidation:                                05-JAN-2005
11:06:57.76
There are 9 consolidation areas
Beginning consolidation pass on area 8, LBN 971163 (51 blocks)
Beginning consolidation pass on area 7, LBN 1015065 (96 blocks)
Beginning consolidation pass on area 6, LBN 6 (741 blocks)

```

```

Beginning consolidation pass on area 5, LBN 868650 (1755 blocks)
Beginning consolidation pass on area 4, LBN 880302 (5283 blocks)
Beginning consolidation pass on area 3, LBN 870408 (9891 blocks)
Beginning consolidation pass on area 2, LBN 885588 (34431 blocks)
Beginning consolidation pass on area 1, LBN 1282092 (643566 blocks)
Beginning consolidation pass on area 0, LBN 750 (867897 blocks)
End free space consolidation:                                05-JAN-2005
11:20:27.04
Begin file defragmentation phase one:                        05-JAN-2005
11:20:27.09
Files processed during defragmentation phase one:
Symbols preceding a file name indicate the following:
Blank   - file has been preprocessed for defragmentation
d       - file has been defragmented
*       - file has been partly defragmented
D       - file has been placed for dormant access
Dd      - file has been defragmented and placed dormant
D*      - file has been placed dormant and been partly defragmented
-D      - file could not be placed dormant
F       - file has been placed for frequent access
Fd      - file has been defragmented and placed frequent
F*      - file has been placed frequent and been partly
defragmented
-F      - file could not be placed frequent
Moved   - file has been moved (number of extents unchanged)
EMPTY   - file was empty; no action taken
FAIL    - file's fragmentation cannot be reduced
IN USE  - new disk space for file was in use; no action taken
OPEN    - file was open; no action taken
NOMOVE  - file may not be moved; no action taken
UNAVL   - file was unavailable; no action taken
d       DISK$JIM:[DFO_TEST_3]DFO_TEST_3_L.TXT;3 (1 extent)
d       DISK$JIM:[000000]DFO_TEST_4_L.TXT;1 (1 extent)
d       DISK$JIM:[DFO_TEST_3]DFO_TEST_3_L.TXT;2 (1 extent)
End file defragmentation phase one:                          05-JAN-2005 11:20:47.38
Statistics:
Number of files processed:                                3
Number of files defragmented:                             3
Number of files partly defragmented:                      0
Number of frequent files placed:                          0
Number of dormant files placed:                           0
Begin free space consolidation:                             05-JAN-2005
11:20:47.63
There are 9 consolidation areas
Beginning consolidation pass on area 1, LBN 1282092 (643566 blocks)
OPEN    DISK$JIM:[000000]DFO_TEST_4.DIR;1
OPEN    DISK$JIM:[000000]DFO_TEST_5.DIR;1
OPEN    DISK$JIM:[000000]DFO_TEST_6.DIR;1
OPEN    DISK$JIM:[000000]DFO_TEST_7.DIR;1
OPEN    DISK$JIM:[000000]DFO_TEST_8.DIR;1
OPEN    DISK$JIM:[000000]DFO_TEST_9.DIR;1
OPEN    DISK$JIM:[000000]DFO_TEST_10.DIR;1
OPEN    DISK$JIM:[000000]DFO_TEST_2.DIR;1
OPEN    DISK$JIM:[000000]DFO_TEST_1.DIR;1
Beginning consolidation pass on area 0, LBN 750 (867897 blocks)
End free space consolidation:                                05-JAN-2005
11:37:43.10
Completion data for volume:

```

Time: 05-JAN-2005
11:37:43.19
Statistics:
Total number of files evaluated: 886
Total number of candidates found: 668
Total number of files defragmented: 549
Total number of files partly defragmented: 65
Total number of files placed frequent: 100
Total number of files placed dormant: 100
Accounting information
Buffered I/O count: 186560
Direct I/O count: 3902971
Page faults: 5638
Peak working set: 30032
Charged CPU time: 01:10:27.54
Elapsed time: 0 20:53:33.93
Defragmentation process normal completion
Process ID: 20E0059A
Device: _\$1\$DKB0:
Time: 05-JAN-2005 11:37:43.71

Appendix F. Defragmentation Messages

F.1. General Message Information

Most messages displayed by the VMS operating system contain a facility code, a severity indicator, an identification field, and the actual text of the message. They have the following format:

```
%FACILITY-L-IDENT, text
```

Where:

FACILITY indicates which VMS facility or layered product is reporting the message.

L indicates how severe an error is being reported. *Table F.1, "Severity Indicators"* describes the severity indicators that may appear in this field.

IDENT is a mnemonic code that uniquely identifies the error within the reported facility.

text is the text of the message.

Table F.1. Severity Indicators

Code	Meaning
S	Success—These messages indicate that the command is successful. In some cases, the command processing continues after issuing a success message. In other cases, the message indicates that the command processing is complete.
I	Information—These messages provide information about unique events that occur during processing. They do not indicate errors in processing.
W	Warning—These messages indicate that some, but not all, of your request is error free. You should verify the command or program output.
E	Error—Error messages indicate that the output or program result is incorrect, but the system may attempt to continue execution.
F	Fatal—Fatal messages indicate that the system cannot continue execution.

F.1.1. Linked Messages

Frequently, more than one message occurs at a time. A series of messages can help you determine the location of an error. Linked messages usually display the return status of the different facilities called and provide a detailed explanation of the error.

F.1.2. Recovery Procedures

Error messages generally occur for the following reasons:

- A command is entered incorrectly or contains a spelling or syntax error or an incorrect file specification.
- A required resource is currently unavailable.
- A programming error is detected during the execution of an image.
- A system software or hardware problem exists.

F.1.3. Recovering from Command Entry Errors

When an error occurs following interactive command entry, you must reenter the command. Messages from the command interpreter are usually followed by a display of the portion of the command line that is rejected. Correct this portion of the line before you reenter the command.

Errors that occur while a command is executing may indicate that you have entered an invalid value for a parameter or qualifier. In these cases, you can use the DCL command `HELP DEFRAGMENTATION` or consult *Chapter 3, "Using Defrag"* or *Chapter 4, "Using Defrag PLUS"* to determine legal values before you reenter the command.

