Veritas InfoScale™ 7.4.2 Storage and Availability Management for Oracle Databases - AIX, Linux, Solaris

VERITAS™

Last updated: 2020-06-01

Legal Notice

Copyright © 2020 Veritas Technologies LLC. All rights reserved.

Veritas and the Veritas Logo are trademarks or registered trademarks of Veritas Technologies LLC or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.

This product may contain third-party software for which Veritas is required to provide attribution to the third-party (“Third-Party Programs”). Some of the Third-Party Programs are available under open source or free software licenses. The License Agreement accompanying the Software does not alter any rights or obligations you may have under those open source or free software licenses. Refer to the third-party legal notices document accompanying this Veritas product or available at:
https://www.veritas.com/about/legal/license-agreements

The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation/reverse engineering. No part of this document may be reproduced in any form by any means without prior written authorization of Veritas Technologies LLC and its licensors, if any.

THE DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. VERITAS TECHNOLOGIES LLC SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

The Licensed Software and Documentation are deemed to be commercial computer software as defined in FAR 12.212 and subject to restricted rights as defined in FAR Section 52.227-19 "Commercial Computer Software - Restricted Rights" and DFARS 227.7202, et seq. "Commercial Computer Software and Commercial Computer Software Documentation," as applicable, and any successor regulations, whether delivered by Veritas as on premises or hosted services. Any use, modification, reproduction release, performance, display or disclosure of the Licensed Software and Documentation by the U.S. Government shall be solely in accordance with the terms of this Agreement.

Veritas Technologies LLC
2625 Augustine Drive
Santa Clara, CA 95054
http://www.veritas.com

Technical Support

Technical Support maintains support centers globally. All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policies. For information about our support offerings and how to contact Technical Support, visit our website:
https://www.veritas.com/support

You can manage your Veritas account information at the following URL:
https://my.veritas.com

If you have questions regarding an existing support agreement, please email the support agreement administration team for your region as follows:

Worldwide (except Japan)	CustomerCare@veritas.com
Japan	CustomerCare_Japan@veritas.com

Documentation

Make sure that you have the current version of the documentation. Each document displays the date of the last update on page 2. The latest documentation is available on the Veritas website:
https://sort.veritas.com/documents

Documentation feedback

Your feedback is important to us. Suggest improvements or report errors or omissions to the documentation. Include the document title, document version, chapter title, and section title of the text on which you are reporting. Send feedback to:
infoscaledocs@veritas.com

You can also see documentation information or ask a question on the Veritas community site:
http://www.veritas.com/community/

Veritas Services and Operations Readiness Tools (SORT)

Veritas Services and Operations Readiness Tools (SORT) is a website that provides information and tools to automate and simplify certain time-consuming administrative tasks. Depending on the product, SORT helps you prepare for installations and upgrades, identify risks in your datacenters, and improve operational efficiency. To see what services and tools SORT provides for your product, see the data sheet:
https://sort.veritas.com/data/support/SORT_Data_Sheet.pdf

Contents

Section 1 Storage Foundation High Availability (SFHA) management solutions for Oracle databases 19

Chapter 1 Overview of Storage Foundation for Databases 20

Introducing Storage Foundation High Availability (SFHA) Solutions for Oracle 20
About Veritas File System 21
About the Veritas File System intent log 22
About extents 23
About file system disk layouts 23
About Veritas Volume Manager 24
About Dynamic Multi-Pathing (DMP) 25
About Cluster Server 25
About Cluster Server agents 25
About Veritas InfoScale Operations Manager 26
Feature support for Oracle across Veritas InfoScale 7.4.2 products 26
Use cases for Veritas InfoScale products 29

Section 2 Deploying Oracle with Veritas InfoScale products 34

Chapter 2 Deployment options for Oracle in a Storage Foundation environment 35

Oracle deployment options in a Storage Foundation environment 35
Oracle single instance in a Storage Foundation environment 36
Single instance Oracle with off-host in a Storage Foundation environment 37
Single instance Oracle in a highly available cluster with Storage Foundation High Availability 38

Single instance Oracle in a parallel cluster with SF Cluster File System HA environment 40
About Oracle RAC in a Storage Foundation for Oracle RAC environment 42
About Oracle in a replicated Storage Foundation environment 43
Deploying Oracle and Storage Foundation in a virtualization environment 44
Deploying Oracle with Storage Foundation SmartMove and Thin Provisioning 45

Chapter 3 Deploying Oracle with Storage Foundation 46

Tasks for deploying Oracle databases 46
Planning your Oracle storage 47
About selecting a volume layout for deploying Oracle 48
Setting up disk group for deploying Oracle 48
Disk group configuration guidelines for deploying Oracle 50
Creating volumes for deploying Oracle 50
Volume configuration guidelines for deploying Oracle 51
Creating VxFS file system for deploying Oracle 51
File system creation guidelines for deploying Oracle 52
Mounting the file system for deploying Oracle 53
Installing Oracle and creating database 54

Chapter 4 Deploying Oracle in an off-host configuration with Storage Foundation 55

Requirements for an off-host database configuration 55

Chapter 5 Deploying Oracle with High Availability 57

Tasks for deploying Oracle in an HA configuration 57
Configuring VCS to make the database highly available 57

Chapter 6 Deploying Oracle with Volume Replicator (VVR) for disaster recovery 58

About deploying Oracle with VVR for disaster recovery 58
Tasks for deploying Oracle with VVR for disaster recovery 59
Setting up the primary node for deploying Oracle 59
Setting up the secondary node for deploying Oracle 60
Configuring VVR between primary and secondary nodes for deploying Oracle 62

Section 3 Configuring Storage Foundation for Database (SFDB) tools 64

Chapter 7 Configuring and managing the Storage Foundation for Databases repository database 65

About the Storage Foundation for Databases (SFDB) repository 66
Requirements for Storage Foundation for Databases (SFDB) tools 66
Storage Foundation for Databases (SFDB) tools availability 66
Configuring the Storage Foundation for Databases (SFDB) tools repository 67
Locations for the SFDB repository 67
Setting up the SFDB repository 69
Backing up and restoring the Storage Foundation for Databases (SFDB) repository 70
Updating the Storage Foundation for Databases (SFDB) repository after adding a node 72
Updating the Storage Foundation for Databases (SFDB) repository after removing a node 73
Removing the Storage Foundation for Databases (SFDB) repository 73

Chapter 8 Configuring authentication for Storage Foundation for Databases (SFDB) tools 75

Configuring vxdbd for SFDB tools authentication 75
Adding nodes to a cluster that is using authentication for SFDB tools 76
Authorizing users to run SFDB commands 77

Section 4 Improving Oracle database performance 79

Chapter 9 About database accelerators 80

About Veritas InfoScale product components database accelerators 80

Chapter 10 Improving database performance with Veritas Extension for Oracle Disk Manager 83

About Oracle Disk Manager in the Veritas InfoScale products environment 83
How Oracle Disk Manager improves database performance in the SFHA environment 85
How Oracle Disk Manager works with Oracle Managed Files 87
Setting up Veritas Extension for Oracle Disk Manager in SFHA environment 89
Configuring the Veritas Extension for Oracle Disk Manager in SFHA environment 90
How to prepare existing database storage for Oracle Disk Manager in SFHA environment 91
Verifying that Oracle Disk Manager is configured in SFHA environment 92
Disabling the Oracle Disk Manager feature in SFHA environment 94

Chapter 11 Improving database performance with Veritas Cached Oracle Disk Manager 96

About Cached ODM in SFHA environment 96
Considerations for using Cached ODM in SFHA environment 96
Cached ODM supported configurations in SFHA environment 97
About Cached ODM Advisor in SFHA environment 97
Configuring Cached ODM in SFHA environment 98
Enabling Cached ODM for file systems in SFHA environment 98
Tuning Cached ODM settings for individual files in SFHA environment 98
Tuning Cached ODM settings via the cachemap in SFHA environment 99
Making the caching settings persistent across mounts in SFHA environment 100
Administering Cached ODM settings with Cached ODM Advisor in SFHA environment 101
Generating reports of candidate datafiles by using Cached ODM Advisor in SFHA environment 104
Enabling and disabling Cached ODM on data files by using Cached ODM Advisor in SFHA environment 109

Display the Cached ODM states of the database files by using Cached ODM Advisor in SFHA environment 111
Show Cached ODM statistics by using Cached ODM Advisor in SFHA environment 112
Displaying ODM I/O statistics by using Cached ODM Advisor in SFHA environment 113

Chapter 12 Improving database performance with Quick I/O 115

About Quick I/O 116
How Quick I/O improves database performance 116
Creating Oracle database files as Quick I/O files using qiomkfile 118
Preallocating space for Quick I/O files using the setext command 121
Accessing regular VxFS files as Quick I/O files 122
Converting Oracle files to Quick I/O files 124
About sparse files 129
Handling Oracle temporary tablespaces and Quick I/O 130
Displaying Quick I/O status and file attributes 131
Extending a Quick I/O file 133
Using Oracle's AUTOEXTEND with Quick I/O files 134
Recreating Quick I/O files after restoring a database 136
Disabling Quick I/O 138
Creating Quick I/O files in Solaris local zone 139

Chapter 13 Improving database performance with Cached Quick I/O 141

About Cached Quick I/O 141
Tasks for setting up Cached Quick I/O 142
Enabling Cached Quick I/O on a file system 142
Enabling and disabling the qio_cache_enable flag 143
Making Cached Quick I/O settings persistent across reboots and mounts 144
Using vxtunefs to obtain tuning information 145
Determining candidates for Cached Quick I/O 147
Collecting I/O statistics 147
About I/O statistics for Oracle 148
Effects of read-aheads on I/O statistics 150
Other tools for analysis 150
Enabling and disabling Cached Quick I/O for individual files 150
Setting cache advisories for individual files 151

Making individual file settings for Cached Quick I/O persistent ... 151
Determining individual file settings for Cached Quick I/O using qioadmin ... 152

Section 5 Using point-in-time copies ... 154

Chapter 14 Understanding point-in-time copy methods ... 155

About point-in-time copies ... 155
When to use point-in-time copies ... 156
About Storage Foundation point-in-time copy technologies ... 157
Point-in-time copy solutions supported by SFDB tools ... 158
About snapshot modes supported by Storage Foundation for Databases (SFDB) tools ... 159
Volume-level snapshots ... 161
Persistent FastResync of volume snapshots ... 161
Data integrity in volume snapshots ... 161
Third-mirror break-off snapshots ... 162
Space-optimized instant volume snapshots ... 163
About Reverse Resynchronization in volume-level snapshots (FlashSnap) ... 164
Types of Reverse Resync operations ... 165
Limitations of Reverse Resync ... 166
Pre-requisites for performing Reverse Resync operation ... 167
Changes introduced in the implementation of Reverse Resync from 5.X ... 167
Configuration parameters for Reverse Resync operation ... 168
Storage Checkpoints ... 169
How Storage Checkpoints differ from snapshots ... 170
How a Storage Checkpoint works ... 171
About Database Rollbacks using Storage Checkpoints ... 175
Storage Checkpoints and Rollback process ... 175
Types of Storage Checkpoints ... 176
Storage Checkpoint space management considerations ... 178
About FileSnaps ... 178
Properties of FileSnaps ... 179
Concurrent I/O to FileSnaps ... 180
Copy-on-write and FileSnaps ... 180
Reading from FileSnaps ... 180
Block map fragmentation and FileSnaps ... 181
Backup and FileSnaps ... 181

Chapter 15 Considerations for Oracle point-in-time copies 182

Considerations for database layouts 182
Supported Oracle configurations 183
Considerations for multitenant database support 184
Behavior of clone databases 185

Chapter 16 Administering third-mirror break-off snapshots 186

Database FlashSnap for cloning 186
Database FlashSnap advantages 187
Preparing hosts and storage for Database FlashSnap 187
Setting up hosts 187
Creating a snapshot mirror of a volume or volume set used by the database 189
Creating a clone of a database by using Database FlashSnap 192
Resynchronizing mirror volumes with primary volumes 199
To reverse resynchronize the snapshot volume 201
Example outputs for reverse resync parameters 201
Recovering the clone database manually 203
Cloning a database on the secondary host 204

Chapter 17 Administering space-optimized snapshots 206

Planning to create an instant space-optimized snapshot 206
Preparing a legacy volume for the creation of an instant snapshot 207
Creating a shared cache object 208
Creating a clone of an Oracle database by using space-optimized snapshots 210
Space-optimized snapshots in HA environment 213
Creating multiple clones using FlashSnap snapshots 214
Recovering the clone database manually 220

Chapter 18 Administering Storage Checkpoints 221

About Storage Checkpoints 221
Database Storage Checkpoints for recovery 222
Advantages and limitations of Database Storage Checkpoints 223
Creating a Database Storage Checkpoint 223
Deleting a Database Storage Checkpoint 224

Mounting a Database Storage Checkpoint 224
Unmounting a Database Storage Checkpoint 225
Creating a database clone using a Database Storage Checkpoint 226
Restoring database from a Database Storage Checkpoint 226
Gathering data for offline-mode Database Storage Checkpoints 228

Chapter 19 Administering FileSnap snapshots 230

Preparing to use FileSnap 230
Creating a clone of an Oracle database by using FileSnap 230
Destroying a database clone created by using FileSnap 233
Destroying a FileSnap snapshot 234
Restoring database files from a FileSnap snapshot 235
Viewing the list of snapshots and clones created by using FileSnap 236

Chapter 20 Backing up and restoring with Netbackup in an SFHA environment 237

About Veritas NetBackup 237
How NetBackup block-level incremental backup works for Oracle database files 238
About using Veritas NetBackup for backup and restore for Oracle 238
About using Veritas NetBackup to backup and restore Oracle Disk Manager files 239
Using NetBackup to backup and restore Quick I/O files for Oracle 239
Using NetBackup in an SFHA Solutions product environment 241
Clustering a NetBackup Master Server 241
Backing up and recovering a VxVM volume using NetBackup 242
Recovering a VxVM volume using NetBackup 243
Backing up an Oracle database using NetBackup 243
Recovering an Oracle database using NetBackup 244

Section 6 Optimizing storage costs for Oracle 245

Chapter 21 Understanding storage tiering with SmartTier 246

About SmartTier 246
About VxFS multi-volume file systems 248
About VxVM volume sets 249
About volume tags 249
SmartTier file management 249
SmartTier sub-file object management 250
How SmartTier for Oracle works 250
How partitions change the way Oracle stores database objects 250
Database extents to file location 253
Statistics in Oracle 253
Optimizing storage for Oracle database objects 254
About Automatic Workload Repository statistics 255
SmartTier in a High Availability (HA) environment 256

Chapter 22 Configuring and administering SmartTier 257

Configuring SmartTier for Oracle 257
SmartTier for Oracle command requirements 258
Defining database parameters 259
Configuring storage classes 261
Converting a Veritas File System (VxFS) to a VxFS multi-volume file system 263
Classifying volumes into a storage class 264
Displaying free space on your storage class 265
Adding new volumes to a storage class 266
Removing volumes from a storage class 267
Optimizing database storage using SmartTier for Oracle 268
Relocating files manually using SmartTier for Oracle 268
Relocating tablespaces manually using SmartTier for Oracle 269
Relocating table partitions manually using SmartTier for Oracle 270
Setting policies to automate storage tiering using SmartTier for Oracle 271
Running reports using SmartTier for Oracle 272
Extent balancing in a database environment using SmartTier for Oracle 274

Extent balancing file system to imrove database performance 275
Creating an extent balanced file system using SmartTier for Oracle 276
Running sub-file database object reports using SmartTier for Oracle 278
Optimizing sub-file database object placement using SmartTier for Oracle 280

Chapter 23 SmartTier use cases for Oracle 283

SmartTier use cases for Oracle 283
SmartTier for Oracle file-based use cases 283
Migrating partitioned data and tablespaces 284
Scheduling the relocation of archive and Flashback logs 286
SmartTier for Oracle sub-file use cases 288
Using AWR statistics to identify sub-file objects for potential relocation 289
Relocating a hot database segment to a higher performance tier 291
Relocating old table data to a lower cost or secondary tier 292

Chapter 24 Compressing files and databases to optimize storage costs 293

Compressed files and databases 293
Supported database versions and environment 294
Compressing archive logs 294
Compressing read-only tablespaces 295
Compressing infrequently accessed table partitions 296
Compressing infrequently accessed datafiles 297
Best practices for compressing files in an Oracle database 298

Chapter 25 Using the Compression Advisor tool 299

About the Compression Advisor tool 299
Compressing Oracle archive logs using Compression Advisor 300
Displaying compression candidate data files using Compression Advisor 301
Compressing Oracle data files using Compression Advisor 302
Displaying compressed data files using Compression Advisor 303
Uncompressing Oracle data files using Compression Advisor 304
Compression Advisor command reference 305

Section 7 Managing Oracle disaster recovery 308

Chapter 26 Using Storage Foundation for Databases (SFDB) tools in a replicated environment 309

About deploying SF Databases tools with Volume Replicator (VVR) 309
Using database accelerators for the Oracle database in a Volume Replicator (VVR) environment 310
About deploying Storage Foundation for Databases (SFDB) tools with Oracle Data Guard 311

Chapter 27 Using volume snapshots in a replicated environment 313

Considerations for using volume snapshots with VVR 313
Creating space-optimized snapshots in a VVR setup 314
Creating third-mirror break-off snapshots in a VVR setup 317
Considerations for using vxsfadm after replication role transfer 321

Section 8 Storage Foundation for Databases administrative reference 322

Chapter 28 Storage Foundation for Databases command reference 323

vxsfadm command reference 323
FlashSnap reference 327
FlashSnap configuration parameters 327
FlashSnap supported operations 329
Space-optimized snapshots reference 331
Space-optimized snapshots configuration parameters 331
Space-optimized snapshots supported operations 333
FileSnap reference 334
FileSnap configuration parameters 334
FileSnap supported operations 335
Database Storage Checkpoints reference 336
Database Storage Checkpoints configuration parameters 336
Database Storage Checkpoints supported operations 339

Chapter 29 Tuning for Storage Foundation for Databases 341

Additional documentation 341
About tuning Veritas Volume Manager (VxVM) 342
About obtaining volume I/O statistics 342
About tuning VxFS 343
How monitoring free space works 344
How tuning VxFS I/O parameters works 345
About tunable VxFS I/O parameters 345
About obtaining file I/O statistics using the Quick I/O interface 350
About I/O statistics data 350
Obtaining file I/O statistics using Veritas extension for Oracle Disk Manager 352
About I/O statistics 353
About tuning Oracle databases 354
Sequential table scans 354
Sequential table scans 355
Asynchronous I/O 355
Tuning buffer cache 356
Tuning buffer cache 356
Setting Oracle block reads during sequential scans 356
Setting slave parameters 358
Configuring memory allocation 358
About tuning AIX Virtual Memory Manager 359
About tuning Solaris for Oracle 362
maxuprc 362
shmmax 363
shmmin 363
shmmni 363
shmseg 363
semmap 363
semmni 363
semmns 363
semmnu 364
semmsl 364

Chapter 30 Troubleshooting SFDB tools 365

About troubleshooting Storage Foundation for Databases (SFDB) tools 365
Running scripts for engineering support analysis for SFDB tools 366

Storage Foundation for Databases (SFDB) tools log files 366
About the vxdbd daemon 366
Starting and stopping vxdbd 366
Configuring listening port for the vxdbd daemon 367
Limiting vxdbd resource usage 367
Configuring encryption ciphers for vxdbd 368
Troubleshooting vxdbd 368
Resources for troubleshooting SFDB tools 369
SFDB logs 369
SFDB error messages 370
SFDB repository and repository files 370
Troubleshooting SmartTier for Oracle 371
Upgrading Storage Foundation for Databases (SFDB) tools from 5.0.x to 7.4.2 (2184482) 372
Troubleshooting Reverse Resynchronization 373

Chapter 31 Manual recovery of Oracle database 375

Manual recovery of a clone database 375
Manual recovery after a restore operation 376

Chapter 32 Storage Foundation for Databases command reference for the releases prior to 6.0 378

About SFDB commands backward compatibility 378
Storage Foundation for Databases (SFDB) tools features which are no longer supported 379
Preparing storage for Database FlashSnap 379
Example procedure to create a snapshot mirror of a volume 379
Upgrading existing volumes to use Veritas Volume Manager 7.4.2 383
Updating after structural changes to the database 391
About creating database snapshots 391
Online database snapshots 391
Tasks before creating a snapshot 392
Creating a snapshot 393
Tasks after creating a snapshot 395
FlashSnap commands 400
Creating a snapplan (dbed_vmchecksnap) 401
Validating a snapplan (dbed_vmchecksnap) 411
Displaying, copying, and removing a snapplan (dbed_vmchecksnap) 413
Creating a snapshot (dbed_vmsnap) 415

Backing up the database from snapshot volumes (dbed_vmclonedb) 418
Cloning a database (dbed_vmclonedb) 422
Resynchronizing the snapshot to your database 430
Removing a snapshot volume 431
Guidelines for Oracle recovery 432
Back up all control files before Storage Rollback 432
Ensure that the control files are not rolled back 433
Ensure that all archived redo logs are available 433
Media recovery procedures 434
Database Storage Checkpoint Commands 435
Creating or updating the repository using dbed_update 436
Creating Storage Checkpoints using dbed_ckptcreate 437
Displaying Storage Checkpoints using dbed_ckptdisplay 439
Mounting Storage Checkpoints using dbed_ckptmount 441
Unmounting Storage Checkpoints using dbed_ckptumount 442
Performing Storage Rollback using dbed_ckptrollback 443
Removing Storage Checkpoints using dbed_ckptremove 444
Cloning the Oracle instance using dbed_clonedb 445

Section 9 Reference 448

Appendix A VCS Oracle agents 449

VCS agents for Oracle 449
Oracle agent functions 449
Resource type definition for the Oracle agent 453
Netlsnr agent functions 458
Resource type definition for the Netlsnr agent 459

Appendix B Sample configuration files for clustered deployments 462

About sample configuration files 462
Sample configuration file for Storage Foundation for High Availability 463
Sample configuration file for Storage Foundation for Clustered File System HA 465
Sample configuration file for Storage Foundation for Oracle RAC 471

Appendix C Database FlashSnap status information 477

About Database FlashSnap status information 477
Database FlashSnap Snapshot status information from the CLI 477

Snapshot status information from the CLI 478
Snapshot database status information from the CLI 480

Appendix D Using third party software to back up files 481

About using third party software to back up files 481
Using third party software to back up files 481
Backing up and restoring Oracle Disk Manager files using Oracle RMAN 481

Section 1

Storage Foundation High Availability (SFHA) management solutions for Oracle databases

- Chapter 1. Overview of Storage Foundation for Databases

Chapter 1

Overview of Storage Foundation for Databases

This chapter includes the following topics:

- Introducing Storage Foundation High Availability (SFHA) Solutions for Oracle
- About Veritas File System
- About Veritas Volume Manager
- About Dynamic Multi-Pathing (DMP)
- About Cluster Server
- About Cluster Server agents
- About Veritas InfoScale Operations Manager
- Feature support for Oracle across Veritas InfoScale 7.4.2 products
- Use cases for Veritas InfoScale products

Introducing Storage Foundation High Availability (SFHA) Solutions for Oracle

This guide documents the deployment and key use cases of the SFDB tools with Storage Foundation High Availability (SFHA) Solutions products in Oracle database environments. It is a supplemental guide to be used in conjunction with SFHA Solutions product guides.

The Storage Foundation for Databases tools provide enhanced management options for Oracle databases. The SFDB tools provide enhanced ease-of-use commands which can be run by a database administrator without root privileges to optimize

storage for an Oracle database environment. This guide documents the deployment and use of the SFDB tools included with SFHA Solutions enterprise products.

Note: The commands used for the Red Hat Enterprise Linux (RHEL) operating system in this document also apply to supported RHEL-compatible distributions.

About Veritas File System

A file system is simply a method for storing and organizing computer files and the data they contain to make it easy to find and access them. More formally, a file system is a set of abstract data types (such as metadata) that are implemented for the storage, hierarchical organization, manipulation, navigation, access, and retrieval of data.

Veritas File System (VxFS) was the first commercial journaling file system. With journaling, metadata changes are first written to a log (or journal) then to disk. Since changes do not need to be written in multiple places, throughput is much faster as the metadata is written asynchronously.

VxFS is also an extent-based, intent logging file system. VxFS is designed for use in operating environments that require high performance and availability and deal with large amounts of data.

The maximum size of the file system you can create depends on the block size.

Block Size	Currently-Supported Maximum File System Size
1024 bytes	68,719,472,624 sectors (≈32 TB)
2048 bytes	137,438,945,248 sectors (≈64 TB)
4096 bytes	274,877,890,496 sectors (≈128 TB)
8192 bytes	549,755,780,992 sectors (≈256 TB)

VxFS major components include:

File system logging	About the Veritas File System intent log
Extents	About extents
File system disk layouts	About file system disk layouts

About the Veritas File System intent log

Most file systems rely on full structural verification by the `fsck` utility as the only means to recover from a system failure. For large disk configurations, this involves a time-consuming process of checking the entire structure, verifying that the file system is intact, and correcting any inconsistencies. VxFS provides fast recovery with the VxFS intent log and VxFS intent log resizing features.

VxFS reduces system failure recovery times by tracking file system activity in the VxFS intent log. This feature records pending changes to the file system structure in a circular intent log. The intent log recovery feature is not readily apparent to users or a system administrator except during a system failure. By default, VxFS file systems log file transactions before they are committed to disk, reducing time spent recovering file systems after the system is halted unexpectedly.

During system failure recovery, the VxFS `fsck` utility performs an intent log replay, which scans the intent log and nullifies or completes file system operations that were active when the system failed. The file system can then be mounted without requiring a full structural check of the entire file system. Replaying the intent log might not completely recover the damaged file system structure if there was a disk hardware failure; hardware problems might require a complete system check using the `fsck` utility provided with VxFS.

The `mount` command automatically runs the VxFS `fsck` command to perform an intent log replay if the mount command detects a dirty log in the file system. This functionality is only supported on a file system mounted on a Veritas Volume Manager (VxVM) volume, and is supported on cluster file systems.

See the `fsck_vxfs`(1M) manual page and `mount_vxfs`(1M) manual page.

The VxFS intent log is allocated when the file system is first created. The size of the intent log is based on the size of the file system—the larger the file system, the larger the intent log. You can resize the intent log at a later time by using the `fsadm` commnad.

See the `fsadm_vxfs`(1M) manual page.

The maximum default intent log size for disk layout Version 7 or later is 256 megabytes.

Note: Inappropriate sizing of the intent log can have a negative impact on system performance.

About extents

An extent is a contiguous area of storage in a computer file system, reserved for a file. When starting to write to a file, a whole extent is allocated. When writing to the file again, the data continues where the previous write left off. This reduces or eliminates file fragmentation. An extent is presented as an address-length pair, which identifies the starting block address and the length of the extent (in file system or logical blocks). Since Veritas File System (VxFS) is an extent-based file system, addressing is done through extents (which can consist of multiple blocks) rather than in single-block segments. Extents can therefore enhance file system throughput.

Extents allow disk I/O to take place in units of multiple blocks if storage is allocated in contiguous blocks. For sequential I/O, multiple block operations are considerably faster than block-at-a-time operations; almost all disk drives accept I/O operations on multiple blocks.

Extent allocation only slightly alters the interpretation of addressed blocks from the inode structure compared to block-based inodes. A VxFS inode references 10 direct extents, each of which are pairs of starting block addresses and lengths in blocks.

Disk space is allocated in 512-byte sectors to form logical blocks. VxFS supports logical block sizes of 1024, 2048, 4096, and 8192 bytes. The default block size is 1 KB for file system sizes of up to 2 TB, and 8 KB for file system sizes 2 TB or larger.

About file system disk layouts

The disk layout is the way file system information is stored on disk. On Veritas File System (VxFS), several disk layout versions are supported to provide new features and specific UNIX environments.

You can use one of the following commands to upgrade the disk layout version.

`vxupgrade`	Upgrades an existing VxFS file system to a supported disk layout version while the file system remains online. See the `vxupgrade`(1M) manual page.
`vxfsconvert`	Upgrades a no-longer supported disk layout version to a supported version while the file system is not mounted. The `vxfsconvert` command can also be used to convert a native file system (ext2, ext3, and ext4) to VxFS, while the file system is not mounted. See the `vxfsconvert`(1M) manual page.

Table 1-1 lists the supported disk layout versions.

Table 1-1 Supported disk layout versions

Version	Supported features
Version 12	Version 12 supports 128 node cluster on CFS
Version 13	■ Added support for WORM ■ Clone creation performance improvement when extended file attributes are used
Version 14	Supports SmartIO FEL-based caching
Version 15	■ Performance enhancements in SELinux attribute storage and retrieval ■ Secure Clock support for WORM files
Version 16	Audit Logging support for WORM files

Currently, only versions 12, 13, 14, 15, and 16 can be created and mounted. Versions 6, 7, 8, 9, 10, and 11 can be mounted, but only for upgrading to a supported version.

About Veritas Volume Manager

Veritas™ Volume Manager (VxVM) by Veritas is a storage management subsystem that allows you to manage physical disks and logical unit numbers (LUNs) as logical devices called volumes. A VxVM volume appears to applications and the operating system as a physical device on which file systems, databases, and other managed data objects can be configured.

VxVM provides easy-to-use online disk storage management for computing environments and Storage Area Network (SAN) environments. By supporting the Redundant Array of Independent Disks (RAID) model, VxVM can be configured to protect against disk and hardware failure, and to increase I/O throughput. Additionally, VxVM provides features that enhance fault tolerance and fast recovery from disk failure or storage array failure.

VxVM overcomes restrictions imposed by hardware disk devices and by LUNs by providing a logical volume management layer. This allows volumes to span multiple disks and LUNs.

VxVM provides the tools to improve performance and ensure data availability and integrity. You can also use VxVM to dynamically configure storage while the system is active.

About Dynamic Multi-Pathing (DMP)

Dynamic Multi-Pathing (DMP) provides multi-pathing functionality for the operating system native devices that are configured on the system. DMP creates DMP metadevices (also known as DMP nodes) to represent all the device paths to the same physical LUN.

DMP metadevices support the OS native logical volume manager (LVM). You can create LVM volumes and volume groups on DMP metadevices.

DMP supports the LVM volume devices that are used as the paging devices.

DMP metadevices support ZFS. You can create ZFS pools on DMP metadevices. DMP supports both root and non-root ZFS pools. For versions prior to Solaris 11 update 1, DMP supports only non-root ZFS file systems.

Veritas Volume Manager (VxVM) volumes and disk groups can co-exist with LVM volumes and volume groups. But, each device can only support one of the types. If a disk has a VxVM label, then the disk is not available to LVM. Similarly, if a disk is in use by LVM, then the disk is not available to VxVM.

Veritas Volume Manager (VxVM) volumes and disk groups can co-exist with ZFS pools, but each device can only support one of the types. If a disk has a VxVM label, then the disk is not available to ZFS. Similarly, if a disk is in use by ZFS, then the disk is not available to VxVM.

About Cluster Server

Cluster Server (VCS) is a clustering solution that provides the following benefits:

- Minimizes downtime.
- Facilitates the consolidation and the failover of servers.
- Effectively manages a wide range of applications in heterogeneous environments.

Before you install the product, read the *Veritas InfoScale 7.4.2 Release Notes*.

To configure the product, follow the instructions in the *Cluster Server Generic Application Agent Configuration Guide.*

About Cluster Server agents

Veritas InfoScale agents provide high availability for specific resources and applications. Each agent manages resources of a particular type. For example, the Oracle agent manages Oracle databases. Typically, agents start, stop, and monitor resources and report state changes.

Before you install VCS agents, review the configuration guide for the agent.

In addition to the agents that are provided in this release, other agents are available through an independent Veritas InfoScale offering called the Cluster Server Agent Pack. The agent pack includes the currently shipping agents and is re-released quarterly to add the new agents that are now under development.

Contact your Veritas InfoScale sales representative for the following details:

- Agents that are included in the agent pack
- Agents under development
- Agents available through Veritas InfoScale Consulting Services

You can download the latest agents from the Veritas Services and Operations Readiness Tools website:

http://sort.veritas.com/agents

About Veritas InfoScale Operations Manager

Veritas InfoScale Operations Manager provides a centralized management console for Veritas InfoScale products. You can use Veritas InfoScale Operations Manager to monitor, visualize, and manage storage resources and generate reports.

Veritas recommends using Veritas InfoScale Operations Manager to manage Storage Foundation and Cluster Server environments.

You can download Veritas InfoScale Operations Manager from https://sort.veritas.com/.

Refer to the Veritas InfoScale Operations Manager documentation for installation, upgrade, and configuration instructions.

The Veritas Enterprise Administrator (VEA) console is no longer packaged with Veritas InfoScale products. If you want to continue using VEA, a software version is available for download from https://www.veritas.com/product/storage-management/infoscale-operations-manager. Storage Foundation Management Server is deprecated.

Feature support for Oracle across Veritas InfoScale 7.4.2 products

Veritas InfoScale solutions and use cases for Oracle are based on the shared management features of Veritas InfoScale Storage Foundation and High Availability (SFHA) Solutions products. Clustering features are available separately through Cluster Server (VCS) as well as through the SFHA Solutions products.

Table 1-2 lists the features supported across SFHA Solutions products. Table 1-3 lists the high availability and disaster recovery features available in VCS.

Table 1-2 Storage management features in Veritas InfoScale products

Storage management feature	Veritas InfoScale Foundation	Veritas InfoScale Storage	Veritas InfoScale Availability	Veritas InfoScale Enterprise
Veritas Extension for Oracle Disk Manager	N	Y **Note:** Supported in the cluster-exclusive mode which allows access to the feature only from a single node.	N	Y
Veritas Extension for Cached Oracle Disk Manager **Note:** Not supported for Oracle RAC.	N	Y	N	Y
Quick I/O **Note:** Not supported on Linux	N	Y	N	Y
Cached Quick I/O **Note:** Not supported on Linux	N	Y	N	Y
Compression	N	Y	N	Y
Deduplication	N	Y	N	Y
Flexible Storage Sharing	N	Y	N	Y
SmartIO **Note:** SFRAC does not support Writeback caching.	N	Y	N	Y
SmartMove	N	Y	N	Y
SmartTier for Oracle	N	Y	N	Y
Thin Reclamation	N	Y	N	Y

Table 1-2 Storage management features in Veritas InfoScale products *(continued)*

Storage management feature	Veritas InfoScale Foundation	Veritas InfoScale Storage	Veritas InfoScale Availability	Veritas InfoScale Enterprise
Portable Data Containers	N	Y	N	Y
Database FlashSnap	N	Y	N	Y
Database Storage Checkpoints	N	Y	N	Y
FileSnap	N	Y	N	Y
Volume replication	N	Y	N	Y
File replication **Note:** Supported on Linux only.	N	Y	N	Y
Advanced support for virtual storage	Y	Y	Y	Y
Clustering features for high availability (HA)	N	N	Y	N
Disaster recovery features (HA/DR)	N	N	Y	N
Dynamic Multi-pathing	Y	Y	Y	Y

Table 1-3 Availability management features in Veritas InfoScale solutions products

Availability management feature	VCS HA/DR
Clustering for high availability (HA)	Y
Database and application/ISV agents	Y
Advanced failover logic	Y
Data integrity protection with I/O fencing	Y
Advanced virtual machines support	Y
Virtual Business Services	Y
Replication agents	Y

Table 1-3 Availability management features in Veritas InfoScale solutions products *(continued)*

Availability management feature	VCS HA/DR
Replicated Data Cluster	Y
Campus (stretch) cluster	Y
Global clustering (GCO)	Y
Fire Drill	Y

Notes:

- Y=Feature is included in your license.
- N=Feature is not supported with your license.