F.1.4. Reporting System Software or Hardware Errors

When a message notifies you that a system software or hardware error has occurred, action is required by the system operator or system manager.

The *Introduction to VMS System Management Manual* provides information on how to submit an SPR to VSI when you encounter a software error.

F.2. Disk File Optimizer Messages

This section lists and describes all messages generated by the Disk File Optimizer software. They are listed in alphabetical order using the IDENT field of the message.

Many of the text portions of the following messages contain \$FAO formatting control sequences. These sequences begin with an exclamation mark. They indicate that variable data is supplied to this message. They also indicate the type of variable data to expect, such as string or numeric data. For example, the DISFAIL1 message contains the following text:

```
failed to disable !AS
```

According to the error description, the failure is related to a specific script supplied in the message. Thus, the !AS control sequence is replaced by the name of the script that could not be disabled.

ABOFAIL1, failed to abort !AD

Explanation: DFG\$DEFRAGMENT returned a failure status when it attempted to abort the specified script. This message is a summary message. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

ABOFAIL2, no script was aborted

Explanation: DFG\$DEFRAGMENT returned a failure status when it attempted to abort all scripts in the scheduling database. This message is a summary message. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

ABORTNOTDONE, abort unsuccessful

Explanation: An ABORT subcommand was unsuccessful. This is a summary message. This message may or may not indicate an error. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

ABOSEN, abort command already issued for !AD

Explanation: An ABORT subcommand had been issued for this script but it had not yet completed when another ABORT subcommand for the same script was issued.

User Action: Wait for the command to complete.

ACCESSERR, access error on file !AD

Explanation: A QIO IO\$_ACCESS function could not be performed on the given file for reasons detailed in associated messages. This message occurs only in the log file or the /MAIL_NOTIFICATION messages.

User Action: Refer to associated messages to determine a course of action.

ACLERR, BUGCHECK - Map pointer copy would run into ACLs

Explanation: The copy made of INDEXF.SYS would run into ACLs.

User Action: Clean up disk and restart program.

ASSIGNERR, channel-assign error for device !AD

Explanation: A channel could not be assigned for the indicated device.

Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

BITMAPCONERR, \$CONNECT error on file !AD:[000000]BITMAP.SYS

Explanation: A \$CONNECT RMS service was unsuccessful for the BITMAP.SYS file of the target device. The RMS error is provided in a subsequent message.

User Action: Determine a course of action based on the RMS error reported.

BITMAPOPENERR, \$OPEN error on file !AD:[000000]BITMAP.SYS

Explanation: A \$OPEN RMS system service could not be performed on the BITMAP.SYS file of the target device. The RMS error is provided in a subsequent message.

User Action: Determine a course of action based on the RMS error reported.

BITMAPREADERR, \$READ error on file !AD:[000000]BITMAP.SYS

Explanation: A \$READ RMS system service failed for the BITMAP.SYS file of the target device. The RMS error is provided in a subsequent message.

User Action: Determine a course of action based on the RMS error reported.

CACHEFLUSHERR, unable to flush extent cache for device !AD

Explanation: An IO\$_ACPCONTROL extent cache flush QIO returned a failure status to the defragmentation process. The root cause is included in associated messages.

User Action: Refer to associated messages to determine a course of action.

CANCELNOTDONE, cancel unsuccessful

Explanation: A CANCEL subcommand was unsuccessful. This is a summary message. This message may or may not indicate an error.

Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

CANFAIL1, failed to cancel !AD

Explanation: DFG\$DEFRAGMENT returned a failure status when it attempted to cancel the specified script. This message is a summary message.

Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

CANFAIL2, no script was canceled

Explanation: DFG\$DEFRAGMENT returned a failure status when it attempted to cancel all scripts in the scheduling database. This message is a summary message. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

CHECKSUMERR, Error when copying map pointers for INDEXF.SYS

Explanation: An error occurred when mapping the new INDEXF.SYS header pointers.

User Action: Submit an SPR.

CLOSEIN, error closing !AD as input

Explanation: An RMS error was encountered when trying to close an input file, such as the options file. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

CLOSEOUT, error closing !AD as output

Explanation: An RMS error was encountered when trying to close an output file. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

CMDUNKNOWN, unknown command; parsing error, please submit an SPR

Explanation: This is an internal error.

User Action: Submit an SPR.

CONNECTERR, \$CONNECT error on log file !AD

Explanation: The RMS \$CONNECT service failed for the log file (occurs in the defragmentation process). Associated messages provide the root cause.

User Action: Refer to associated messages to determine a course of action.

CONSFALL, free-space consolidation failed for !AD

Explanation: Free-space consolidation failed for the given disk

User Action: None, unless indicated by other messages.

COPYRIGHT, © Copyright 2020 VMS Software, Inc., (VSI), L.P. CPUOUTRNG, !SL is an invalid CPU percentage; use 0-100

Explanation: User specified a CPU percent utilization limit that was not within the 0-100 range.

User Action: Specify valid CPU percentages from 0 to 100, inclusive.

CREATEERR, error creating !AD

Explanation: The \$CREATE RMS service failed for the specified file.

Associated messages provide the root cause.

User Action: Refer to associated messages to determine a course of action.

CREPRCFALL, error creating detached defragmentation process

Explanation: The scheduler could not create the detached defragmentation process. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

CRESECFALL, error creating global section

Explanation: The global section used for communications could not be created by the scheduler for reasons given in associated error messages. This error is caused by a failure status from the \$CRMPSC system service, which could indicate insufficient system wide global section resources.

User Action: Refer to associated messages to determine a course of action.

DBACCCONSTR, constraint violation in relation !AD (operation !AD)

Explanation: A constraint in the database was violated. The most common reason is an attempt to enter a duplicate record. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

DBACCDUP, duplicate record in relation !AD (operation !AD)

Explanation: An attempt was made to enter a duplicate record in the database. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

DBACCERR2, database-access error: !AD transaction

Explanation: The database could not be accessed. The cause of this problem may be an undefined logical, corrupt database, DFG software error, or RMS software error. Associated messages provide more detail.

User Action: Verify that all DFG logicals are defined. Refer to associated messages to determine if additional action is necessary.

DBACCERR, database-access error; relation !AD, operation !AD, status !SL

Explanation: The database could not be accessed. The cause of this problem may be an undefined logical, corrupt database, DFG software error, or RMS software error.

User Action: Verify that all DFG logicals are defined. Refer to associated messages to determine if additional action is necessary.

DBACCW2, database-access warning: !AD transaction, sqlcode !SL

Explanation: The Rdb software returned a warning status to the scheduler.

The problem may be that the database is corrupt, that the disk file optimizer software has is in error, or that the Rdb software is in error.

User Action: Try the operation again. If the error persists, re-create the database. If the error persists after re-creating the database, submit an SPR.

DBACCW, database-access warning: relation !AD, operation !AD, sqlcode !SL

Explanation: The Rdb software returned a warning status to the scheduler.

The problem may be that the database is corrupt, that the disk file optimizer software is in error, or that the Rdb software is in error.

User Action: Try the operation again. If the error persists, re-create the database. If the error persists after re-creating the database, submit an SPR.

DBBUSY, accessing the defragmentation database; please wait

Explanation: Informational message that indicates the DCL interface is waiting for the database access/update to complete.

User Action: None.

DBCANTREAD, cannot read a needed record from relation !AD

Explanation: The database could not be read. The cause of this problem may be an undefined logical, corrupt database, DFG software error, or Rdb software error. This problem can also be caused if the RMU Monitor process (RDMS_MONITOR) is not executing. Associated messages provide more detail.

User Action: Verify that all DFG logicals are defined and ensure that RDMS_MONITOR is running. If RDMS_MONITOR is not running, it can be started by invoking @SYS \$STARTUP:RMONSTART.COM. Refer to associated messages to determine if additional action is necessary.

DBCANTWRITE, cannot write or update a record in relation !AD

Explanation: The database could not be updated. The cause of this problem may be an undefined logical, corrupt database, DFG software error, or Rdb software error. This problem can also be caused if the RMU Monitor process (RDMS_MONITOR) is not executing. Associated messages provide more detail.

User Action: Try the command again. If the error persists, re-create the database. If the error persists after re-creating the database, submit an SPR.

DBCLEARED, database cleared

Explanation: A database has been cleared.

User Action: None.

DBCLEAR, clearing database

Explanation: A database is being cleared.

User Action: None.

DBCREATED, database created

Explanation: A database has been created.

User Action: None.

DBCREATE, creating database

Explanation: A database is being created.

User Action: None.

DBINVXLNAM, invalid translated name

Explanation: This is an internal error.

User Action: Submit an SPR.

DBNOREC, no record found in relation !AD (operation !AD)

Explanation: A database record could not be found. This error can occur if the database is corrupt, the disk file optimizer software is in error, or the Rdb software is in error.

User Action: Submit an SPR.

DBROLLBACK, database updates rolled back

Explanation: All updates to the database have been rolled back.

User Action: None

DBUPDATERR, database update failed

Explanation: This is a summary message indicating that database updates were unsuccessful. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

DEACCESSERR, deaccess error on file !AD

Explanation: The specified file could not be deaccessed by the defragmentation process (using the IO \$_DEACCESS QIO). Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

DELERR, cannot delete !AD, notification message

Explanation: An attempt to spawn a subprocess to delete the mail message sent as a result of the / MAIL_NOTIFICATION qualifier failed. This error occurs in the scheduler code.

User Action: Refer to associated messages to determine a course of action.

DFGNOSTART, defragmentation process failed to start

Explanation: This message occurs when a defragmentation process cannot be started. Associated messages indicate why. This message does not occur with Defrag PLUS.

User Action: Refer to associated messages to determine a course of action.

DFGSTART, started defragmentation process PID: !XL

Explanation: This message indicates that a defragmentation process has been started. This message does not occur with Defrag PLUS.

User Action: None.

DFGTMO, defragmentation process timed out on startup

Explanation: The defragmentation process did not map the global section created by the scheduler within a timeout period.

User Action: Ensure a copy of DFG\$ENGINE_MAIN exists in SYS\$SYSTEM and try the command again. If the error persists, submit an SPR.

DIOOUTRNG, !SL is an invalid direct I/O limit, use positive values

Explanation: The user specified a negative number for a direct I/O limit.

User Action: Specify positive values for the IO_LIMIT keyword parameter.

DIRVERERR, directory file !AD is not .DIR;1

Explanation: A file with the directory flag set in its file header is not version 1 or does not have the extension .DIR

User Action: Rename or delete the file.

DISFAIL1, failed to disable !AD

Explanation: This is a summary message indicating DFG\$DEFRAGMENT returned a failure status for a DISABLE command. This failure is related to a specific script supplied in the message. Associated messages provide the cause of the failure.

User Action: Refer to associated messages to determine a course of action.

DISFAIL2, no script was disabled

Explanation: This is a summary message indicating the specified script was not disabled. Associated messages indicate the reason the disable operation failed.

User Action: Refer to associated messages to determine a course of action.

DISKFULL, disk !AD too full for defragmentation

Explanation: This error occurs when the defragmentation process finds that a disk is too full for defragmentation. The process experiencing the error will terminate (if a single volume or a stripe set was defragmented) or continue with the next volume in the volume set.

User Action: None.

DUPDEFRAGER, duplicate concurrent defragmentation processes for device !AD

Explanation: The defragmentation process detected a currently running defragmentation process on the target device. Concurrent defragmentation processes on the same target device are not allowed; thus, this process exits. This ensures that two defragmentation processes do not attempt to defragment the same disk at the same time.

User Action: Ensure that no two defragmentation processes attempt to defragment the same device at the same time.

DUPSCHEM, duplicate concurrent scheduler on node !AD

Explanation: The scheduler detected another currently running scheduler on the current node. Concurrent schedulers on the same node are not allowed.

User Action: Ensure that no two schedulers execute on the same node at the same time.

DUPSCRIPT, duplicate script name: !AD

Explanation: An attempt was made to define two scripts with the same name.

User Action: Reissue the command using a unique script name for the /SCRIPT qualifier.