Notes:

- SmartTier for Oracle is an expanded and renamed version of Dynamic Storage Tiering (DST).
- All features listed in Table 1-2 and Table 1-3 are supported on AIX, Linux, Solaris except as noted. Consult specific product documentation for information on supported operating systems.

Use cases for Veritas InfoScale products

Veritas InfoScale Storage Foundation and High Availability (SFHA) Solutions product components and features can be used individually and in concert to improve performance, resilience and ease of management for your storage and applications. This guide documents key use cases for the management features of SFHA Solutions products.

Note: The commands used for the Red Hat Enterprise Linux (RHEL) operating system in this document also apply to supported RHEL-compatible distributions.

Table 1-4 Key use cases for SFHA Solutions products

Use case	Veritas InfoScale feature
Improve Oracle database performance using SFHA Solutions database accelerators to enable your database to achieve the speed of raw disk while retaining the management features and convenience of a file system. See "About Veritas InfoScale product components database accelerators" on page 80.	Veritas Extension for Oracle Disk Manager See "About Oracle Disk Manager in the Veritas InfoScale products environment" on page 83. Veritas Extension for Cached Oracle Disk Manager See "About Cached ODM in SFHA environment" on page 96. **Note:** Cached ODM is not supported for Oracle RAC Quick I/O See "About Quick I/O" on page 116. Cached Quick I/O See "About Cached Quick I/O" on page 141. **Note:** Quick I/O and Cached Quick I/O are not supported on Linux.
Protect your data using SFHA Solutions Flashsnap, Storage Checkpoints, and NetBackup point-in-time copy methods to back up and recover your data. See "About point-in-time copies" on page 155.	FlashSnap Storage Checkpoints NetBackup with SFHA Solutions
Process your data off-host to avoid performance loss to your production hosts by using SFHA Solutions volume snapshots.	FlashSnap
Optimize copies of your production database for test, decision modeling, and development purposes by using SFHA Solutions point-in-time copy methods.	FlashSnap
Make file level point-in-time snapshots using SFHA Solutions space-optimized FileSnap when you need finer granualarity for your point-in-time copies than file systems or volumes. You can use FileSnap for cloning virtual machines.	FileSnap

Table 1-4 Key use cases for SFHA Solutions products *(continued)*

Use case	Veritas InfoScale feature
Maximize your storage utilization using SFHA Solutions SmartTier to move data to storage tiers based on age, priority, and access rate criteria. See "About SmartTier" on page 246.	SmartTier
Maximize storage utilization for data redundancy, high availability, and disaster recovery, without physically shared storage.	Flexible Storage Sharing
Improve your data efficiency on solid state drives (SSDs) through I/O caching using advanced, customizable hueristics to determine which data to cache and how that data gets removed from the cache.	SmartIO read caching for applications running on VxVM volumes SmartIO read caching for applications running on VxFS file systems SmartIO write caching for applications running on VxFS file systems SmartIO caching for databases on VxFS file systems SmartIO caching for databases on VxVM volumes SmartIO write-back caching for databases is not supported on SFRAC See the *Veritas InfoScale 7.4.2 SmartIO for Solid-State Drives Solutions Guide*.
Plan a maintenance of virtual machines in a vSphere environment for a planned failover and recovery of application during unplanned failure using the Just In Time Availability solution.	Just In Time Availability solution
Improve the native and optimized format of your storage devices using the Veritas InfoScale solution which provides support with the advanced format or 4K (4096 bytes) sector devices (formatted with 4KB) in storage environments.	Veritas InfoScale 4K sector device support solution

Table 1-4 Key use cases for SFHA Solutions products *(continued)*

Use case	Veritas InfoScale feature
Multiple parallel applications in a data warehouse that require flexible sharing of data such as ETL pipeline, where output of one stage becomes input for the next stage. (for example, accounting system needs to combine data from different applications such as sales, payroll and purchasing)	Verita InfoScale application isolation More information: Application isolation in CVM environments with disk group sub-clustering Enabling the application isolation feature in CVM environments Disabling the application isolation feature in a CVM cluster Setting the sub-cluster node preference value for master failover Changing the disk group master manually For information, see the *Storage Foundation Cluster File System High Availability Administrator's Guide*.
Relax complete zoning requirement of SAN storage to all CVM nodes. This enables merging of independent clusters for better manageability.	Verita InfoScale application isolation More information: Application isolation in CVM environments with disk group sub-clustering Enabling the application isolation feature in CVM environments Disabling the application isolation feature in a CVM cluster Setting the sub-cluster node preference value for master failover Changing the disk group master manually For information, see the *Storage Foundation Cluster File System High Availability Administrator's Guide*.

Table 1-4 Key use cases for SFHA Solutions products *(continued)*

Use case	Veritas InfoScale feature
Enabling multiple independent clustered applications to use a commonly shared pool of scalable DAS storage. This facilitates adding of storage-only nodes to cluster for growing storage capacity and compute nodes for dedicated application use.	Verita InfoScale application isolation More information: Application isolation in CVM environments with disk group sub-clustering Enabling the application isolation feature in CVM environments Disabling the application isolation feature in a CVM cluster Setting the sub-cluster node preference value for master failover Changing the disk group master manually For information, see the *Storage Foundation Cluster File System High Availability Administrator's Guide*.

Section 2

Deploying Oracle with Veritas InfoScale products

- Chapter 2. Deployment options for Oracle in a Storage Foundation environment
- Chapter 3. Deploying Oracle with Storage Foundation
- Chapter 4. Deploying Oracle in an off-host configuration with Storage Foundation
- Chapter 5. Deploying Oracle with High Availability
- Chapter 6. Deploying Oracle with Volume Replicator (VVR) for disaster recovery

Chapter 2

Deployment options for Oracle in a Storage Foundation environment

This chapter includes the following topics:

- Oracle deployment options in a Storage Foundation environment
- Oracle single instance in a Storage Foundation environment
- Single instance Oracle with off-host in a Storage Foundation environment
- Single instance Oracle in a highly available cluster with Storage Foundation High Availability
- Single instance Oracle in a parallel cluster with SF Cluster File System HA environment
- About Oracle RAC in a Storage Foundation for Oracle RAC environment
- About Oracle in a replicated Storage Foundation environment
- Deploying Oracle and Storage Foundation in a virtualization environment
- Deploying Oracle with Storage Foundation SmartMove and Thin Provisioning

Oracle deployment options in a Storage Foundation environment

You can deploy Oracle with Veritas InfoScale Storage Foundation and High Availability products in the following setups:

- Oracle single instance in a Storage Foundation environment
- Oracle single instance and Oracle RAC with off-host in a Storage Foundation environment
- Oracle single instance with High Availability (HA)
- Oracle RAC with Storage Foundation for Oracle RAC
- Oracle single instance with SFCFSHA
- Oracle single instance with Veritas Volume Replicator (VVR) or Oracle Data Guard for disaster recovery
- Oracle single instance with Storage Foundation in a virtual environment

Storage Foundation for Databases (SFDB) tools support all of these setups.

Oracle single instance in a Storage Foundation environment

If you are deploying single instance Oracle with Storage Foundation, your setup configuration will reflect the following conditions:

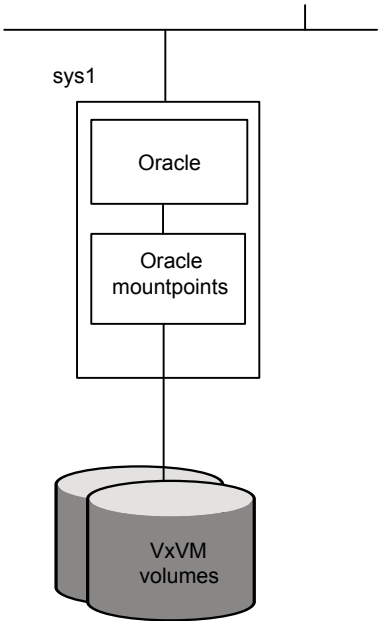
- A single instance of Oracle is set up on *sys1* with SF.
- The database and datafiles are online on *sys1*.
- You must run the SFDB tools commands on *sys1*.

For information about Storage Foundation for Databases (SFDB) repository or repository database:

See "About the Storage Foundation for Databases (SFDB) repository" on page 66.

Figure 2-1 shows an Oracle single instance deployment in a Storage Foundation environment.

Figure 2-1 Oracle single instance with Storage Foundation



Single instance Oracle with off-host in a Storage Foundation environment

If you are deploying single instance Oracle with Storage Foundation in an off-host setup, your configuration will reflect the following conditions:

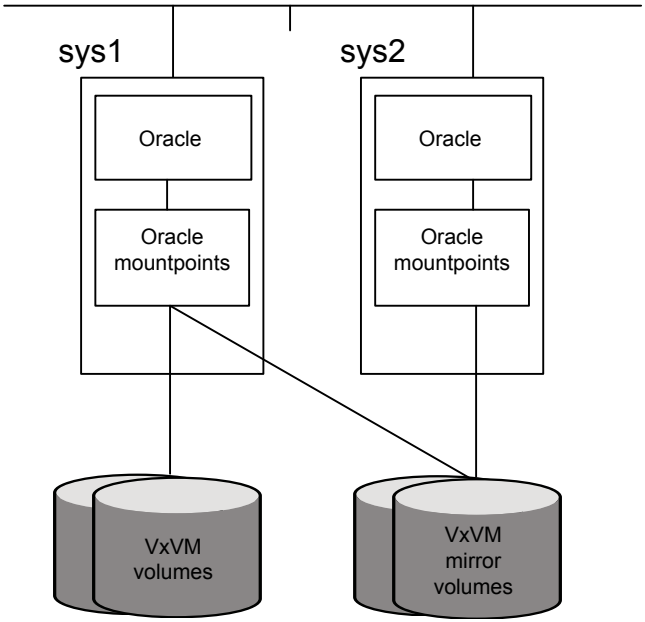
- A single instance of Oracle is set up on *sys1* with SF.
- The database and datafiles are online on *sys1*.
- *sys1* and *sys2* share the same storage.

For information about Storage Foundation for Databases (SFDB) repository or repository database:

See "About the Storage Foundation for Databases (SFDB) repository" on page 66.

Figure 2-2 shows an Oracle single instance deployment with off-host in a Storage Foundation environment.

Figure 2-2 Oracle single instance off-host setup with Storage Foundation environment



Single instance Oracle in a highly available cluster with Storage Foundation High Availability

If you are deploying Oracle with Storage Foundation High Availability (SFHA), your setup configuration will reflect the following conditions:

- A highly available single instance of Oracle is set up on *sys1* and *sys2* with SFHA.
- The database and datafiles are online on *sys1*.

- You must run the SFDB tools commands on *sys1* where the database is online.

For information about Storage Foundation for Databases (SFDB) repository or repository database:

See "About the Storage Foundation for Databases (SFDB) repository" on page 66.

Figure 2-3 and Figure 2-4 show an Oracle single instance in a Storage Foundation High Availability environment and a failover scenario in such a setup.

Figure 2-3 Oracle single instance with SFHA

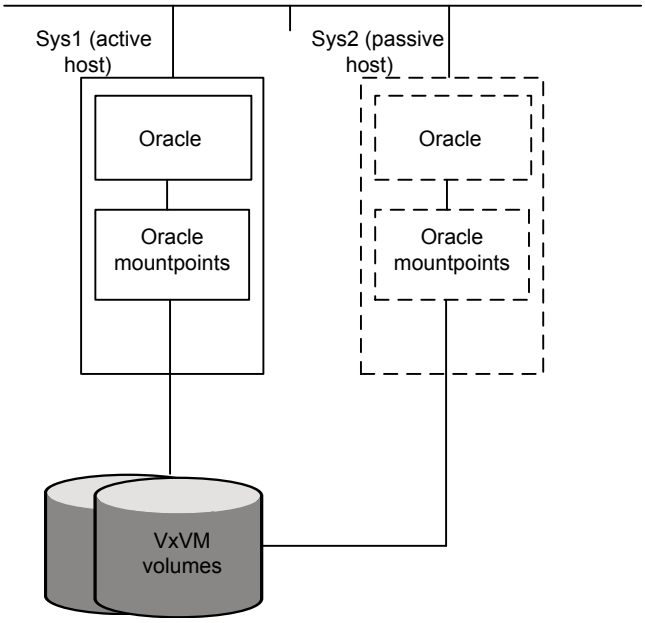
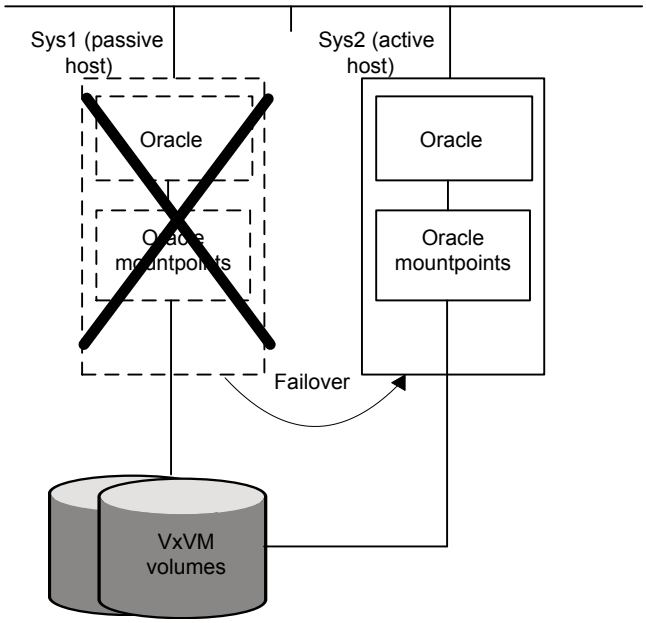


Figure 2-4 Oracle single instance with SFHA failover setup



Single instance Oracle in a parallel cluster with SF Cluster File System HA environment

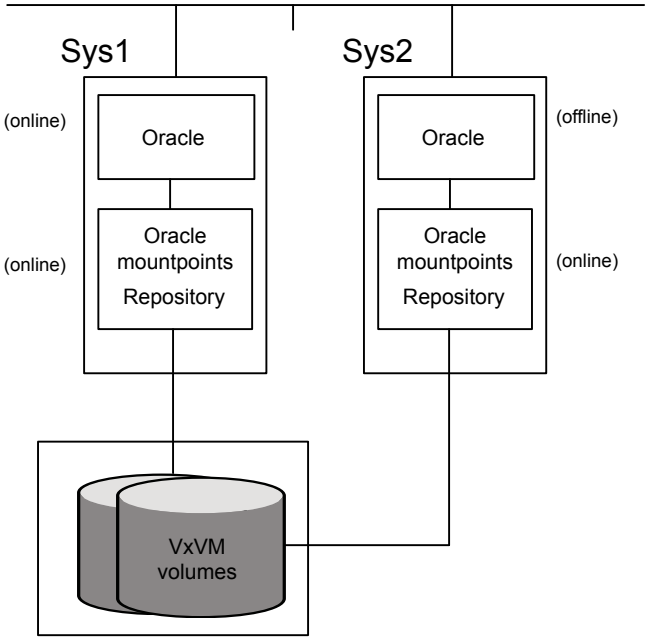
If you are deploying Oracle single instance with SF Cluster File System HA, your setup configuration will reflect the following conditions:

- A highly available parallel cluster with a Oracle is set up on *sys1* and *sys2* with SF Cluster File System HA.
- The database is online on *sys1*.
- The datafiles are mounted and shared on *sys1* and *sys2*.
- The database repository is mounted and shared on *sys1* and *sys2*.

- The SFDB tools commands will fail on *sys2*.

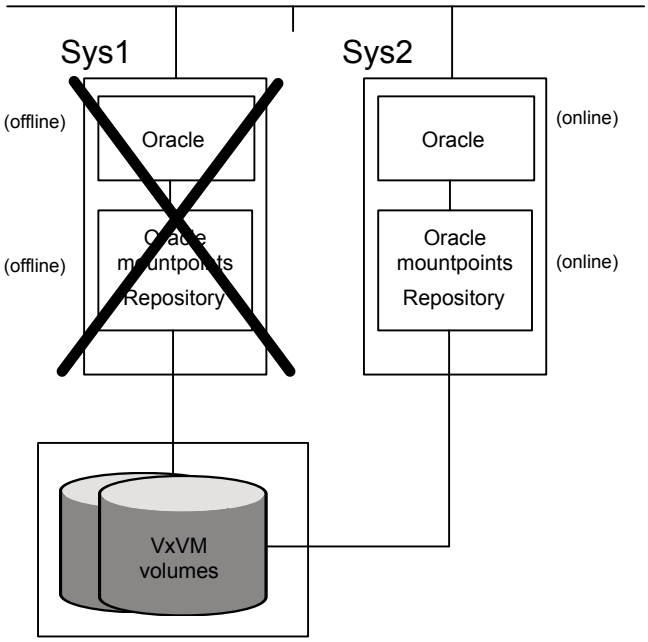
In the figures below the repository directory resides in the Oracle mount points.

Figure 2-5 SF Cluster File System HA setup example



The failover to the backup system is automatic rather than manual for SF Cluster File System HA.

Figure 2-6 SFCFSHA HA failover example



About Oracle RAC in a Storage Foundation for Oracle RAC environment

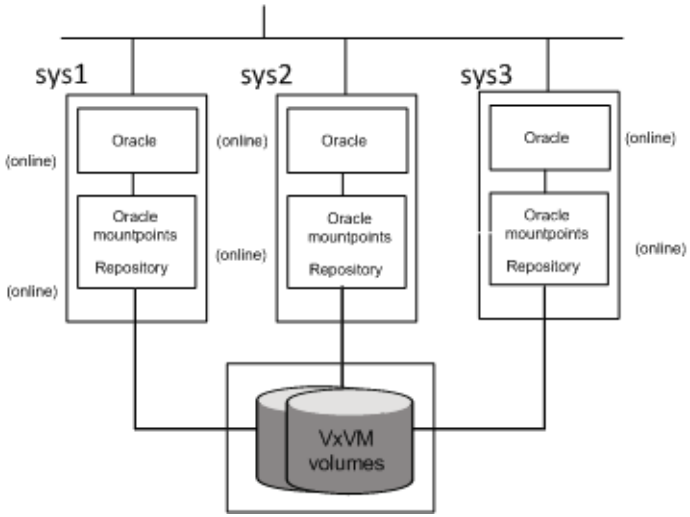
If you are deploying Oracle RAC with Storage Foundation for Oracle RAC, your setup configuration will reflect the following conditions:

- A highly available parallel cluster with a multiple instances of Oracle is set up on *sys1, sys2*, and *sys3* with SF for Oracle RAC.
- The database is online on *sys1*, *sys2*, and *sys3*.
- The datafiles are mounted and shared on *sys1*, *sys2*, and *sys3*.
- The SFDB tools is mounted and shared on *sys1*, *sys2*, and *sys3*.