EMPDFGSCH, empty defragmentation and scheduling item lists

Explanation: Both the scheduler list and the defragmentation list were empty.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

EMPDFG, empty defragmentation item list

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

ENAFAIL1, failed to enable !AD

Explanation: This is a summary message indicating the specified script was not enabled. Associated messages indicate the reason the enable operation failed.

User Action: Refer to associated messages to determine a course of action.

ENAFAIL2, no script was enabled

Explanation: This is a summary message indicating that no script could be enabled (the user specified / ALL on the command line). Associated messages indicate the reason the enable operation(s) failed.

User Action: Refer to associated messages to determine a course of action.

ENGLIMEXC, process limit exceeded; increase process limit to run

Explanation: Starting another defragmentation process on this node would exceed the user-defined maximum number of defragmentation processes for this node.

User Action: If the limit is acceptable, no action is required. Otherwise, change the limit using the SET subcommand.

ENGMAXTIM, time limit exceeded; defragmentation process stopped

Explanation: A defragmentation process was terminated because its time limit was exceeded. This is a limit imposed by the user with the /TIME_LIMIT qualifier.

User Action: None or change the time limit.

ENQLOCKERR, error while attempting to queue a process lock for device !AD

Explanation: The defragmentation process received an unexpected failure status from the \$ENQ system service while trying to get an exclusive lock on the target device. Associated messages provide more detail. This error does not occur for concurrent defragmentation processes.

User Action: Refer to associated messages to determine a course of action.

EXTHDRERR, unable to access extent header(s) for file !AD

Explanation: The defragmentation process could not access the given file's extent header(s) through the IO\$_ACCESS QIO.

User Action: Refer to associated messages to determine a course of action.

FREEVMERR, unable to free dynamic memory

Explanation: The defragmentation process called the LIB\$FREE_VM service incorrectly.

User Action: Submit an SPR.

GETDVIERR, error in getting device information from !AD

Explanation: The defragmentation process received failure status from the GETDVI system service. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

GETVMERR, unable to allocate dynamic memory

Explanation: The defragmentation process received failure status from the LIB\$GET_VM system service. Associated messages provide more detail.

User Action: If the error is due to insufficient system dynamic memory, system resources are insufficient to defragment the volume. If this is not the case, submit an SPR.

GLBSECMAPERR, unable to map global section

Explanation: The defragmentation process could not map the system wide global section created by the scheduler. This error is very unlikely, but could be caused if the scheduler exits after creating the global section and the defragmentation process, but before the defragmentation process has a chance to map the global section. It will also occur if the user tries to run the DFG\$ENGINE_MAIN program directly.

User Action: None.

IDXFILEFULL, INDEXF.SYS is completely full. You must delete some files

Explanation: The index file or header is full. The program needs at least one free file header.

User Action: Delete unneeded file(s).

IDXOPT, Your index file is already optimal, no further work needed.

Explanation: The index file is already optimal. No further work is needed.

User Action: None. The program has completed.

IEDBDATA, logically inconsistent data in relation !AD

Explanation: The database contains inconsistent data. The root of the problem may be that the RMS database is corrupt, that the DFG software has an error.

User Action: Submit an SPR. Include a dump of the scheduling database contents, if possible.

IELOGNAM1, internal error; invalid supersede flag

Explanation: This is an internal error.

User Action: Submit an SPR.

IESCHFILNODTYP, relation !AD: illegal file or node type (!SL)

Explanation: This is an internal scheduler error.

User Action: Submit an SPR.

IESCHFUNC, illegal function code (!SL) in !AD

Explanation: This is an internal scheduler error.

User Action: Submit an SPR.

IESCHIMPOSS, internal logic error in !AD

Explanation: This is an internal scheduler error.

User Action: Submit an SPR.

IESE1, internal memory allocation calculation error; submit an SPR

Explanation: A consistency check found the global section allocated was not large enough to hold the necessary data. This is an internal error.

User Action: Submit an SPR.

IESRCINCONSIS, internal error; inconsistent source !AD

Explanation: This error occurs when the source for one module is not compatible with the source for another module.

User Action: Submit an SPR.

IEVERNOD1, internal error while verifying node

Explanation: This is an internal error

User Action: Submit an SPR.

IMEDALL, all scripts have been !AD

Explanation: Success message for any immediate command (ENABLE, DISABLE) where /ALL was specified.

User Action: None.

IMEDSUCC, script !AD has been !AD

Explanation: Success message for any immediate command (ENABLE, DISABLE) where specific scripts were specified.

User Action: None.

INTERNALERR, internal software error number !UL

Explanation: This is an internal error encountered in the defragmentation process code.

User Action: Submit an SPR. Include the software error number.

INTERR2, internal error: !AD !XL

Explanation: This is an internal error.

User Action: Submit an SPR. Include the error number and the associated string supplied in the error message.

INTERR, internal error: !AD

Explanation: This is an internal error.

User Action: Submit an SPR. Include the internal error string.

INVDEVCONT, device not suitable for defragmentation, do you wish to continue (Y, [N])?

Explanation: The user specified a device for a defragment operation that is currently unavailable or unsuitable for defragmentation. However, since the device may become available by the time the defragmentation operation begins, the user is allowed the option of continuing or aborting the command.

User Action: If the device specified is correct, but is currently off line or, for some other reason, is unsuitable for defragmentation, enter Y in response to the prompt. Otherwise, enter N or press Return to abort the command.

INVDEVNAM, !AD is an invalid device name

Explanation: The supplied device name was invalid.

User Action: Specify a valid device name.

INVDEV, device unsuitable for defragmentation

Explanation: Target device not suitable for defragmentation.

User Action: Ensure that the target device for defragmentation is an ODS-2 volume (or stripe set or volume set). Further, it must be a local device available from the issuing node.

INVDFGLST, invalid defragmentation item list

Explanation: The defragmentation list passed to DFG\$DEFRAGMENT was invalid, because of one of the following:

- A call to DFG\$DEFRAGMENT using a scheduler list that indicates the defragmentation list should be empty. For example, DFG\$K_SHOW_SCRIPT must not have a defragmentation item list.