- Clustered ODM is supported for this configuration.

In the figure below the repository directory resides in the Oracle mount points.

Figure 2-7 Oracle RAC in a Storage Foundation for Oracle RAC environment



For an SF Oracle RAC configuration, the systems are online in parallel and do not use failover mechanisms within the cluster.

See the *Veritas InfoScale Storage Foundation for Oracle RAC Administrator's Guide*.

About Oracle in a replicated Storage Foundation environment

In an Oracle environment, there are two supported replication options: Veritas Volume Replicator (VVR) and Oracle Data Guard.

Storage Foundation for Databases (SFDB) tools support these replication tools in the single instance mode on the secondary site.

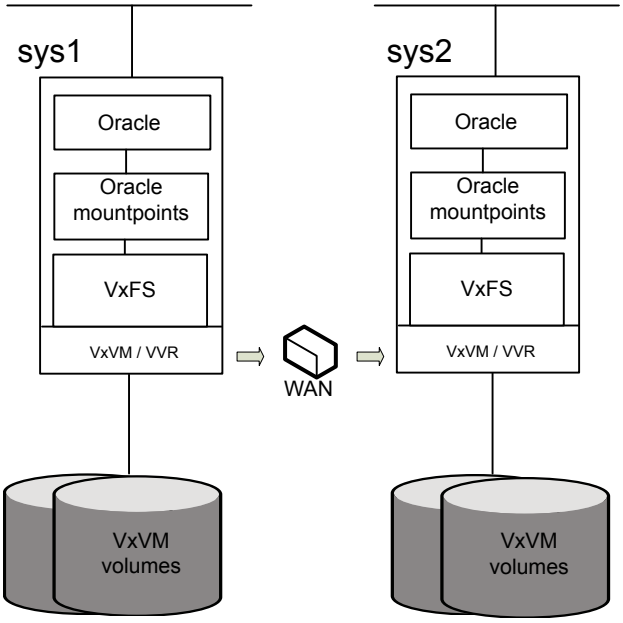
If you are deploying Oracle single instance in a Storage Foundation environment with VVR for disaster recovery, your setup configuration will reflect the following conditions:

- A single instance of Oracle is set up on *sys1*.
- The database and datafiles are online on *sys1*.
- `ORACLE_HOME` is set up on *sys2*.

- The data volumes on *sys1* are set up for replication to *sys2*.
- A consistent copy of the application data on the *sys1* volumes is available on *sys2*.

Figure 2-8 shows an Oracle single instance deployment in a Storage Foundation environment with VVR.

Figure 2-8 Oracle single instance - Storage Foundation with VVR



Deploying Oracle and Storage Foundation in a virtualization environment

If you are deploying the Storage Foundation for Databases (SFDB) tools in a virtual machine environment, the following are supported:

- WPARs on AIX

- VMware on Linux
- Oracle VM Server for SPARC on Solaris

Deploying Oracle with Storage Foundation SmartMove and Thin Provisioning

You can use SmartMove and Thin Provisioning with Storage Foundation products and your Oracle database.

When data files are deleted, you can reclaim the storage space used by these files if the underlying devices are thin reclaimable LUNs. For this purpose, use the Storage Foundation Thin Reclamation feature.

See the *Storage Foundation Administrator's Guide*.

Chapter 3

Deploying Oracle with Storage Foundation

This chapter includes the following topics:

- Tasks for deploying Oracle databases
- Planning your Oracle storage
- About selecting a volume layout for deploying Oracle
- Setting up disk group for deploying Oracle
- Creating volumes for deploying Oracle
- Creating VxFS file system for deploying Oracle
- Mounting the file system for deploying Oracle
- Installing Oracle and creating database

Tasks for deploying Oracle databases

If you are deploying an Oracle single instance database in a Storage Foundation environment, complete these tasks in the order listed below:

Determine the number and sizes of file systems you need for the database you want to create.	See "Planning your Oracle storage" on page 47.

Create a volume layout.	See "About selecting a volume layout for deploying Oracle" on page 48. See "Setting up disk group for deploying Oracle" on page 48. See "Disk group configuration guidelines for deploying Oracle" on page 50. See "Creating volumes for deploying Oracle" on page 50. See "Volume configuration guidelines for deploying Oracle" on page 51.
Create and mount file systems.	See "Creating VxFS file system for deploying Oracle" on page 51. See "File system creation guidelines for deploying Oracle" on page 52. See "Mounting the file system for deploying Oracle" on page 53.
Install Oracle and create database.	See "Installing Oracle and creating database" on page 54.

Planning your Oracle storage

Determine the number of file systems and size of each file system depending upon the size of your database. Veritas recommends creating the following file systems for your Oracle database.

- File system for Oracle Home
  Oracle binaries are installed on this file system.
- File system for data files
  Data files are stored on this file system. The size of the file system depends upon the size of your database. You need to create volumes to meet your file system needs. Disk mirrors can be used to safeguard against disk failures and striping can be used for better performance.
  Depending upon factors such as database types, I/O patterns, indexes, and temp files, you can create one or more file systems for storing data files.
- File system for redo logs
  Redo logs are stored on this file system. It is preferable to create the file system for redo logs on fast storage devices for better performance.
- File system for archived redo logs

Archived redo logs are stored on this file system.

About selecting a volume layout for deploying Oracle

Veritas Volume Manager (VxVM) offers a variety of layouts that enables you to configure your database to meet performance and availability requirements. The proper selection of volume layouts provides optimal performance for the database workload.

An important factor in database performance is the tablespace placement on the disks.

Disk I/O is one of the most important determining factors of database performance. Having a balanced I/O load usually means optimal performance. Designing a disk layout for the database objects to achieve balanced I/O is a crucial step in configuring a database. When deciding where to place tablespaces, it is often difficult to anticipate future usage patterns. VxVM provides flexibility in configuring storage for the initial database set up and for continual database performance improvement as needs change. VxVM can split volumes across multiple drives to provide a finer level of granularity in data placement. By using striped volumes, I/O can be balanced across multiple disk drives. For most databases, ensuring that different containers or tablespaces, depending on database, are distributed across the available disks may be sufficient.

Striping also helps sequential table scan performance. When a table is striped across multiple devices, a high transfer bandwidth can be achieved by setting the Oracle parameter `DB_FILE_MULTIBLOCK_READ_COUNT` to a multiple of full stripe size divided by `DB_BLOCK_SIZE`.

If you plan to use the Database FlashSnap feature (point-in-time copy) for your Oracle database and use it on either the same host or for off-host processing or backup, the layout of volumes should meet the FlashSnap requirements.

Setting up disk group for deploying Oracle

Before creating volumes and filesystem for a database, you must set up a disk group for each database.

Review the disk group configuration guidelines before creating disk groups.

See “Disk group configuration guidelines for deploying Oracle” on page 50.

To create a disk group

- Use the `vxdg` command as follows.

  ```
  # /opt/VRTS/bin/vxdg init disk_group disk_name=disk_device
  ```

  For example, to create a disk group named *PRODdg* on a raw disk partition, where the disk name *PRODdg01* references the disk within the disk group:

  AIX

  ```
  # /opt/VRTS/bin/vxdg init PRODdg PRODdg01=Disk_0
  ```

  Linux

  ```
  # /opt/VRTS/bin/vxdg init PRODdg PRODdg01=sda
  ```

  Solaris

  ```
  # /opt/VRTS/bin/vxdg init PRODdg PRODdg01=c1t1d0s2
  ```

To add disks to a disk group

- Use the `vxdg` command as follows.

  ```
  # /opt/VRTS/bin/vxdg -g disk_group adddisk disk_name=disk_device
  ```

For example, to add a disk named PRODdg02 to the disk group PRODdg:

AIX

```
# /opt/VRTS/bin/vxdg -g PRODdg adddisk PRODdg02=Disk_0

# /opt/VRTS/bin/vxdg -g PRODdg adddisk PRODdg03=Disk_1

# /opt/VRTS/bin/vxdg -g PRODdg adddisk PRODdg04=Disk_2
```

Linux

```
# /opt/VRTS/bin/vxdg -g PRODdg adddisk PRODdg02=sda

# /opt/VRTS/bin/vxdg -g PRODdg adddisk PRODdg03=sdb

# /opt/VRTS/bin/vxdg -g PRODdg adddisk  PRODdg04=sdc
```

Solaris

```
# /opt/VRTS/bin/vxdg -g PRODdg adddisk PRODdg02=c1t2d0s2

# /opt/VRTS/bin/vxdg -g PRODdg adddisk PRODdg03=c1t3d0s2
```

```
# /opt/VRTS/bin/vxdg -g PRODdg adddisk PRODdg04=c1t4d0s2
```

Disk group configuration guidelines for deploying Oracle

Follow these guidelines when setting up disk groups.

- Only disks that are online and do not already belong to a disk group can be used to create a new disk group.
- Create one disk group for each database.
- The disk group name must be unique. Name each disk group using the Oracle database instance name specified by the environment variable ORACLE_SID and a dg suffix. The dg suffix helps identify the object as a disk group.
- Each disk name must be unique within the disk group.
- In a VVR environment, do not create database files using file systems or volumes that are not in the same disk group.

Note: You must have root privileges to execute all the disk group related VxVM commands.

See the *Storage Foundation Administrator's Guide*.

Creating volumes for deploying Oracle

Veritas Volume Manager (VxVM) uses logical volumes to organize and manage disk space. A volume is made up of portions of one or more physical disks, so it does not have the limitations of a physical disk.

Review the volume configuration guidelines before creating volumes.

See “Volume configuration guidelines for deploying Oracle” on page 51.

To create a volume

◆ Use the `vxassist` command as follows.

```
# /opt/VRTS/bin/vxassist -g disk_group make volume_name volume_size
 disk_name
```

The following is an example of creating a volume using the `vxassist` command:

To create a 1 GB volume called db01 on the PRODdg disk group:

```
#/opt/VRTS/bin/vxassist -g PRODdg make db01 1g PRODdg01
```

Volume configuration guidelines for deploying Oracle

Follow these guidelines when selecting volume layouts.

- Put the database log files on a file system created on a striped and mirrored (RAID-0+1) volume separate from the index or data tablespaces. Stripe multiple devices to create larger volumes if needed. Use mirroring to improve reliability. Do not use VxVM RAID-5 for redo logs.
- When normal system availability is acceptable, put the tablespaces on filesystems created on striped volumes for most OLTP workloads.
- Create striped volumes across at least four disks. Try to stripe across disk controllers.

  For sequential scans, do not stripe across too many disks or controllers. The single thread that processes sequential scans may not be able to keep up with the disk speed.
- For most workloads, use the default 64 K stripe-unit size for striped volumes.
- When system availability is critical, use mirroring for most write-intensive OLTP workloads. Turn on Dirty Region Logging (DRL) to allow fast volume resynchronization in the event of a system crash.
- For most decision support system (DSS) workloads, where sequential scans are common, experiment with different striping strategies and stripe-unit sizes. Put the most frequently accessed tables or tables that are accessed together on separate striped volumes to improve the bandwidth of data transfer.

Creating VxFS file system for deploying Oracle

To create a Veritas File System (VxFS) file system, use the `mkfs` or the `mkfs_vxfs` commands.

Review the file system creation guidelines before creating VxFS file systems.

See “File system creation guidelines for deploying Oracle” on page 52.

To create a VxFS file system on an existing volume

◆ Use the `mkfs` command as follows:

- AIX

```
# /usr/sbin/mkfs -V vxfs generic_options\
 -o specific_options special size
```

- Solaris

```
# /usr/sbin/mkfs -F vxfs generic_options\
 -o specific_options special size
```

- Linux

```
# /usr/sbin/mkfs -t vxfs generic_options\
 -o specific_options special size
```

Where:

- `vxfs` is the file system type
- `generic_options` are the options common to most file systems
- `specific_options` are options specific to the VxFS file system
- `special` is the full path name of the raw character device or the VxVM volume on which to create the file system
- (optional) `size` is the size of the new file system

If you do not specify size, the file system will be as large as the underlying volume.

For example, to create a VxFS file system that has an 8 KB block size and supports files larger than 2 GB on the newly created db01 volume:

- `# /usr/sbin/mkfs -V vxfs -o largefiles,bsize=8192,logsize=2000 \ /dev/vx/rdsk/PRODdg/db01`

The `-o largefiles` option allows you to create files larger than 2GB.

Note: Because size is not specified in this example, the size of the file system will be calculated automatically to be the same size as the volume on which the file system is created.

File system creation guidelines for deploying Oracle

Follow these guidelines when creating VxFS file systems.

- Specify the maximum block size and log size when creating file systems for databases.
- Do not disable the intent logging feature of the file system.
- Create separate file systems for redo logs, control files, data files, tmp files, and archive redo logs.
- When using the command line, use the mount points to name the underlying volumes. For example, if a file system named /db01 is to be created on a mirrored

volume, name the volume db01 and the mirrors db01-01 and db01-02 to relate to the configuration objects. If you are using the vxassist command or the GUI, this is transparent.

- The block size of your Oracle database should be a multiple of the file system block size. If possible, keep them of the same size.

See the *Storage Foundation Administrator's Guide*.

Mounting the file system for deploying Oracle

After creating a VxFS file system, as a root user, mount the file system using the `mount` command.

See the man pages for the `mount` and the `mount_vxfs` commands for more information.

To mount a file system

◆ Use the `mount` command as follows:

- AIX

```
# /usr/sbin/mount -V vxfs special /mount_point
```

- Solaris

```
# /usr/sbin/mount -F vxfs special /mount_point
```

- Linux

```
# /usr/sbin/mount -t vxfs special /mount_point
```

Where:

- `vxfs` is the file system type
- `special` is a block special device
- `/mount_point` is the directory where the file system will be mounted

For example, to mount a file system named `/db01` that supports large files on volume `/dev/vx/dsk/PRODdg/db01`

- `# /usr/sbin/mount -V vxfs -o largefiles /dev/vx/dsk/PRODdg/db01 \ /db01`

Installing Oracle and creating database

Use tools such as runInstaller and dbca to install Oracle and create database. While installing Oracle and creating database, use the various file systems that are created for specific purposes.

See "Planning your Oracle storage" on page 47.

Review database layouts considerations and supported configurations for deploying Oracle.

See "Considerations for database layouts" on page 182.

See "Supported Oracle configurations" on page 183.

For information on installing the Oracle software and creating Oracle databases, refer to Oracle documentation.

Chapter 4

Deploying Oracle in an off-host configuration with Storage Foundation

This chapter includes the following topics:

- Requirements for an off-host database configuration

Requirements for an off-host database configuration

If you are using Storage Foundation Database (SFDB) tools to set up an Oracle single instance database in an off-host configuration, ensure the following.

- All the tasks for deploying a single instance Oracle database in a Veritas InfoScale Storage Foundation environment are completed.
  See "Tasks for deploying Oracle databases" on page 46.
- The following requirements are met.
  - All files are on VxFS file systems over VxVM volumes. Raw devices are not supported.
  - There are no symbolic links to database files.
  - ORACLE_HOME is on a separate file system.
  - Archive logs are on a separate VxFS file system and are separate from the VxFS file system containing Oracle data files or ORACLE_HOME.
  - The database does not contain BFILES and external tables.

- Oracle datafiles, archive logs, redo logs, and control files are in a single or multiple disk groups.
- The flash recovery area must be on a separate file system from ORACLE_HOME.
- The product versions installed on the primary and secondary hosts are the same.
- The same version of Oracle is installed on both hosts, the Oracle binaries and data files are on different volumes and disks.
- The UNIX login for the database user and group must be the same on both hosts. The UNIX UID and GID must also be the same.
- You must have an Veritas InfoScale Enterprise license on both hosts.

Chapter 5

Deploying Oracle with High Availability

This chapter includes the following topics:

- Tasks for deploying Oracle in an HA configuration
- Configuring VCS to make the database highly available

Tasks for deploying Oracle in an HA configuration

If you are deploying an Oracle single instance database in a Storage Foundation High Availability (SFHA) environment, complete the following tasks.

Complete the tasks for deploying an Oracle single instance database in a Storage Foundation enviornment.	See "Tasks for deploying Oracle databases" on page 46.
Configure VCS to make the database highly available.	See "Configuring VCS to make the database highly available" on page 57.

Configuring VCS to make the database highly available

To make your Oracle database highly available, you need to bring your database configuration under Cluster Server (VCS) control.

See the *Veritas InfoScale Cluster Server Administrator's Guide*.

Chapter 6

Deploying Oracle with Volume Replicator (VVR) for disaster recovery

This chapter includes the following topics:

- About deploying Oracle with VVR for disaster recovery
- Tasks for deploying Oracle with VVR for disaster recovery
- Setting up the primary node for deploying Oracle
- Setting up the secondary node for deploying Oracle
- Configuring VVR between primary and secondary nodes for deploying Oracle

About deploying Oracle with VVR for disaster recovery

Veritas Volume Replicator (VVR) is a data-replication software designed to contribute to an effective disaster recovery plan for your Oracle environment. VVR enables you to maintain a consistent copy of the Oracle data at one or more remote locations. VVR replicates the application writes on the volumes at the source location to one or more remote locations across any distance. If a disaster occurs at the source location, you can use the copy of the Oracle data at the remote location and restart the application at the remote location.

See the *Veritas InfoScale 7.4.2 Replication Administrator's Guide*.

Tasks for deploying Oracle with VVR for disaster recovery

If you are setting up an Oracle database in a Storage Foundation environment with Veritas Volume Replicator (VVR) for disaster recovery, complete these tasks in the following order:

Set up the primary node in a VVR environment.	See "Setting up the primary node for deploying Oracle" on page 59.
Set up the secondary node in a VVR environment.	See "Setting up the secondary node for deploying Oracle" on page 60.
Configure VVR between primary and secondary nodes.	See "Configuring VVR between primary and secondary nodes for deploying Oracle" on page 62.

Setting up the primary node for deploying Oracle

The following example procedures are for the Linux platform.

To set up the primary node

1. Create a disk group on the primary node.

```
# vxdg init vvrdg sdag sdah sdai sdaj
```

2. Create a data volume on the primary node.

```
# vxassist -g vvrdg make datavol 4500M sdag
```

3. Create an archive log volume on the primary node.

```
# vxassist -g vvrdg make archvol 4500M sdah
```

4. Create and mount file systems for the data volume and the archive log volume.

```
# mkfs -t vxfs /dev/vx/rdsk/vvrdg/datavol
# mkdir /data
# mount -t vxfs /dev/vx/dsk/vvrdg/datavol /data
# mkfs -t vxfs /dev/vx/rdsk/vvrdg/archvol
# mkdir /arch
# mount -t vxfs /dev/vx/dsk/vvrdg/archvol /arch
```

5 Assign the ownership of the data volume and the archive log volume to the Oracle user.

```
# chown ora11gr2:dba /data /arch
```

6 Create SRLs (Storage Replicator Logs) for the data volume and the archive log volume on disks that do not have other volumes on them.

```
# vxassist -g vvrdg make datavol_srl 4500M sdai
# vxassist -g vvrdg make archvol_srl 4500M sdaj
```

Setting up the secondary node for deploying Oracle

To set up the secondary node

1 Create a disk group with the same name as the primary node disk group.

```
# vxdg init vvrdg sdag sdah sdai sdaj
```

2 Create directories for the data volume and the archive log volume, and assign their ownership to the Oracle user.

```
# mkdir -p /data
# mkdir -p /arch
# chown -R ora11gr2:dba /data
# chown -R ora11gr2:dba /arch
```

3 Create a data volume and an archive log volume of the same names and lengths as the primary node volumes.

```
# vxassist -g vvrdg make datavol 4500m alloc=sdj,sdk
# vxassist -g vvrdg make archvol 4500m alloc=sdl,sdm
```

4 Add a DCM (Data Change Map) log to the data volume and the archive log volume.

```
# vxassist -g vvrdg addlog datavol logtype=dcm nlog=2 logdisk=sdj,sdk
# vxassist -g vvrdg addlog archvol logtype=dcm nlog=2 logdisk=sdl,sdm
```

5 Change the ownership and permission of the directories that contain data volumes and archive log volumes.

```
# chown -R ora11gr2:dba /data*
# chown -R ora11gr2:dba /arch*
```

6 Prepare the data volume and the archive log volume for snapshot operations with DRL (Dirty Region Logging) disabled.

```
# vxsnap -g vvrdg prepare datavol drl=no alloc=sdn,sdo
# vxsnap -g vvrdg prepare archvol drl=no alloc=sdp,sdq
```

Note: To use the data on the secondary host to perform off-host processing operations, use snapshots of the secondary data volumes. Do not mount the secondary RVG volumes directly, even in read-only mode.

A snapshot is an image of the online data volumes at a specific point-in-time. The data on the original volumes may change but the data on the snapshot can still be used as a stable and independent copy for various purposes.

---

7 Verify that the data volume and the archive log volume are prepared for instant snapshots.

```
# vxprint -g vvrdg -F%instant datavol
# vxprint -g vvrdg -F%instant archvol
```

Note: VVR provides two methods of creating snapshots: instant snapshots and traditional snapshots. You would require Veritas InfoScale Enterprise and Veritas InfoScale Storage licenses for the instant snapshot feature of VxVM.

---

8 Add mirrors to the data volume and the archive log volume for use in creating third-mirror break-off snapshots.

```
# vxsnap -g vvrdg addmir datavol alloc=sdr
# vxsnap -g vvrdg addmir archvol alloc=sds
```

9 Set the `dbed_flashsnap` tag on the break-off data plexes for the data volume and the archive log volume.

```
# vxedit -g vvrdg set putil2=dbed_flashsnap datavol-04
# vxedit -g vvrdg set putil2=dbed_flashsnap archvol-04
```

10 Create SRLs (Storage Replicator Logs) for data volumes and archive log volumes on the secondary node.

```
# vxassist -g vvrdg make datavol_srl 4500M alloc=sdt
# vxassist -g vvrdg make archvol_srl 4500M alloc=sdu
```

Note: Create an SRL of the same name as the primary SRL. Please note that the SRL cannot be a volume set or a component volume of a volume set.

If the Primary RVG (Replicated Volume Group) includes a volume set, make sure that the component volumes on the secondary to be added have identical names, lengths, and indices as the component volumes on the primary.

Configuring VVR between primary and secondary nodes for deploying Oracle

To configure VVR between primary and secondary nodes

1 Obtain the primary disk group ID.

```
# vxprint -l  vvrdg
```

2 Open the `/etc/vx/vras/.rdg` file on the primary node.

3 Append the disk group ID in the `/etc/vx/vras/.rdg` file.

Note: Ensure that each disk group ID entry in the `.rdg` file is on a separate line.

---

4 Open the `/etc/vx/vras/.rdg` file on the secondary node.

5 Append the disk group ID in the `/etc/vx/vras/.rdg` file.

Note: Ensure that each disk group ID entry in the `.rdg` file is on a separate line.

---

6. From the primary node, plumb for any random IP address on a network interface.
7. From the secondary node, plumb for any random IP address on a network interface.
8. Verify that you can ping these IP addresses from both the nodes.

```
# ifconfig network_interface inet primary_IP netmask \
 subnet_mask up
# ifconfig network_interface inet secondary_IP netmask \
 subnet_mask up
```

9. Create the primary RVGs (Replicated Volume Group) of the RDS (Replicated Data Set) by using the following command for all the volumes that you want to replicate.

```
# vradmin -g vvrdg createpri vvrdg_datavol_rvg datavol datavol_srl
# vradmin -g vvrdg createpri vvrdg_archvol_rvg archvol archvol_srl
```

10. Create secondary RVGs of the same name as the primary RVGs and add them to the RDS.

```
# vradmin -g vvrdg addsec vvrdg_datavol_rvg primary_IP \
 secondary_IP
# vradmin -g vvrdg addsec vvrdg_archvol_rvg primary_IP \
 secondary_IP
```

11. Start replication to the secondary node.

```
# vradmin -g vvrdg -a startrep vvrdg_datavol_rvg secondary_IP
# vradmin -g vvrdg -a startrep vvrdg_archvol_rvg secondary_IP
```

12. Stop replication.

```
# vradmin -g vvrdg -f -s stoprep vvrdg_datavol_rvg secondary_IP
# vradmin -g vvrdg -f -s stoprep vvrdg_archvol_rvg secondary_IP
```

See the *Veritas InfoScale 7.4.2 Replication Administrator's Guide*.

Section 3

Configuring Storage Foundation for Database (SFDB) tools

- Chapter 7. Configuring and managing the Storage Foundation for Databases repository database
- Chapter 8. Configuring authentication for Storage Foundation for Databases (SFDB) tools

Chapter 7

Configuring and managing the Storage Foundation for Databases repository database

This chapter includes the following topics:

- About the Storage Foundation for Databases (SFDB) repository
- Requirements for Storage Foundation for Databases (SFDB) tools
- Storage Foundation for Databases (SFDB) tools availability
- Configuring the Storage Foundation for Databases (SFDB) tools repository
- Backing up and restoring the Storage Foundation for Databases (SFDB) repository
- Updating the Storage Foundation for Databases (SFDB) repository after adding a node
- Updating the Storage Foundation for Databases (SFDB) repository after removing a node
- Removing the Storage Foundation for Databases (SFDB) repository

About the Storage Foundation for Databases (SFDB) repository

The Storage Foundation for Databases (SFDB) repository or repository database stores metadata information required by the Storage Foundation for Databases tools.

Note: The repository database requires only occasional interaction outside of the initial installation and configuration of Veritas InfoScale Enterprise products.

In this release of Storage Foundation products, the SFDB repository is stored in a relational database and is managed by SQLite3.

Requirements for Storage Foundation for Databases (SFDB) tools

Product requirements are included in the *Veritas InfoScale 7.4.2 Release Notes*.

The hardware compatibility list contains information about supported hardware and is updated regularly. Refer to the latest information on supported hardware at:

https://www.veritas.com/content/support/en_US/doc/infoscale_hcl_73_731_74_unix

For the most current information on the single instance Oracle versions and the Oracle RAC versions that are supported with InfoScale products, refer to the database support matrix at:

https://www.veritas.com/bin/support/docRepoServlet?bookId=112632971-112632974-1&requestType=pdf

Review the current Oracle documentation to confirm the compatibility of your hardware and software.

Storage Foundation for Databases (SFDB) tools availability

SFDB tools for Oracle databases are included for the following products:

- Storage Foundation, which supports host systems with single-instance Oracle

  ---

  Note: Veritas InfoScale Enterprise licensing required.

- Veritas InfoScale Storage Foundation for Cluster File System HA, which supports clustered host systems with automatic failover and single-instance Oracle
- Storage Foundation for Oracle RAC, which supports clustered host systems with automatic failover and multiple-instance Oracle

For information on SFDB tools feature changes and issues for this release, see the *Veritas InfoScale 7.4.2 Release Notes* for the most current and complete information.

Configuring the Storage Foundation for Databases (SFDB) tools repository

The Storage Foundation for Databases (SFDB) repository stores information used by SFDB tools. The repository contains Storage Checkpoint information, tablespace and datafile information, Database FlashSnap information, and SmartTier for Oracle parameters.

The SFDB repository is automatically created when you run `dbed_update or vxsfadm` for the first time.

Note: During upgrade you need to migrate the repository to the new format. If you are upgrading from release 5.1 onwards, this step is not required.

If the repository already exists, the command will refresh it. To use `dbed_update`:

- The database must be up and running.
- The ORACLE_SID and the ORACLE_HOME variables must be specified with and -S and -H options.

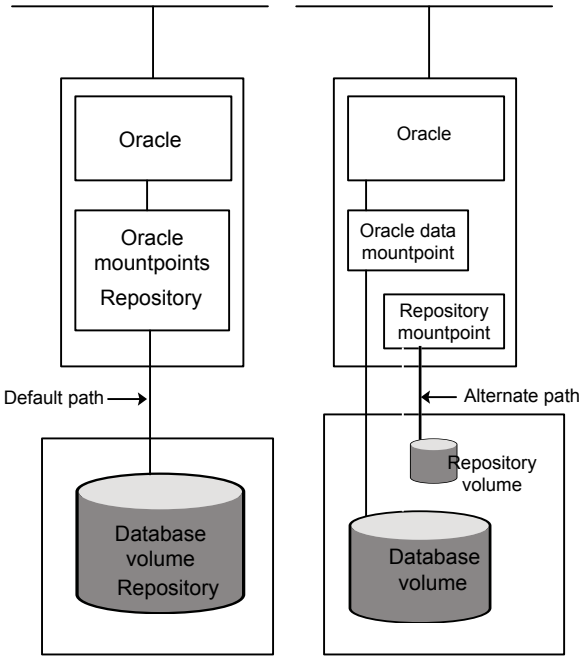
Locations for the SFDB repository

Locations for the repository:

- The default is the directory where the first data file of the SYSTEM tablespace resides.
- You can specify an alternate location using –R option of dbed_update.

In the figure below the repository directory resides in the Oracle mount points.

Figure 7-1 Locations for the SFDB repository



The alternate location must be:

- For Storage Foundation: on the local file system
- For Storage Foundation HA: on shared storage. The administrator must:
  - Configure a failover resource
  - Make sure that the repository and the Oracle database are failed over together to the same host
- For Storage Foundation Cluster File System (HA): on shared storage. The administrator must:
  - Configure Cluster File System or a failover resource
  - Make sure that the repository and the Oracle database are failed over together to the same host
- For Storage Foundation for Oracle RAC: on shared storage. CFS must be mounted on all nodes.

Note: Using the same alternate path for several databases is not supported.

Setting up the SFDB repository

The usage for `dbed_update` is as follows:

```
dbed_update -S ORACLE_SID -H ORACLE_HOME
     [ -P ORACLE_PFILE ]
     [ -R REPOSITORY_PATH ] [ -I ]
```

Options you can use:

Option	Sample value	Description
-S	ORACLE_SID	Specifies the name of the Oracle database whose information will be retrieved.
-H	ORACLE_HOME	The ORACLE_HOME setting for the ORACLE_SID database.
-P	ORACLE_PFILE	Specifies the fully qualify path of the Oracle pfile if it is not located under ORACLE_HOME/dbs directory.
-R	REPOSITORY_PATH	Specifies a user-defined location for the repository. In a RAC or HA situation, it should be located on shared storage. The default location is on the mount point of the SYSTEM tablespace of the Oracle database. This way it is shared easily among cluster nodes. If `REPOSITORY_PATH` is "/" the repository is not created in "/". It is created in the default path, since SFDB tools do not support creating repositories in "/".
-I	N/A	An integrity check of the SFDB repository is performed, without refreshing from the Oracle database . If successful, the string "ok" is printed to `stdout`. In addition, the full path of the repository file is shown. Whenever refreshing from the database, an integrity check is automatically performed.

Option	Sample value	Description
-h	N/A	Displays the help page.

To set up the SFDB repository

◆ As Oracle DBA user, run the following to create or update the SFDB repository for the database.

```
$ /opt/VRTS/bin/dbed_update -S $ORACLE_SID -H $ORACLE_HOME
```

Backing up and restoring the Storage Foundation for Databases (SFDB) repository

The SFDB repository should be backed up after significant changes such as creating a new Flashsnap configuration or creating Storage Checkpoints.

For the SFDB backup repository:

- The same default location is used as for the SFDB repository.
- The location can be specified by using the -d *directory* option.
- SFDB repositories for different databases must not share same backup location.
- The disk space at backup location is automatically managed.
- Each SFDB repository backup file has a timestamp in its file name.
- The backup command preserves only last seven backups.

For high availability environments:

- Make sure the backup location is accessible by all nodes.
- The backup command can be run from any node.

Usage for `sfua_rept_util`:

```
sfua_rept_util -S ORACLE_SID -o backup  [-d <directory>]
sfua_rept_util -S ORACLE_SID -o list  [-d <directory>]
sfua_rept_util -S ORACLE_SID -o restore  -f <backupfile> \
[-d <directory>]
```

Table 7-1 Options for `sfua_rept_util`

Option	Value	Description
-S	*ORACLE_SID*	Specifies the SID of the database for which the repository is being backed up.
-o	backup	Specifies that the operation is to make a backup.
-o	list	Specifies that the operation is to list the backup files.
-o	restore	Specifies that the operation is a restoration of the selected backup file.
-d	*directory*	Specifies the directory location for the backup or restore of the file. This option is not mandatory. If you use this option for backup, you must use this option for listing available backups and restore.
-f	*filename*	Specifies the file name for the repository backup which will be restored . This option is required for restoring a backup file.

To back up an SFDB repository

1. Verify that the database is online and that the backup location directory has write permission enabled for the DBA Oracle user.
2. Verify the SFDB repository is up to date:

```
$ dbed_update -S $ORACLE_SID -H $ORACLE_HOME
```

3. As Oracle DBA user, run the backup command with the appropriate options.

   For example:

```
$ /opt/VRTSdbed/bin/sfua_rept_util -S Prod -o backup
Backing up Repository for SID='Prod'
Target directory='/etc/vx/vxdba/logs/oragrid/backup'
Copying repository for SID=Prod to '/etc/vx/vxdba/logs/oragrid/backup'
Backup Repository for SID='Prod' Completed
```

   Where oragrid is ORACLE_USER .

To list SFDB repository backups

- As Oracle DBA user, run the list command.

```
$ /opt/VRTSdbed/bin/sfua_rept_util -S Prod -o list
Listing Repository Backups for SID='Prod'
TIME                 SIZE              NAME
2012-06-12-19-43-55 14824 rept_Prod_2012-06-12-19-43-55.tgz
```

To restore an SFDB repository

- As Oracle user, run the restore command with the appropriate options.

  For example:

```
$ /opt/VRTSdbed/bin/sfua_rept_util -S Prod -o restore
-f rept_Prod_2012-06-12-19-43-55.tgz
Restoring Repository for SID='Prod'
Target directory='/etc/vx/vxdba/logs/oragrid/backup'
Restore Repository for SID='Prod' Completed
```

  Where oragrid is ORACLE_USER.

Updating the Storage Foundation for Databases (SFDB) repository after adding a node

After adding a node to a cluster, update the SFDB repository to enable access for the new node.

To update the SFDB repository after adding a node

1. Copy the `/var/vx/vxdba/rep_loc` file from one of the nodes in the cluster to the new node.
2. If the `/var/vx/vxdba/auth/user-authorizations` file exists on the existing cluster nodes, copy it to the new node.

   If the `/var/vx/vxdba/auth/user-authorizations` file does not exist on any of the existing cluster nodes, no action is required.

   This completes the addition of the new node to the SFDB repository.

Updating the Storage Foundation for Databases (SFDB) repository after removing a node

After removing a node from a cluster, you do not need to perform any steps to update the SFDB repository.

For information on removing the SFDB repository after removing the product:

See "Removing the Storage Foundation for Databases (SFDB) repository" on page 73.

Removing the Storage Foundation for Databases (SFDB) repository

After removing the product, you can remove the SFDB repository file and any backups.

Removing the SFDB repository file disables the SFDB tools.