- Something was wrong with the semantics of the defragmentation item list, or a particular item descriptor was formatted improperly.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

INVDSLST, invalid defragmentation and scheduler item lists

Explanation: Both the scheduler list and the defragmentation list passed to DFG\$DEFRAGMENT were invalid.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

INVFILNAM, !AD is an invalid file specification

Explanation: An invalid file name was given for either the log file or the options file. Associated messages describe why the file name is invalid.

User Action: Based on the associated messages, specify a valid file name.

INVITEMCOMB, invalid combination of DFG\$K_INTERVAL and DFG\$K_PREDECESSOR

Explanation: The DFG\$K_INTERVAL item descriptor is not allowed with the DFG\$K_PREDECESSOR item descriptor in a call to DFG\$DEFRAGMENT.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

INVITMLST, invalid combination of items in item list

Explanation: This message indicates the dfg_list or sch_list parameter to DFG\$DEFRAGMENT was logically invalid. One of the lists contained item codes that were not compatible.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

INVITMUKN, an item list contained an unknown item code

Explanation: This message indicates the dfg_list or sch_list parameter to DFG\$DEFRAGMENT contained an item descriptor with an unknown item code. It may also indicate a faulty item_list_2 returned from DFG\$DEFRAGMENT as part of the SHOW subcommand.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

INVITM, invalid item list; item !SL is in error

Explanation: This message indicates the dfg_list or sch_list parameter to DFG\$DEFRAGMENT was incorrectly formatted or logically invalid. It provides the item number that was incorrect. Associated messages indicate further errors.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

INVLMFLIC, insufficient license units to run DEFRAGMENT

Explanation: The DFG software is not licensed to run on this node.

User Action: Install the DFG license.

INVLOGNAM, !AD is an invalid logical name

Explanation: A logical name translation failed because the logical name was invalid. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

INVSCHLST, invalid scheduler item list

Explanation: The scheduler list passed to DFG\$DEFRAGMENT was invalid.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

INVSCRNAM, !AD is an invalid script name

Explanation: The user tried to name a script using characters that are not supported for script names.

User Action: Reenter the script name using valid characters.

INVSTARTCMD, script !AD already has a queue entry

Explanation: The user entered a START subcommand for a script that was already started.

User Action: None.

INVSTRDSCNBR, invalid string descriptor in item !SL of item list

Explanation: An invalid string descriptor was pointed to from an item list passed to DFG \$DEFRAGMENT. The item number of the item descriptor containing the offending string descriptor is supplied by this message.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

INVSTRDSC, invalid string descriptor

Explanation: An invalid string descriptor was pointed to from an item list passed to DFG \$DEFRAGMENT.

User Action: If the call to DFG\$DEFRAGMENT was from VSI supplied software, submit an SPR. Otherwise, correct the calling program.

L5NOLOG, log file not specified, but defragmentation level is 5

Explanation: Occurs if /LEVEL=5 is specified without the /LOG qualifier.

Defragmentation statistics are unavailable without the /LOG qualifier; thus, it is required when /LEVEL=5 is specified.

User Action: Specify /LOG={logfile} when using the /LEVEL=5 qualifier.

LIMEXC, !AD limit exceeded

Explanation: A user-specified CPU_LIMIT, IO_LIMIT, or PROCESS_LIMIT has been exceeded.

User Action: If the limit is correctly set, no action is required. If the limit is incorrect, set a new limit using the SET subcommand.

LSTBADVOL, incorrect device name !AD specified on file list

Explanation: A device name was specified with a file on the frequent, dormant, or exclude lists that does not match the target device name.

User Action: Correct the device name or remove it altogether.

LVLOUTRNG, !SL is an invalid defragmentation level; use 1-5

Explanation: The defragmentation level specified by the user on the command line was not a number from 1 to 5, inclusive.

User Action: Specify a valid defragmentation level.

MISSING_LOGICAL, the logical name !AD is not defined

Explanation: A required logical name is not defined.

User Action: Define the indicated logical name and try the command again.

MISSNGVAL, !AD qualifier requires a value in this context

Explanation: A qualifier needed a value, such as a script name, but the user did not provide one.

User Action: Provide the value requested.

MONBRKNDB, monitor unable to access database for script information

Explanation: This error occurs if the monitor cannot access the scheduling database. The most likely causes of this error are: the logical name DFG\$DEFRAGMENT is not defined; the database does not exist.

User Action: If this message persists and the other defragmentation software is working (scheduler, DCL interface, defragmentation processes), submit an SPR.

MONINVCOL, too few columns in display, please resize

Explanation: The display has too few columns to display meaningful information.

User Action: If your terminal is a DECwindows DECterm, resize the window as appropriate. If your terminal is a VT100-style device, submit an SPR.

MONINVROW, too few rows in display, please resize

Explanation: The display has too few rows to display meaningful information.

User Action: If your terminal is a DECwindows DECterm, resize the window as appropriate. If your terminal is a VT100-style device, submit an SPR.

MONNOVOL, !AD is not currently being defragmented

Explanation: An attempt was made to monitor a script that is not executing.

User Action: Monitor only executing defragmentation processes.

MONRDERR, error !SL while reading the monitor interface

Explanation: This error occurs if a process cannot read data using the interface that exists between the monitor process, the scheduling process, and the defragmentation process. The process experiencing the error will continue, though not all data may be available for the monitor display.

User Action: None.

MONUNEXP, unexpected error from !AD

Explanation: An error occurred in a call to the Screen Management facility (SMG). The SMG routine that returned the error is displayed. A subsequent error message contains the SMG error code.

User Action: Refer to the subsequent error message to determine a course of action.

MONWRERR, error !SL while writing the monitor interface

Explanation: This error occurs if a process cannot write data using the interface that exists between the monitor process, the scheduling process, and the defragmentation process. The process experiencing the error will continue, though not all data may be available for the monitor display.

User Action: None.

MOVFILINFO1, Relative volume number !UL

Explanation: This is issued to give more information to the maintainer about a movefile error (either MOVFILQIOERR or MOVFILSTSERR).

User Action: Submit an SPR.

MOVFILQIOERR, QIO error on movefile for volume !AD

Explanation: This is an internal error; the movefile QIO was formatted incorrectly by the defragmentation process.

User Action: Submit an SPR.

MOVFILSTSERR, status error on movefile for volume !AD

Explanation: The movefile QIO returned an error status to the defragmentation process. Subsequent messages provide the specific error encountered.

User Action: Refer to subsequent messages to determine a course of action.

MULTSUCC, multiple successors not allowed

Explanation: The script named as predecessor already has a successor defined.

User Action: Specify a different predecessor or none.

NODEIGNOR, the /NODE qualifier was ignored

Explanation: The /NODE qualifier was specified with show script, but no CPU, I/O, or process parameter was selected. Any node name is meaningless when applied to show script only. This message is informational.

User Action: None.

NOINHER, inherit script !AD not found

Explanation: The script specified in the /INHERIT qualifier could not be found in the SCRIPTS relation.

User Action: Reissue the command using a defined script name for the /INHERIT qualifier.

NOKILL, failed attempt to stop a defragmentation process

Explanation: An attempt (by the scheduler) to abort the defragmentation process failed. Associated messages provide more detail. It could simply be that the defragmentation process has completed or is suspended.

User Action: Refer to associated messages to determine a course of action.

NOMEINTERFACE, monitor interface not established

Explanation: This error occurs if the defragmentation process cannot establish the monitor interface. In this case, the error appears in the defragmentation process log file. The error can also occur when executing the monitor function if the monitor interface is not yet established. The error can also appear in the scheduler log file if the interface is not established.

User Action: If you are trying to run the monitor, try the command again as the defragmentation process may not yet have created the interface. If the error occurs in the defragmentation process or scheduler log files, determine a course of action based on any associated messages. If the error is due to insufficient ENQUE limit, increase the ENQUE limit of the user process running monitor or the scheduler process, whichever is appropriate.

NONODE, !AD not found in configuration

Explanation: A node name that was associated with a script was not present in the configuration. This error can also occur during scheduler initialization if both the SYSGEN parameter, SCSNODE, and the logical, SYS\$NODE are undefined on the node where the scheduler is running. In the case of the scheduler, the node name reported in the message will be blank.

User Action: If the error occurs in the scheduler log file, define either the logical name SYS\$NODE or the SYSGEN parameter SCSNODE and restart the scheduler. If the error occurs when entering a command, ensure you have spelled the node name correctly and that the node exists on the local system.

NONOTIFY, mail notification of defragmentation process completion failed

Explanation: An attempt to send mail to the mail distribution list specified in DFG\$MAIL_ADDRESS has failed.

User Action: Verify that the logical DFG\$MAIL_ADDRESS is defined. Refer to associated messages to determine if additional action is necessary.

NOPRED, predecessor script !AD not found

Explanation: The script specified in the /PREDECESSOR qualifier could not be found in the SCRIPTS relation.

User Action: Check spelling of the script name used with the /PREDECESSOR qualifier. Use the name of a script that has already been defined (with the VOLUME subcommand) for the /PREDECESSOR qualifier.

NOPRIV, insufficient privileges; you need BYPASS, CMKRNL, DETACH, EXQUOTA, NETMBX, SHARE, SYSGBL, SYSLCK, SYSPRV, TMPMBX, and WORLD

Explanation: The user has insufficient privileges to run DEFRAGMENT.

User Action: Set the BYPASS, CMKRNL, DETACH, EXQUOTA, NETMBX, SHARE, SYSGBL, SYSLCK, SYSPRV, TMPMBX, and WORLD privileges and try the command again.

NOQENTRY, no queue entry for script !AD

Explanation: A queue entry was not found for a script that was to be canceled, disabled, or enabled. (The database did not have a record that matched the user-supplied script name for one of these commands.)

User Action: Check the spelling of the script name. Use only names that have previously been defined with the VOLUME subcommand.

NOSCHED, scheduling operations may not have completed

Explanation: One or more scheduling operations may not have completed.

This is a summary message. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

NOSCRIP, script name !AD is not defined in the database

Explanation: The named script is not defined in the database.

User Action: Check the spelling of the script name and try the command again.

NOSCRNAM, script name !AD not found

Explanation: A script name specified by the user was not in the scheduling database.

User Action: Check the spelling of the script name and try the command again.

NOSHODAT, error retrieving show data from the scheduling database

Explanation: There was an error retrieving show data from the scheduling database.

User Action: Refer to associated messages to determine a course of action.

NOTACCVIO, cannot access notification data in !AD

Explanation: This is an internal error. It occurs if one of the internal user notification routines is called with an inaccessible parameter.

User Action: Submit an SPR.

NOTBADDATA, invalid data found in !AD

Explanation: This is an internal error. It occurs if one of the internal user notification routines is called with invalid data.

User Action: Submit an SPR.

NOTCLOSERR, error closing temporary notification file

Explanation: An error occurred while closing the temporary notification file.

User Action: Refer to associated messages to determine a course of action.

NOTDISK, this device is not a disk, recheck the device name

Explanation: The device for which you requested defragmenting is not a disk.

User Action: Check the device name and try the command again.

NOTEXEC, script !AD not executing

Explanation: An attempt was made to abort a script that was not executing.

User Action: None.

NOTEXTEND, error extending the scratch file needed for defragmenting INDEXF.SYS

Explanation: The temporary scratch file needed to defragment INDEXF.SYS could not be created. Its creation would have caused a multi-header entry or there was not enough space on the volume.

User Action: Before defragmenting INDEXF.SYS, be sure to defragment the volume and consolidate free space. Also, make sure that the largest free space on the volume is larger than INDEXF.SYS. To check the largest free space and extents of INDEXF.SYS, execute the following command:

```
$ DEFRAGMENT SHOW disk:/VOLUME/LOCATION=[000000]INDEXF.SYS
```

NOTFILWRERR, error writing to temporary notification file

Explanation: The notification message could not be written.

User Action: Refer to associated messages to determine a course of action.

NOTLOCDEV, !AD is not local to !AD

Explanation: The user has specified a target device that is not local to the node that is trying to run the defragmentation process.

User Action: Nonlocal devices (such as DFS mounted devices) cannot be defragmented.