To remove the SFDB repository

1 Identify the SFDB repositories created on the host.

```
# cat /var/vx/vxdba/rep_loc
```

```
{
   "sfae_rept_version" : 1,
   "oracle" : {
      "SFAEDB" : {
         "location" : "/data/sfaedb/.sfae",
         "old_location" : "",
         "alias" : [
            "sfaedb"
         ]
      }
   }
}
```

2 Remove the directory identified by the `location` key.

```
# rm -rf /data/sfaedb/.sfae
```

DB2 9.5 and 9.7:

```
# rm -rf /db2data/db2inst1/NODE0000/SQL00001/.sfae
```

DB2 10.1 and 10.5:

```
# rm -rf /db2data/db2inst1/NODE0000/SQL00001/MEMBER0000/.sfae
```

3 Remove the repository location file.

```
# rm -rf /var/vx/vxdba/rep_loc
```

This completes the removal of the SFDB repository.

Chapter 8

Configuring authentication for Storage Foundation for Databases (SFDB) tools

This chapter includes the following topics:

- Configuring vxdbd for SFDB tools authentication
- Adding nodes to a cluster that is using authentication for SFDB tools
- Authorizing users to run SFDB commands

Configuring vxdbd for SFDB tools authentication

To configure vxdbd, perform the following steps as the root user

1. Run the `sfae_auth_op` command to set up the authentication services.

```
# /opt/VRTS/bin/sfae_auth_op -o setup
Setting up AT
Starting SFAE AT broker
Creating SFAE private domain
Backing up AT configuration
Creating principal for vxdbd
```

2. Stop the `vxdbd` daemon.

```
# /opt/VRTS/bin/sfae_config disable
vxdbd has been disabled and the daemon has been stopped.
```

3 Enable authentication by setting the `AUTHENTICATION` key to `yes` in the `/etc/vx/vxdbed/admin.properties` configuration file.

If `/etc/vx/vxdbed/admin.properties` does not exist, then use`cp /opt/VRTSdbed/bin/admin.properties.example /etc/vx/vxdbed/admin.properties`.

4 Start the `vxdbd` daemon.

```
# /opt/VRTS/bin/sfae_config enable
vxdbd has been enabled and the daemon has been started.
It will start automatically on reboot.
```

The `vxdbd` daemon is now configured to require authentication.

Adding nodes to a cluster that is using authentication for SFDB tools

To add a node to a cluster that is using authentication for SFDB tools, perform the following steps as the root user

1 Export authentication data from a node in the cluster that has already been authorized, by using the `-o export_broker_config` option of the `sfae_auth_op` command.

Use the `-f` option to provide a file name in which the exported data is to be stored.

```
# /opt/VRTS/bin/sfae_auth_op \
-o export_broker_config -f exported-data
```

2 Copy the exported file to the new node by using any available copy mechanism such as `scp` or `rcp`.

3 Import the authentication data on the new node by using the `-o import_broker_config` option of the `sfae_auth_op` command.

Use the `-f` option to provide the name of the file copied in Step 2.

```
# /opt/VRTS/bin/sfae_auth_op \
 -o import_broker_config -f exported-data
Setting up AT
Importing broker configuration
Starting SFAE AT broker
```

4 Stop the `vxdbd` daemon on the new node.

```
# /opt/VRTS/bin/sfae_config disable
vxdbd has been disabled and the daemon has been stopped.
```

5 Enable authentication by setting the `AUTHENTICATION` key to `yes` in the `/etc/vx/vxdbed/admin.properties` configuration file.

If `/etc/vx/vxdbed/admin.properties` does not exist, then use `cp /opt/VRTSdbed/bin/admin.properties.example /etc/vx/vxdbed/admin.properties`

6 Start the `vxdbd` daemon.

```
# /opt/VRTS/bin/sfae_config enable
vxdbd has been enabled and the daemon has been started.
It will start automatically on reboot.
```

The new node is now authenticated to interact with the cluster to run SFDB commands.

Authorizing users to run SFDB commands

To authorize users to run SFDB commands on each node of the setup or cluster, perform the following step as the root user

◆ Use the `-o auth_user` option for the `sfae_auth_op` command and provide the user name as argument to the `-u` option:

```
# /opt/VRTS/bin/sfae_auth_op \
-o auth_user -u oragrid
Creating principal oragrid@system1.example.com
```

With an off-host setup in which the off-host node is not part of the same cluster, use an additional `-h` option and perform the following steps:

1. On each primary node or nodes, which are part of a cluster, add `-h` ***off-host hostname*** to the `sfae_auth_op` command, for example:

   ```
   # /opt/VRTS/bin/sfae_auth_op -o auth_user -u oragrid -h myoffhost
   Creating principal oragrid@dblxx64-2-v6.vxindia.veritas.com
   ```

2. On the off-host node, add `-h` ***primary node*** to the `sfae_auth_op` command, for example:

   ```
   # /opt/VRTS/bin/sfae_auth_op -o auth_user -u oragrid -h dblxx64-2-v6
   Creating principal oragrid@myoffhost.vxindia.veritas.com
   ```

Section 4

Improving Oracle database performance

- Chapter 9. About database accelerators
- Chapter 10. Improving database performance with Veritas Extension for Oracle Disk Manager
- Chapter 11. Improving database performance with Veritas Cached Oracle Disk Manager
- Chapter 12. Improving database performance with Quick I/O
- Chapter 13. Improving database performance with Cached Quick I/O

Chapter 9

About database accelerators

This chapter includes the following topics:

- About Veritas InfoScale product components database accelerators

About Veritas InfoScale product components database accelerators

The major concern in any environment is maintaining respectable performance or meeting performance service level agreements (SLAs). Veritas InfoScale product components improve the overall performance of database environments in a variety of ways.

Table 9-1 Veritas InfoScale product components database accelerators

Veritas InfoScale database accelerator	Supported databases	Use cases and considerations
Oracle Disk Manager (ODM)	Oracle	■ To improve Oracle performance and manage system bandwidth through an improved Application Programming Interface (API) that contains advanced kernel support for file I/O. ■ To use Oracle Resilvering and turn off Veritas Volume Manager Dirty Region Logging (DRL) to increase performance, use ODM. ■ To reduce the time required to restore consistency, freeing more I/O bandwidth for business-critical applications, use SmartSync recovery accelerator.
Cached Oracle Disk Manager (Cached OD M)	Oracle	To enable selected I/O to use caching to improve ODM I/O performance, use Cached ODM.
Quick I/O (QIO)	Oracle DB2 Sybase	To achieve raw device performance for databases run on VxFS file systems, use Quick I/O.
Cached Quick I/O (Cached QIO)	Oracle DB2 Sybase	To further enhance database performance by leveraging large system memory to selectively buffer the frequently accessed data, use Cached QIO.
Concurrent I/O	DB2 Sybase	Concurrent I/O (CIO) is optimized for DB2 and Sybase environments To achieve improved performance for databases run on VxFS file systems without restrictions on increasing file size, use Veritas InfoScale Concurrent I/O.

These database accelerator technologies enable database performance equal to raw disk partitions, but with the manageability benefits of a file system. With the Dynamic Multi-pathing (DMP) feature of Storage Foundation, performance is maximized by load-balancing I/O activity across all available paths from server to

array. DMP supports all major hardware RAID vendors, hence there is no need for third-party multi-pathing software, reducing the total cost of ownership.

Veritas InfoScale database accelerators enable you to manage performance for your database with more precision.

For details about using ODM, Cached ODM, QIO, and Cached QIO for Oracle, see *Veritas InfoScale Storage and Availability Management for Oracle Databases*.

For details about using QIO, Cached QIO, and Concurrent I/O for DB2, see *Veritas InfoScale Storage and Availability Management for DB2 Databases*.

For details about using ODM and Cached ODM for Oracle, see *Veritas InfoScale Storage and Availability Management for Oracle Databases*.

For details about using Concurrent I/O for DB2, see *Veritas InfoScale Storage and Availability Management for DB2 Databases*.

Chapter 10

Improving database performance with Veritas Extension for Oracle Disk Manager

This chapter includes the following topics:

- About Oracle Disk Manager in the Veritas InfoScale products environment
- Setting up Veritas Extension for Oracle Disk Manager in SFHA environment
- Configuring the Veritas Extension for Oracle Disk Manager in SFHA environment
- How to prepare existing database storage for Oracle Disk Manager in SFHA environment
- Verifying that Oracle Disk Manager is configured in SFHA environment
- Disabling the Oracle Disk Manager feature in SFHA environment

About Oracle Disk Manager in the Veritas InfoScale products environment

Veritas Extension for Oracle Disk Manager enhances file management and disk I/O throughput. The features of Oracle Disk Manager are optimized for Oracle databases in a Veritas File System environment. Oracle Disk Manager enables you to improve database throughput for I/O intensive workloads with special I/O optimization.

Veritas Extension for Oracle Disk Manager supports Oracle Resilvering. With Oracle Resilvering, the storage layer receives information from the Oracle database as to which regions or blocks of a mirrored datafile to resync after a system crash. Oracle Resilvering avoids overhead from the Volume Manager Dirty Region Logging (DRL) for files which Oracle itself can recover, which increases performance.

Oracle Disk Manager reduces administrative overhead by providing enhanced support for Oracle Managed Files. Veritas Extension for Oracle Disk Manager has Quick I/O-like capabilities, but is transparent to the user. Unlike Veritas InfoScale Quick I/O, files managed using Veritas Extension for Oracle Disk Manager do not require special file naming conventions. The Oracle Disk Manager interface uses regular database files.

Oracle Disk Manager reduces administrative overhead by providing enhanced support for Oracle Managed Files. Veritas Extension for Oracle Disk Manager is transparent to the user. The Oracle Disk Manager interface uses regular database files.

Note: Quick I/O is not supported on Linux.

Database administrators can choose the datafile type used with the Oracle product. Historically, choosing between file system files and raw devices was based on manageability and performance. The exception to this is a database intended for use with Oracle Parallel Server, which requires raw devices on most platforms. If performance is not as important as administrative ease, file system files are typically the preferred file type. However, while an application may not have substantial I/O requirements when it is first implemented, I/O requirements may change. If an application becomes dependent upon I/O throughput, converting datafiles from file system to raw devices is often necessary.

Oracle Disk Manager was designed to work with Oracle databases to provide both performance and manageability. Oracle Disk Manager provides support for Oracle's file management and I/O calls for database storage on VxFS file systems and on raw volumes or partitions. This feature is provided as a dynamically-loaded shared library with which Oracle binds when it is loaded. The Oracle Disk Manager library works with an Oracle Disk Manager driver that is loaded in the kernel to perform its functions.

Note: Raw devices are not supported on Linux.

The benefits of using Oracle Disk Manager are as follows:

- True kernel asynchronous I/O for files and raw devices
- True kernel asynchronous I/O for files

- Reduced system call overhead
- Improved file system layout by preallocating contiguous files on a VxFS file system
- Performance on file system files that is equivalent to raw devices
- Transparency to users

How Oracle Disk Manager improves database performance in the SFHA environment

Oracle Disk Manager improves database I/O performance to VxFS file systems by:

Feature	Description
Support for kernel asynchronous I/O	Asynchronous I/O performs non-blocking system level reads and writes, allowing the system to perform multiple I/O requests simultaneously. Kernel asynchronous I/O is better than library asynchronous I/O because the I/O is queued to the disk device drivers in the kernel, minimizing context switches to accomplish the work.
Support for direct I/O and avoiding double buffering	I/O on files using read() and write() system calls typically results in data being copied twice: once between the user and kernel space, and the other between kernel space and the disk. In contrast, I/O on raw devices is copied directly between user space and disk, saving one level of copying. As with I/O on raw devices, Oracle Disk Manager I/O avoids the extra copying. Oracle Disk Manager bypasses the system cache and accesses the files with the same efficiency as raw devices. Avoiding double buffering reduces the memory overhead on the system. Eliminating the copies from kernel to user address space significantly reduces kernel mode processor utilization freeing more processor cycles to execute the application code.

Feature	Description
Avoids kernel write locks on database files	When database I/O is performed by way of the write() system call, each system call acquires and releases a kernel write lock on the file. This lock blocks reads during the write and prevents simultaneous write operations on the same file. Because database systems usually implement their own locks for managing concurrent access to files, write locks unnecessarily serialize I/O. Oracle Disk Manager bypasses file system locking and lets the database server control data access.
Support for many concurrent I/Os in one system call	When performing asynchronous I/O, an Oracle process may try to issue additional I/O requests while collecting completed I/Os, or it may try to wait for particular I/O requests synchronously, as it can do no other work until the I/O is completed. The Oracle process may also try to issue requests to different files. All this activity can be accomplished with one system call when Oracle uses the Oracle Disk Manager I/O interface. This interface reduces the number of system calls performed to accomplish the same work, reducing the number of user space/kernel space context switches. **Note:** You can use the `FILESYSTEMIO_OPTIONS` initialization parameter to enable or disable asynchronous I/O, direct I/O, or Concurrent I/O on file system files. This parameter is applicable to JFS and JFS2 files only. This parameter is not applicable to VxFS files, ODM files, or QIO files. For further information, refer to the Oracle Disk Manager documentation from Oracle.
Avoids duplicate opening of files per Oracle instance	Oracle Disk Manager allows files to be opened once, providing a “file identifier.” This is called “identifying” the files. The same file identifiers can be used by any other processes in the Oracle instance. The file status is maintained by the Oracle Disk Manager driver in the kernel. The reduction in file open calls reduces processing overhead at process initialization and termination, and it reduces the number of file status structures required in the kernel.

Feature	Description
Allocation of contiguous datafiles	Oracle Disk Manager can improve performance for queries, such as sort and parallel queries, that use temporary tablespaces. Without Oracle Disk Manager, Oracle does not initialize the datafiles for the temporary tablespaces. Therefore, the datafiles become sparse files and are generally fragmented. Sparse or fragmented files lead to poor query performance. When using Oracle Disk Manager, the datafiles are initialized for the temporary tablespaces and are allocated in a contiguous fashion, so that they are not sparse.

About SmartSync in SFHA environment

SmartSync recovery accelerator increases the avaiability of mirrored volumes by only resynchronizing changed data. SmartSync reduces the time required to restore consistency, freeing more I/O bandwidth for business-critical applications.

SmartSync is not supported for Oracle databases that are configured on raw volumes, because Oracle does not support the raw volume interface.

ODM has been enhanced to make use of file-level SmartSync, which is called ODM Resilvering. No configuration is needed. This is available by default.

How Oracle Disk Manager works with Oracle Managed Files

The following example illustrates the relationship between Oracle Disk Manager and Oracle Managed Files (OMF). The example shows the `init.ora` contents and the command for starting the database instance. To simplify Oracle UNDO management, the `init.ora` parameter `UNDO_MANAGEMENT` is set to `AUTO`. This is known as System-Managed Undo.

Note: Before building an OMF database, you need the appropriate `init.ora` default values. These values control the location of the `SYSTEM` tablespace, online redo logs, and control files after the `CREATE DATABASE` statement is executed.

```
$ cat initPROD.ora
UNDO_MANAGEMENT = AUTO
DB_CREATE_FILE_DEST = '/PROD'
DB_CREATE_ONLINE_LOG_DEST_1 = '/PROD'
db_block_size = 4096
db_name = PROD
$ sqlplus /nolog
```

```
SQL> connect / as sysdba
SQL> startup nomount pfile= initPROD.ora
```

The Oracle instance starts.

```
Total System Global Area 93094616 bytes
Fixed Size 279256 bytes
Variable Size 41943040 bytes
Database Buffers 50331648 bytes
Redo Buffers 540672 bytes
```

To implement a layout that places files associated with the `EMP_TABLE` tablespace in a directory separate from the `EMP_INDEX` tablespace, use the `ALTER SYSTEM` statement. This example shows how OMF handles file names and storage clauses and paths. The layout allows you to think of the tablespaces as objects in a file system as opposed to a collection of datafiles. Since OMF uses the Oracle Disk Manager file resize function, the tablespace files are initially created with the default size of 100MB and grow as needed. Use the `MAXSIZE` attribute to limit growth.

The following example shows the commands for creating an OMF database and for creating the `EMP_TABLE` and `EMP_INDEX` tablespaces in their own locale.

Note: The directory must exist for OMF to work, so the `SQL*Plus HOST` command is used to create the directories:

```
SQL> create database PROD;
```

The database is created.

```
SQL> HOST mkdir /PROD/EMP_TABLE;
SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST = '/PROD/EMP_TABLE';
```

The system is altered.

```
SQL> create tablespace EMP_TABLE DATAFILE AUTOEXTEND ON MAXSIZE \
500M;
```

A tablespace is created.

```
SQL> ALTER SYSTEM SET DB_CREATE_FILE_DEST = '/PROD/EMP_INDEX';
```

The system is altered.

```
SQL> create tablespace EMP_INDEX DATAFILE AUTOEXTEND ON MAXSIZE \
100M;
```

A tablespace is created.

Use the `ls` command to show the newly created database:

```
$ ls -lFR
total 638062
drwxr-xr-x 2 oracle11g dba 96 May  3 15:43 EMP_INDEX/
drwxr-xr-x 2 oracle11g dba 96 May  3 15:43 EMP_TABLE/
-rw-r--r-- 1 oracle11g dba 104858112 May 3 17:28 ora_1_BEhYgc0m.log
-rw-r--r-- 1 oracle11g dba 104858112 May 3 17:27 ora_2_BEhYu4NA.log
-rw-r--r-- 1 oracle11g dba 806912 May 3 15:43 ora_BEahlfUX.ctl
-rw-r--r-- 1 oracle11g dba 10489856 May 3 15:43 ora_sys_undo_BEajPSVq.dbf
-rw-r--r-- 1 oracle11g dba 104861696 May 3 15:4 ora_system_BEaiFE8v.dbf
-rw-r--r-- 1 oracle11g dba 186 May 3 15:03 PROD.ora

./EMP_INDEX:
total 204808
-rw-r--r-- 1 oracle11g dba 104861696 May 3 15:43
ora_emp_inde_BEakGfun.dbf

./EMP_TABLE:
total 204808
-rw-r--r-- 1 oracle11g dba 104861696 May 3 15:43
ora_emp_tabl_BEak1LqK.dbf
```

Setting up Veritas Extension for Oracle Disk Manager in SFHA environment

Before setting up Veritas Extension for Oracle Disk Manager, the following conditions must be met:

Prerequisites

- A Veritas InfoScale Storage Foundation Enterprise or Standard product must be installed on your system.
- A supported Oracle database version must be installed on your system.
- If Cached Quick I/O is available, do not enable Oracle Disk Manager when Cached Quick I/O is enabled for datafiles.