NOTODSDEV, !AD is not an ODS-2 mounted device

Explanation: The user has specified a device for defragmentation that is not a mounted ODS-2 format volume.

User Action: Specify an ODS-2 format volume.

NOTRAN, error translating !AD

Explanation: There was an error translating the supplied logical name.

User Action: Define the indicated logical using the /EXECUTIVE_MODE and /SYSTEM qualifiers.

NOTSTOPPED, defragmentation process, ID=!XL, was not stopped

Explanation: An attempt to stop a defragmentation process failed.

Associated messages provide more detail. The most likely cause of this message is that the defragmentation process is no longer running.

User Action: None.

NOVOLUME, no volume name specified for VOLUME subcommand

Explanation: DFG\$DATABASE received a dfg_list item list that did not contain a volume name item descriptor.

User Action: If this error occurs with VSI supplied software, submit an SPR.

Otherwise, correct the calling program.

NULLENGINE, a defragmentation process cannot be started for script !AD

Explanation: An attempt to start a defragmentation process failed.

Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

OPENIN, error opening !AD as input

Explanation: The specified file could not be opened for input. Associated messages indicate the reason.

User Action: Refer to associated messages to determine a course of action.

OPENMAILERR, cannot open mail file for /MAIL_NOTIFICATION

Explanation: The mail file cannot be opened by the scheduler when attempting to notify users by way of the /MAIL_NOTIFICATION qualifier.

Associated messages indicate the reason.

User Action: Refer to associated messages to determine a course of action.

OPENOUT, error opening !AD as output

Explanation: The specified file could not be opened for output. Associated messages indicate the reason.

User Action: Refer to associated messages to determine a course of action.

OPTBADVOL, invalid device specified on line !SL, options file !AD

Explanation: A volume was specified with a file name in the options file and the volume was not the target defragmentation volume.

User Action: Correct the invalid line in the options file. Specify no device name, or specify the target device name.

OPTFILERR, options file error

Explanation: The options file contained errors. This is a summary message.

Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

OPTHASFNF, file not found on line !SL, options file !AD

Explanation: A file specified as /EXCLUDE, /FREQUENT, or /DORMANT in the options file was not found.

User Action: No action is necessary. To eliminate this message, however, you should remove the line in the options file that is reported in this message.

Since the file given on this line is no longer on the target device, this line has no effect.

OPTINVQUAL, invalid qualifier on line !SL, options file !AD

Explanation: A classification for a file was given that was not one of:

/FREQUENT, /DORMANT, /EXCLUDE.

User Action: Check spelling and ensure that only classifications of /FREQUENT, /DORMANT, or /EXCLUDE are used for files listed in the options file.

OPTMULCLA, multiple classifications on line !SL, options file !AD

Explanation: A file was classified with multiple classifications (that is, file.ext/FREQ/DORM). This error prevents the defragmentation process from being started.

User Action: Specify only one file classification per line in the options file.

OPTNOTCLA, file not classified on line !SL, options file !AD

Explanation: A file was in the options file, but was not classified as /FREQUENT, /DORMANT, or /EXCLUDE.

User Action: Make sure the file on the line specified in the options file has one of the above-listed classifications.

OPTRMSERR, RMS error on line !SL, options file !AD

Explanation: An unanticipated error was returned from the \$PARSE or \$SEARCH RMS routines.

User Action: Refer to the specified line in the options file. Also check associated messages for more information.

PARSEFAIL, error parsing !AD

Explanation: The RMS \$PARSE service failed for the specified file.

User Action: Refer to associated messages to determine a course of action.

PRCOUTRNG, !SL is an invalid process limit; specify positive values

Explanation: The process limit was negative.

User Action: Specify only positive values for the PROCESS_LIMIT keyword parameter to the SET subcommand.

PREDNOTUSRNAME, the predecessor for script !AD is not a user-provided name

Explanation: The predecessor for a script was not a user-provided name. This message can occur during the SHOW subcommand when the /COMMAND_LINE qualifier is present. Because script names beginning with DFG\$ are reserved for the DEFRAGMENT software, they are never listed when the /COMMAND_LINE qualifier is used. This warning message indicates the script whose predecessor was not listed.

User Action: If the output of the SHOW subcommand was directed to a file, edit the file and name the script to be used as a predecessor using the /SCRIPT qualifier. Then, modify the script listed in this message so the /PREDECESSOR qualifier indicates the script name used.

PROCABORTED, defragmentation process aborted

Explanation: The defragmentation process has aborted. It could be that the total defragmentation time has been exceeded and the scheduler has stopped the defragmentation software. Thus, this message does not necessarily indicate failure.

User Action: Refer to associated messages to determine a course of action.

PROCESS_START, defragmentation process start

Explanation: This message is generated by the defragmentation process after it has completed initialization.

User Action: None.

PROCESS_STOP, defragmentation process normal completion

Explanation: This message is generated by the defragmentation process after the process has successfully completed.

User Action: None

QUEDALL, !AD request for all scripts has been successfully queued

Explanation: Any of the queued commands where /ALL was specified were successful.

User Action: None

QUEDNONAM, !AD request for unnamed script has been successfully queued

Explanation: A defragmentation request was successfully processed; the user did not supply a script name.

User Action: None

QUEDSUCC, !AD request for script !AD has been successfully queued

Explanation: A defragmentation process was successfully queued and had a user-specified script name.

User Action: None

REMFAIL1, failed to remove !AD from the database

Explanation: This is a summary message indicating the specified script was not removed. Associated messages indicate the reason the remove operation failed.

User Action: Refer to associated messages to determine a course of action.

REMFAIL2, no script was removed from the database

Explanation: This is a summary message indicating that no script was removed from the scheduling database when the user had specified the /ALL qualifier.

User Action: Refer to associated messages to determine a course of action.

REMOVENOTDONE, remove unsuccessful

Explanation: A REMOVE subcommand was unsuccessful. This is a summary message. This message may or may not indicate an error.

Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

REMOVSEEN, remove command for !AD already issued

Explanation: A REMOVE subcommand had been issued for this script, but it had not yet completed when another REMOVE subcommand for the same script was issued.

User Action: None.

RESAREA, Map pointer copy would extend into a reserved area

Explanation: The copy of INDEXF.SYS map pointers would run into a reserved area.

User Action: Submit an SPR.

RMSERR, RMS operation !AD failed (file !AD)

Explanation: The specified RMS operation failed.

User Action: Refer to associated messages to determine a course of action.

SCHEDERR, scheduler cannot complete scheduling tasks

Explanation: This is a summary message indicating that the scheduler did not complete all its scheduling tasks.

User Action: Refer to associated messages to determine a course of action.

SCHEXIT, scheduler exited at user request

Explanation: The scheduler exited normally. This occurs if the user executes the DFG \$STOP_SCHEDULER.COM procedure, runs the SYS\$SYSTEM:DFG\$STOP_SCHEDULER.EXE image, or executes the DFG\$SHUTDOWN.COM procedure.

User Action: None.

SCHSTART, !AD started on !AD !%D

Explanation: A scheduler started. This message appears at the beginning of every scheduler log.

User Action: None.

SCRDIS, script !AD is already disabled

Explanation: An attempt was made to disable a script that was already disabled.

User Action: None.

SCRENA, script !AD is already enabled

Explanation: An attempt was made to enable a script that was already enabled.

User Action: None.

SEARCHLOOP, loop in \$SEARCH: same file returned twice

Explanation: RMS's \$SEARCH function did not return files in alphabetical order. This may be caused by a corrupt directory file or by an alias that points to a deleted file.

User Action: Investigate the directory structure of the disk.

SENDMAILERR, mail notification failed

Explanation: This is a summary message that occurs when the scheduler does not send the mail notification message requested by the user with /MAIL_NOTIFICATION. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

SETFAIL, failed to set any specified parameters

Explanation: DFG\$DEFRAGMENT returned a failure status for a SET command.

User Action: Refer to associated messages to determine a course of action.

SETNODFAIL, a SET subcommand failed for node !AD

Explanation: A DEFRAGMENT SET CPU, I/O, or PROCESS command failed for the indicated node.

User Action: Refer to associated messages to determine a course of action.

SETSUCC, specified parameters successfully set

Explanation: The database was successfully updated with the new CPU_LIMIT, IO_LIMIT, and PROCESS_LIMIT values.

User Action: None.

SHDWSETERR, device is a member of a shadow set; specify the shadow set name

Explanation: The user attempted to defragment a member of a shadow set.

Attempting to defragment members of a shadow set is not supported.

User Action: Defragment the entire shadow set by specifying the shadow set name (DUSXXX) on the DEFRAGMENT command line.

SHODATMIS, !AD missing from show data

Explanation: This is an internal error.

User Action: Submit an SPR.

SHOSCRFAIL, a show script command failed

Explanation: A SHOW SCRIPT command did not complete successfully.

Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

STAFAIL, failed to start !AD

Explanation: DFG\$DEFRAGMENT returned a failure status for a START subcommand. This is a summary message and probably does not indicate any real error. Most likely, the script was already started. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

STARTNOTDONE, start command unsuccessful

Explanation: A START subcommand was unsuccessful. This is a summary message. This message may or may not indicate an error. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

STARTSEEN, start command for !AD already issued

Explanation: A START subcommand had been issued for this script but it had not yet completed when another START subcommand for the same script was issued.

User Action: None.

SYSSRVERR, call to system service !AD failed

Explanation: A system service call failed.

User Action: Refer to associated messages to determine a course of action.

TIMEINV, invalid time string !AD from !AD

Explanation: The syntax of the time string supplied in the message was invalid or out of range. The offending routine is also listed in the message.

User Action: Submit an SPR containing the information in this message.

UNRECOVERERR, unrecoverable error

Explanation: The defragmentation process encountered an unrecoverable error; no further processing occurred.

User Action: Refer to associated messages to determine a course of action.

USERABORT, command aborted by user

Explanation: The user decided not to continue after encountering the INVDEVCONT message. This is a summary message.

User Action: None.

USER_ERR, user error

Explanation: The user made an error. Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

USRKILLENG, an ABORT command terminated the defragmentation process

Explanation: Because of a user-issued ABORT command, a defragmentation process terminated.

User Action: None.

VIRMEMERR, device is a member of a constituent set; specify the set name

Explanation: The user attempted to defragment a virtual member of a disk set, such as a stripeset member.

User Action: Defragment the entire set by specifying the set name on the DEFRAGMENT command line.

VOLFAIL1, volume command failed; script !AD was not entered into the database

Explanation: The specified script was not entered into the database. Some portion of command validation failed. The exact nature of the error is provided in associated messages.

User Action: Refer to associated messages to determine a course of action.

VOLFAIL2, volume command failed; unnamed script was not entered into the database

Explanation: The command was not entered into the database. Some portion of command validation failed. The exact nature of the error is provided in associated messages.

User Action: Refer to associated messages to determine a course of action.

VOLSETERR, device is a member of a volume set; specify the volume set name

Explanation: The user attempted to defragment a member of a volume set.

Attempting to defragment members of a volume set is not supported.

User Action: Defragment the entire volume set by specifying the volume set name on the DEFRAGMENT command line.

VOLUMELOCKERR, volume lock error for device !AD

Explanation: Error trying to allocation-lock the volume. This error may only occur in the defragmentation process code. This prevents file structure changes for a short time while the defragmentation process reads a fresh copy of BITMAP.SYS.

User Action: Refer to associated messages to determine a course of action.

VOLUMENOTDONE, volume command unsuccessful

Explanation: A VOLUME subcommand was unsuccessful. This is a summary message. This message may or may not indicate an error.

Associated messages provide more detail.

User Action: Refer to associated messages to determine a course of action.

VOLUNLOCKERR, volume unlock error for device !AD

Explanation: The target device could not be allocation-unlocked by the defragmentation process.

User Action: Refer to associated messages to determine a course of action.

VPAEND, !SL hotfile records added to the /FREQUENT list.

Explanation: This message indicates the VPA database query has completed. It also indicates the total number of hotfile records found that were related to the target device.

User Action: None.