Usage Notes

When the Quick I/O feature is available, Oracle Disk Manager uses the Quick I/O driver to perform asynchronous I/O. Do not turn off the Quick I/O mount option, which is the default.

Configuring the Veritas Extension for Oracle Disk Manager in SFHA environment

If `ORACLE_HOME` is on a shared file system, perform the following procedure from any node. Otherwise, perform the following procedure on each node. *`ORACLE_HOME`* is the location where Oracle database binaries have been installed.

To the configure the Veritas Extension for Oracle Disk Manager

1. Log in as `oracle`.
2. If the Oracle database is running, then shut down the Oracle database.
3. After shutting down the Oracle database, disable the ODM module:

   For AIX:

   ```
   # /etc/rc.d/rc2.d/S99odm stop
   ```

   For supported Linux distributions:

   ```
   # systemctl stop vxodm
   ```

   For Solaris:

   ```
   # /usr/sbin/svcadm disable vxodm
   ```

4. Verify that the following library exists:

   For AIX:

   `/opt/VRTSodm/lib/libodm64.so`

   For Linux:

   `/opt/VRTSodm/lib64/libodm.so`

   For Solaris SPARC:

   `/opt/VRTSodm/lib/sparcv9/libodm.so`

5. Link the Oracle ODM library present in `ORACLE_HOME` with the Veritas extension for the Oracle Disk Manager library:

   For all supported Oracle versions:

   - Change to the `$ORACLE_HOME/lib` directory:

     ```
     # cd $ORACLE_HOME/lib
     ```

   - Make a backup of `libodm18.so`.

```
# mv libodm<version>.so libodm<version>.so.oracle-`date '+%m_%d_%y-%H_
```

- Link `libodm<version>.so` with Veritas ODM library:

  ```
  $ cd $ORACLE_HOME/rdbms/lib
  $ mkdir -p odm
  $ cd $ORACLE_HOME/rdbms/lib/odm
  ```

  For AIX:

  # ln -s /opt/VRTSodm/lib/libodm64.so libodm<*version*>.so

  For Linux:

  # ln -s /opt/VRTSodm/lib64/libodm.so libodm<*version*>.so

  For Solaris SPARC:

  # ln -s /usr/lib/sparcv9/libodm.so libodm<*version*>.so

6 Load the ODM module:

  For AIX:

  ```
  # /etc/rc.d/rc2.d/S99odm start
  ```

  For supported Linux distributions:

  ```
  # systemctl start vxodm
  ```

  For Solaris:

  ```
  # /usr/sbin/svcadm enable vxodm
  ```

7 Start the Oracle database.

8 To confirm that the Oracle database starts with the Veritas Extension for ODM, the alert log will contain the following text:

  ```
  Oracle instance running with ODM:
   Veritas 7.4.2.0000 ODM Library, Version 2.0
  ```

How to prepare existing database storage for Oracle Disk Manager in SFHA environment

Non-Quick I/O files in a VxFS file system work with Oracle Disk Manager without any changes. The files are found and identified for Oracle Disk Manager I/O by default. To take full advantage of Oracle Disk Manager datafiles, files should not be fragmented.

If you are using Quick I/O files in a VxFS file system and you want to move to Oracle Disk Manager, convert the Quick I/O files to normal files.

Note: Quick I/O is not supported on Linux.

Files in a VxFS file system work with Oracle Disk Manager without any changes. The files are found and identified for Oracle Disk Manager I/O by default. To take full advantage of Oracle Disk Manager datafiles, files should not be fragmented.

Verifying that Oracle Disk Manager is configured in SFHA environment

Before verifying that Oracle Disk Manager is configured, make sure that the following conditions are met:

AIX Prerequisites

- `/opt/VRTSodm/lib/libodm64.so` must exist.
- `$ORACLE_HOME/rdbms/lib/odm/libodm`*`version`*`.so` is linked to `/opt/VRTSodm/lib/libodm64.so`.
- The `VRTSodm` package must be installed.

Linux Prerequisites

- `/opt/VRTSodm/lib64/libodm.so` must exist.
- `$ORACLE_HOME/rdbms/lib/odm/libodm`*`version`*`.so` is linked to `/opt/VRTSodm/lib64/libodm.so`.
- The `VRTSodm` package must be installed.

Solaris Prerequisites

- `/opt/VRTSodm/lib/libodm.so` must exist.
- `$ORACLE_HOME/rdbms/lib/odm/libodm`*`version`*`.so` is linked to `/opt/VRTSodm/lib/sparcv9/libodm.so`.
- The `VRTSodm` package must be installed.

To verify that Oracle Disk Manager is configured

1 Verify that the ODM feature is included in the license:

```
# /opt/VRTS/bin/vxlicrep | grep ODM


QLOGODM                     = Enabled
   ODM                                  = Enabled
```

The output verifies that ODM is enabled.

Note: Verify that the license key containing the ODM feature is not expired. If the license key has expired, you will not be able to use the ODM feature.

---

2 Check that the `VRTSodm` package is installed:

Solaris:

```
# pkginfo VRTSodm
```

AIX:

```
# lslpp -L VRTSodm
```

Linux:

```
# rpm -qa | grep VRTSodm
```

or

```
# rpm -qi VRTSodm
```

3 Check that the Veritas ODM library, `libodm64.so` or `libodm.so` is present; for example:

For AIX, use `# ls -lL /opt/VRTSodm/lib/libodm64.so`.

For Linux, use `# ls -lL /opt/VRTSodm/lib/libodm64.so`.

For Solaris, use `# ls -lL /usr/lib/sparcv9/libodm.so`.

To verify that Oracle Disk Manager is running

1 Start the Oracle database.

2 Check that the instance is using the Oracle Disk Manager function:

```
# cat /dev/odm/stats
# echo $?
0
```

3 Verify that the Oracle Disk Manager is loaded:

Solaris:

```
# modinfo | grep ODM | grep VRTS
293 7b6d2000 18078  30   1   odm (VRTS ODM 7.4.1.0,REV=internal-b)
```

Linux:

```
# lsmod | grep odm
vxodm   164480  1
fdd 78976 1 vxodm
```

AIX:

You can use the `genkld` or the `genkex` commands:

```
# genkld | grep odm
or
# genkex | grep odm


        d05a3100    22182 /usr/lib/libodm.a[shr.o]
 90000001c0ec000    1675 /ora_base/db_home/lib/libodm12.so
 900000000bbf000    1059e /opt/VRTSodm/lib/libodm64.so
 90000000043d300    24167 /usr/lib/libodm.a[shr_64.o]
bash-3.2# genkex | grep odm
f1000000c037a000    27000 /usr/lib/drivers/vxodm.ext_71
```

4 In the alert log, verify the Oracle instance is running. The log should contain output similar to the following:

```
Oracle instance running with ODM:
Veritas 7.4.1.0000 ODM Library, Version 2.0
```

Disabling the Oracle Disk Manager feature in SFHA environment

Since the Oracle Disk Manager feature uses regular files, you can access these files as regular VxFS files as soon as the feature is disabled.

The steps for disabling the Oracle Disk Manager feature are the same for both 32- and 64-bit versions of Oracle.

Note: Quick I/O is not supported on Linux.

Note: To convert to VxFS with Quick I/O, disable Oracle Disk Manager using the following procedure, then convert the files to Quick I/O files.

See "Converting Oracle files to Quick I/O files" on page 124.

Before disabling the Oracle Disk Manager feature, you may want to back up your files.

To disable the Oracle Disk Manager feature in an Oracle instance

1. Shut down the database instance.
2. On AIX, use the `rm` and `ln` commands to remove the link to the Oracle Disk Manager Library.

```
$ rm $ORACLE_HOME/rdbms/lib/odm/libodmversion.so
$ ln -s ${ORACLE_HOME}/lib/libodmdversion.so \
${ORACLE_HOME}/lib/libodmversion.so
```

3. On Linux, use the `rm` and `ln` commands to remove the link to the Oracle Disk Manager Library.

```
$$ORACLE_HOME/rdbms/lib/odm/libodmversion.so
$ ln -s ${ORACLE_HOME}/lib/libodmdversion.so \
${ORACLE_HOME}/lib/libodmversion.so
```

4. On Solaris, use the `rm` and `ln` commands to remove the link to the Oracle Disk Manager Library.

```
$ rm $ORACLE_HOME/rdbms/lib/odm/libodmversion.so
$ ln -s ${ORACLE_HOME}/lib/libodmdversion.so \
${ORACLE_HOME}/lib/libodmv.so
```

5. Restart the database instance.

Chapter 11

Improving database performance with Veritas Cached Oracle Disk Manager

This chapter includes the following topics:

- About Cached ODM in SFHA environment
- Configuring Cached ODM in SFHA environment
- Administering Cached ODM settings with Cached ODM Advisor in SFHA environment

About Cached ODM in SFHA environment

ODM I/O normally bypasses the file system cache and directly reads from and writes to disk. Cached ODM enables some I/O to use caching and read ahead, which can improve ODM I/O performance for certain workloads. Cached ODM can perform a conditional form of caching that is based on per-I/O hints from Oracle. The hints indicate what Oracle will do with the data. ODM uses these hints to perform caching and read ahead for some reads, but ODM avoids caching other reads, even for the same file.

Considerations for using Cached ODM in SFHA environment

Cached ODM is most useful for read-intensive workloads. For write-intensive workloads or low memory environments, Cached ODM is not advised.

Cached ODM advantages over ODM:

- ODM does direct I/O
  - Oracle can use larger System Global Area (SGA).
  - Read-aheads are not supported.
- Some read-intensive Oracle workloads can perform poorly when ODM is used.
- Oracle SGA is not good enough for some cases where a host may have more than one database.
  - Host may have more than one database. (Pagecache can serve multiple databases.)
  - Parallel Query processes many times do not use SGA

See "Enabling Cached ODM for file systems in SFHA environment" on page 98.

Cached ODM supported configurations in SFHA environment

Cached ODM is supported for:

- Storage Foundation (HA)
- Storage Foundation Cluster File System (HA)
- Local and cluster mounts

Storage Foundation for Oracle RAC is not currently supported.

Cached ODM does not affect the performance of files and file systems for which you did not enable caching.

About Cached ODM Advisor in SFHA environment

The Cached ODM Advisor is a tool for database administrators which provides the following:

- Cached ODM Advisor enables you to manage and configure cached ODM on database files without requiring root privileges.
- The settings applied with the Cached ODM Advisor are automatically persistent and common across all the cluster nodes.

Note: Cached ODM Advisor does not provide an interface to the ODM cachemap.

Configuring Cached ODM in SFHA environment

Configuring Cached ODM requires first enabling Cached ODM for a file system. After enabling Cached ODM, it can be configured in two ways:

- The primary configuration method: turn caching on or off for all I/O on a per-file basis.
- The secondary configuration method: adjust the ODM cachemap. The cachemap maps file type and I/O type combinations into caching advisories.

See "Tuning Cached ODM settings for individual files in SFHA environment" on page 98.

See "Tuning Cached ODM settings via the cachemap in SFHA environment" on page 99.

Enabling Cached ODM for file systems in SFHA environment

Cached ODM is initially disabled on a file system. You can enable Cached ODM for a file system by setting the `odm_cache_enable` option of the `vxtunefs` command after the file system is mounted.

See the `vxtunefs`(1M) manual page.

Note: The `vxtunefs` command enables conditional caching for all of the ODM files on the file system.

To enable Cached ODM for a file system

1. Enable Cached ODM on the VxFS file system `/database01`:

```
# vxtunefs -o odm_cache_enable=1 /database01
```

2. Optionally, you can make this setting persistent across mounts by adding a file system entry in the file `/etc/vx/tunefstab`:

```
/dev/vx/dsk/datadg/database01 odm_cache_enable=1
```

   See the `tunefstab`(4) manual page.

3. For Cluster File System, you must modify /etc/vx/tunefstab on all nodes.

Tuning Cached ODM settings for individual files in SFHA environment

You can use the `odmadm setcachefile` command to override the cachemap for a specific file so that ODM caches either all or none of the I/O to the file. The caching

state can be ON, OFF, or DEF (default). The DEF caching state is conditional caching, meaning that for each I/O, ODM consults the cachemap and determines whether the specified file type and I/O type combination should be cached. The ON caching state causes the specified file always to be cached, while the OFF caching state causes the specified file never to be cached.

See the `odmadm`(1M) manual page.

Note: The cache advisories operate only if Cached ODM is enabled for the file system. If the `odm_cache_enable` flag is zero, Cached ODM is OFF for all of the files in that file system, even if the individual file cache advisory for a file is ON.

To enable unconditional caching on a file

- Enable unconditional caching on the file `/mnt1/file1`:

```
# /opt/VRTS/bin/odmadm setcachefile /mnt1/file1=on
```

  With this command, ODM caches all reads from `file1`.

To disable caching on a file

- Disable caching on the file `/mnt1/file1`:

```
# /opt/VRTS/bin/odmadm setcachefile /mnt1/file1=off
```

  With this command, ODM does not cache reads from `file1`.

To check on the current cache advisory settings for a file

- Check the current cache advisory settings of the files `/mnt1/file1` and `/mnt2/file2`:

```
# /opt/VRTS/bin/odmadm getcachefile /mnt1/file1 /mnt2/file2
/mnt1/file1,ON
/mnt2/file2,OFF
```

To reset all files to the default cache advisory

- Reset all files to the default cache advisory:

```
# /opt/VRTS/bin/odmadm resetcachefiles mountpoint
```

Tuning Cached ODM settings via the cachemap in SFHA environment

You can use the `odmadm setcachemap` command to configure the cachemap. The cachemap maps file type and I/O type combinations to caching advisories. ODM

uses the cachemap for all files that have the default conditional cache setting. Such files are those for which caching has not been turned on or off by the `odmadm setcachefile` command.

See the `odmadm`(1M) manual page.

By default, the cachemap is empty, but you can add caching advisories by using the `odmadm setcachemap` command.

To add caching advisories to the cachemap

1. Add a caching advisory to the cachemap:

   ```
   # /opt/VRTS/bin/odmadm setcachemap \
    data/data_read_seq=cache,readahead
   ```

   With this example command, ODM uses caching and readahead for I/O to online log files (`data`) that have the `data_read_seq` I/O type. You can view the valid file type and I/O type values from the output of the `odmadm getcachemap` command.

   See the `odmadm`(1M) manual page.

2. The cachemap is local on each node. To make the same caching decisions on each node in a cluster, keep the cachemap settings consistent by running the same `/opt/VRTS/bin/odmadm setcachemap` commands on each node in a cluster.

Making the caching settings persistent across mounts in SFHA environment

By default, the Cached ODM settings are not persistent across mounts. You can make the settings persistent by addng them to odmadm configuration files.

To make the caching setting persistent across mounts

1. Cachemap settings can be added to /etc/vx/odmadm:

```
setcachemap data/read_data_header=cache
setcachemap all/datapump=cache,readahead
```

2. The cachemap is local on each node. To keep the cachemap consistent in a cluster, the contents of /etc/vx/odmadm must be the same on each node.
3. Per-file settings can be manually added to the lost+found/odmadm file in the file system.

   For example, to disable caching for oradata/file1 each time the filesystem is mounted, enter the following in the lost+found/odmadm file in the file system.

```
setcachefile oradata/file1=off
```

   The per-file settings in lost+found/odmadm file may also be managed using Cached ODM Manager. The file should not be edited manually while using the Cached ODM Manager.

   See "Enabling and disabling Cached ODM on data files by using Cached ODM Advisor in SFHA environment" on page 109.

Administering Cached ODM settings with Cached ODM Advisor in SFHA environment

The Cached ODM Advisor simplifies the task of managing the cached ODM settings for database administrators:

- Cached ODM Advisor enables you to manage and configure cached ODM on database files without requiring root privileges.
- The settings applied with the Cached ODM Advisor are automatically persistent and common across all the cluster nodes.
- While the Cached ODM Advisor does not provide an interface to the ODM cachemap, it enables setting the cached ODM setting to ON/ OFF (and not to DEF).
- ODM Advisor lets you generate a variety of reports that help you determine which datafiles are suitable for enabling Cached ODM.

The Cached ODM Advisor command `dbed_codm_adm` should be run by the DBA user.

Note: If you are using Cached ODM Advisor for clone databases, a clone database will not have the cached ODM settings enabled when it is enabled for the primary database. You must manually enable cached ODM by setting `odm_cache_enable = 1` for the clone database.

The Cached ODM Advisor command `dbed_codm_adm` syntax is illustrated below and its options are listed in Table 11-1.

```
Cached ODM Administration
dbed_codm_adm
Usage:
-S ORACLE_SID -H ORACLE_HOME -o display [ -n num -c col -t tbs ]
-S ORACLE_SID -H ORACLE_HOME -o display [ -n num ] [ -t tbs ] [-p] [-a]
                            [ -d start_date end_date
                              -m start_time end_time ]
                            [ -b blacklist_file ]
                            [ datafile ]
-S ORACLE_SID -H ORACLE_HOME -o display [ -n num ] [ -t tbs ] [-p] [-a]
                            [ -u snapshot_range [ -u snapshot_range ] ]
                            [ -b blacklist_file ]
                            [ datafile ]
-S ORACLE_SID -H ORACLE_HOME -o display [ -n num ] [ -u num_snapshots]
                            [ -t tbs ]
                            [ -b blacklist_file ]
                            [ datafile ]
-S ORACLE_SID -H ORACLE_HOME -o [ on | off | odmstats ] datafile | -f filelist
-S ORACLE_SID -H ORACLE_HOME -o add-to-blacklist { datafile | -f filelist }
-S ORACLE_SID -H ORACLE_HOME -o remove-from-blacklist { datafile | -f filelist }
-S ORACLE_SID -H ORACLE_HOME -o print-blacklist
-S ORACLE_SID -H ORACLE_HOME -o filestate [ datafile | -f filelist ]
-o iostats
```

Table 11-1 Cached ODM Advisor command options

Option	Use
-S ORACLE_SID	Specify the ORACLE_SID. ORACLE_SID is the SID of the Oracle database to be used.
-H ORACLE_HOME	Specify the ORACLE_HOME. ORACLE_HOME is the HOME of the Oracle instance to be used.
-o display	Generate a report of read activity on datafiles.

Table 11-1 Cached ODM Advisor command options *(continued)*

Option	Use
-o filestate	Diaplay whether datafiles have Cached ODM turned on or not.
-o on	Enable Cached ODM.
-o off	Disable Cached ODM.
-o odmstats	Displays Cached ODM I/O statistics for datafiles.
-o iostats	Displays cumulative I/O statistics for file-type and I/O-type combinations.
-c column_index	Specify the column on which the basic report is sorted. This is an optional field. By default I/O statistics would be sorted on number of physical reads (PHYRDS).
-n num	Change the default number of files included in the report. Use this option together with `-o display` to show the top num files which are candidates for enabling Cached ODM.
-f filelist	Specify a filename that has the list of datafiles to operate on with one file name on each line. If you do not provide a list, all of the datafiles will be processed, except in the case of the `-o on` or `-o off` option.
-t tbs	Specify the tablespace name for which the report is to be generated.
filename	Specify a single file name to be processed.
-d start_date end_date	Specify a date range for which the report is to be generated. The dates are inclusive and they must be specified in yyyy/mm/dd format.
-m start_time end_time	Specify a time range for which the report is to be generated. The start time and end time must be specified as the hour of day in 24-hour format.
-u snapshot_specification	Specify the range of AWR (Automatic Workload Repository) snapshots to consider for the report.
-b blacklist_file	Specify an alternate file to use as a blacklist.
-o add-to-blacklist	Add a file to the default blacklist.
-o remove-from-blacklist	Remove a file from the default blacklist.

Table 11-1 Cached ODM Advisor command options *(continued)*

Option	Use
-o print-blacklist	Display the blacklisted files.
-a	Include files that have Cached ODM enabled in generated reports.
-p	Suppress tabular formatting to generate output suitable for programmatic processing.

Generating reports of candidate datafiles by using Cached ODM Advisor in SFHA environment

You can use the `dbed_codm_adm` command to generate a number of reports that help you determine which datafiles are suitable for enabling Cached ODM. These reports are generated by using the `-o display` option of the `dbed_codm_adm` command. Most of these reports require that the database have periodic AWR (Automatic Workload Repository) snapshots enabled.

Generating a basic report of the current read activity by using Cached ODM Advisor in SFHA environment

This basic report summarizes the read activity that has occurred since the instance started. The report summarizes the information available in the V$FILESTAT view provided by Oracle. By default, it shows the 10 files that have had the maximum number of physical reads (PHYRDS). You can change the number of files shown by using the `-n` option and the sort column by using the `-c` option. You can restrict the report to a specific tablespace by using the `-t`option.

Note: The `dbed_codm_adm` command generates a warning if the system free memory is less than 25% of the total system memory. Enabling Cached ODM in the absence of sufficient free memory can have an adverse impact on performance.

This report is the only kind that can be generated if the database does not have periodic AWR snapshots enabled.

The following is an example of generating a report of datafiles with the maximum read activity since the database instance started.

```
$ dbed_codm_adm -S sfaedb -H $ORACLE_HOME -o display

File I/O statistics from Oracle V$FILESTAT view sorted on PHYRDS in descending order:
FILENAME                PHYRDS PHYWRTS PHYBLKRD PHYBLKWRT READTIM WRITETIM
```

```
--------------------------------------------------------------------------

/data/sfaedb/data02.dbf 2225795 1453298 2291527 2087071 5371446 521421
/data/sfaedb/data03.dbf 2225697 1368342 2291338 2135174 9070509 552163
/data/sfaedb/data04.dbf 2156475 1327998 2221101 2054575 9036635 584703
/data/sfaedb/data05.dbf 2127539 1343213 2187749 2060534 8854222 626665
/data/sfaedb/data06.dbf 2076911 1306907 2135606 1972178 8600763 659874
/data/sfaedb/data07.dbf 2025810 1265414 2084207 1910216 8434117 667847
/data/sfaedb/data08.dbf 1937825 1171931 1996027 1779837 7991536 673212
/data/sfaedb/data09.dbf 1875733 1154502 1934094 1786911 7777172 692082
/data/sfaedb/data10.dbf 1819224 1114452 1877443 1721618 7521514 690706
/data/sfaedb/data01.dbf 1750829 1086044 1805138 1691258 7267005 357762
```

Generating summary reports of historical activity by using Cached ODM Advisor in SFHA environment

The Cached ODM Advisor can generate reports that summarize historical read activity on the datafiles. These reports use data from the AWR snapshots periodically taken by the database. You can use several options to fine-tune the report.

Reports generated using either of the following options have the following columns:

- Tablespace name
- File name
- Average physical reads per snapshot
- Average physical blocks read per snapshot
- Read time (in seconds)

The reports can be restricted to a specific tablespace by using the `-t`option. Datafiles which have Cached ODM enabled are not displayed in the report. If you would like to include those datafiles in the report as well, use the `-a` option. By default, the top 10 datafiles are displayed in the report. This default can be changed by using the `-n` option. The `-p` option generates a programmatically parseable report by disabling of the table formatting performed by default.

Using the Cached ODM Advisor blacklist to ignore files in SFHA environment

If tuning and measurement indicates that enabling Cached ODM on certain datafiles does not improve performance, you can add the files to a blacklist. Files added to the blacklist are not included in any of the reports generated by using the `-o display` option of `dbed_codm_adm`.

The blacklist file is stored in the file system at `lost+found/odmadm-blacklist`. You can also provide an alternate blacklist location using the `-b` option.

You can use the `-o add-to-blacklist` and `-o remove-from-blacklist` options to manage the default blacklist. These options can be used with either a single file name or with a list of files using the `-f filelist` option. The `-o print-blacklist` option displays the list of blacklisted files. The blacklist is consulted only when generating a report. It has no effect on enabling or disabling Cached ODM on datafiles.

Average metric values during a time interval by using Cached ODM Advisor in SFHA environment

You can use the `-d start_date end_date` and `-m start_hour end_hour` options to restrict the report to activities that were recorded during the specified time interval. The `start_date` and `end_date` parameters are inclusive. Therefore, activities occuring on these dates are also included in the generated report. The `start_hour` and `end_hour` parameters specify the time interval bounding the AWR snapshots on each day. This report is useful when you start using the Cached ODM Advisor utility for the first time and need a list of datafiles that see the most read activity during workload hours.

The following is an example of generating a report of read activity on the DATA tablespace between 9 am and 5 pm, from 2011/07/09 to 2011/07/11.

```
$ dbed_codm_adm -S sfaedb -H $ORACLE_HOME -o display \
 -d 2011/07/09 2011/07/11 -m 9 17 -t DATA
Tablespace  Filename                     Average   Average   Read
                                         Physical  Physical  Time (s)
                                         Reads per Blocks
                                         Snapshot  Read per
                                                   Snapshot
DATA        /data/sfaedb/data03.dbf      13848     13848     497.16
DATA        /data/sfaedb/data04.dbf      13103     13104     472.68
DATA        /data/sfaedb/data05.dbf      12656     12656     451.79
DATA        /data/sfaedb/data06.dbf      12154     12154     430.81
DATA        /data/sfaedb/data07.dbf      11937     11937     423.87
DATA        /data/sfaedb/data08.dbf      11853     11853     417.74
DATA        /data/sfaedb/data09.dbf      11720     11720     415.45
DATA        /data/sfaedb/data10.dbf      11298     11299     400.59
DATA        /data/sfaedb/data01.dbf      10610     10611     390.25
Average per-block read time: 33.539 ms
```

Average metric values during an interval specified as a range of AWR snapshots by using Cached ODM Advisor in SFHA environment

You can use the `-u snapshot_specification` option of the `dbed codm_adm` command to restrict the report to activities that were recorded in a specific range of AWR snapshots. The `snapshot_specification` can be a single integer, in which case the report considers that many snapshots starting from the latest, in a reverse order. This allows you to generate a report on the recent read activity. Alternatively, the `snapshot_specification` can be used to specify a range of AWR snapshot ids, in the format `starting_snapshot_id-ending_snapshot_id`. This makes the report consider the snapshots in the specified range of snapshot ids.

The following is an example of generating a report of read activity during the interval specified by the last 10 AWR snapshots.

```
$ dbed_codm_adm -S sfaedb -H $ORACLE_HOME -o display -u 10 -t DATA
Tablespace  Filename                    Average   Average   Read
                                        Physical  Physical  Time (s)
                                        Reads per Blocks
                                        Snapshot  Read per
                                                  Snapshot
DATA        /data/sfaedb/data06.dbf     31188     31188     1547.12
DATA        /data/sfaedb/data05.dbf     31131     31131     1542.57
DATA        /data/sfaedb/data03.dbf     30757     30757     1500.78
DATA        /data/sfaedb/data04.dbf     30687     30687     1525.45
DATA        /data/sfaedb/data07.dbf     30569     30569     1522.47
DATA        /data/sfaedb/data08.dbf     28781     28781     1422.77
DATA        /data/sfaedb/data09.dbf     27789     27789     1372.41
DATA        /data/sfaedb/data10.dbf     26828     26828     1317.54
DATA        /data/sfaedb/data01.dbf     24978     24978     1223.66
Average per-block read time: 46.449 ms
```

The following is an example of generating a report of read activity during the interval specified by AWR snapshot id 80 to 90.

```
$ dbed_codm_adm -S sfaedb -H $ORACLE_HOME -o display -u 80-90 -t DATA
Tablespace  Filename                    Average   Average   Read
                                        Physical  Physical  Time (s)
                                        Reads per Blocks
                                        Snapshot  Read per
                                                  Snapshot
DATA        /data/sfaedb/data03.dbf     38896     38896     1544.60
DATA        /data/sfaedb/data04.dbf     37649     37649     1586.97
DATA        /data/sfaedb/data05.dbf     37139     37139     1532.11
DATA        /data/sfaedb/data06.dbf     35875     35875     1462.15
```

```
DATA        /data/sfaedb/data07.dbf     34451     34451     1404.10
DATA        /data/sfaedb/data08.dbf     32219     32219     1284.36
DATA        /data/sfaedb/data10.dbf     31451     31451     1247.67
DATA        /data/sfaedb/data01.dbf     31377     31377     1254.35
DATA        /data/sfaedb/data09.dbf     31320     31320     1243.18
Average per-block read time: 37.691 ms
```

Generating historical activity reports on individual datafiles by using Cached ODM Advisor in SFHA environment

When considering datafiles as candidates for enabling Cached ODM, it is useful to be able to get more detailed reports for individual datafiles. You can generate such a report by specifying a datafile with the `-o display` option of the `dbed_codm_adm` command.

This report has the following columns:

- Snapshot ID
- Interval end time
- Physical reads
- Physical block reads
- Read time (in seconds)

You can use the date and time range options (`-d` and `-m`) and the snapshot range option (`-u`) of the `dbed_codm_adm` command to control the activity considered for these reports. In addition, when generating report on an individual datafile, you can generate a comparative report by using multiple snapshot id ranges. You can provide the `-u starting_snapshot_id-ending_snapshot_id` option a second time with a different range of snapshot ids and then generate a report comparing the read performance on the datafile during the two intervals. You can use this report to compare the read performance before and after enabling Cached ODM.

The following is an example of generating a report comparing the read performance on a datafile before and after enabling Cached ODM on that datafile.

```
$ dbed_codm_adm -S sfaedb -H $ORACLE_HOME -o display \
 -u 58-69 -u 70-80 /data/sfaedb/data02.dbf
Filename: /data/sfaedb/data02.dbf
Current ODM caching status: ON
Currently blacklisted: no

from AWR snapshot id 58 to 69
Snapshot  Interval End Time     Physical    Physical   Read Time
Id                              Reads       Blocks     (s)
```

```
                                          Read
58        2011/07/11 01:30:22   33026       33026      1188.13
59        2011/07/11 02:30:35   39158       39158      1358.33
60        2011/07/11 03:30:43   49186       49186      1694.08
61        2011/07/11 04:30:53   46941       46941      1652.38
62        2011/07/11 05:30:00   44090       44090      1519.78
63        2011/07/11 06:30:09   43095       43095      1511.17
64        2011/07/11 07:30:19   41272       41272      1430.41
65        2011/07/11 08:30:26   39869       39869      1379.52
66        2011/07/11 09:30:33   38540       38540      1362.99
67        2011/07/11 10:30:45   37157       37157      1309.31
68        2011/07/11 11:30:01   36034       36034      1250.8
69        2011/07/11 12:30:10   46344       46344      934.87
Average per-block read time: 33.538 ms

from AWR snapshot id 70 to 80
Snapshot  Interval End Time     Physical    Physical   Read Time
Id                              Reads       Blocks     (s)
                                            Read
70        2011/07/11 13:30:16   48041       48041      951.42
71        2011/07/11 14:30:24   46978       46978      809.31
72        2011/07/11 15:30:36   47077       47077      127.41
73        2011/07/11 16:30:47   44140       44140      370.01
74        2011/07/11 17:30:56   41467       41467      993.7
75        2011/07/11 18:30:10   39998       39998      867.62
76        2011/07/11 19:30:18   40812       40812      517.06
77        2011/07/11 20:30:29   38918       38918      706.19
78        2011/07/11 21:30:38   37572       37572      856.43
79        2011/07/11 22:30:53   37694       61675      594.25
80        2011/07/11 23:30:04   38048       38048      151.29
Average per-block read time: 14.327 ms
```

Enabling and disabling Cached ODM on data files by using Cached ODM Advisor in SFHA environment

Using the Cached ODM Advisor to manage the Cached ODM settings of datafiles provides a greater degree of control and ease of use than manually adding each file to the odmadm configuration file. Cached ODM Advisor makes the Cached ODM settings persistent by making an entry for each datafile that gets configured with the tool in the *mountpoint*/lost+found/odmadm configuration file, creating it the first time if necessary.

If you are using Cached ODM Advisor to manage the Cached ODM settings of datafiles, the odmadm configuration file should not be edited manually. The settings applied with Cached ODM Advisor for a data file take precedence over any configuration for the same file specified in the /etc/vx/odmadm configuration file.

You can use Cached ODM Advisor to enable and disable Cached ODM settings on database datafiles.

```
dbed_codm_adm -S ORACLE_SID -H ORACLE_HOME -o [ on | off ]  datafile | \
-f list_file
```

For enabling or disabling Cached ODM on database data files:

- Values for `datafile` and `list_file` should contain absolute pathnames.
- The `dbed_codm_adm` command only turns on Cached ODM on a file level.

Prerequisites	■ Specified files must be database datafile, or an error is returned. ■ Specified files must be on a VXFS filesystem, or an error is returned.
Notes	■ The `dbed_codm_adm` does not enable Cached ODM based on I/O hint, it only displays I/O hint information. ■ You must supply either a datafile name or the `-f list_file` option.

To enable Cached ODM

◆ To enable Cached ODM on the datafile */oradata/tb1.dbf*, run:

```
$ dbed_codm_adm  -S prod -H /orahome -o on /oradata/tb1.dbf
```

To disable Cached ODM on a list of datafiles

◆ To disable Cached ODM on a list of datafiles where */tmp/list_file* contains a list of datafiles name, run

```
$ dbed_codm_adm -S prod -H /orahome -o off -f /tmp/list_file
```

If user wants to turn on Cache ODM on a file and the per file system flag odm_cache_enable is not set, dbed_codm_adm will automatically turn it on for the filesystem. This flag needs to be turned on first so that file level caching can be turned on. Since this requires root access, it will be done through the DBED daemon.

Flow of control for enabling/disabling caching examples:

If you want to enable Cached ODM for file abc:

- The Cached ODM Advisor will enable filesystem odm_cache_enable flag if it has not been enabled yet.
- The Cached ODM Advisor will execute the `odmadm setcachefile abc=on` command and also add or update the entry into the configuration file.

If you want to disable caching for abc:

- The Cached ODM Advisor will only execute the command `odmadm setcachefile abc=off` and also update the entry in the configuration file.
- Cached ODM Advisor will not disable filesystem `odm_cache_enable` flag, since there may be other files that is still cached ODM enabled. The SFDB tools do not provide the command to disable filesystem `odm_cache_enable` flag. If the filesystem `odm_cache_enable`flag is disabled by other means, the per file caching will still be there just that caching will not be in effect.

Display the Cached ODM states of the database files by using Cached ODM Advisor in SFHA environment

Following is the command option to display all the database files and their states. The datafile `| -f list_file` parameter is optional. If no datafile or `-f list_file` parameter is supplied, all the files in the database will be displayed.

```
dbed_codm_adm -S ORACLE_SID -H ORACLE_HOME -o filestate \
[ datafile | -f list_file ]
```

Cached ODM is configured at two levels:

- File System level
  For the file system level, there are two states: ON or OFF.
- File level
  For the file level, there are three states: ON, OFF or DEF (default).

Table 11-2 illustrates how to determine the most effective Cached ODM setting.

Table 11-2

		File Level	
File system level	DEF	ON	OFF
ON	DEF	ON	OFF
OFF	OFF	OFF	OFF

The `dbed_codm_adm` command displays the effective state for the Cached ODM configuration. In the case of non-existent or non-File System files, a special state called error state (ERR) is displayed.

Output is sorted from files with Cached ODM set to ON first then the files with cached ODM set to OFF for ease in finding all the files with Cached ODM ON. If there are thousands of files to display,

To display CODM states for files

◆ To display CODM states for files listed in */tmp/list* run:

```
$ dbed_codm_adm -S prod -H /orahome -o filestate -f /tmp/list

Effective Cached ODM Status For Datafiles

(State value 'ERR' implies non-vxfs or non-existent file.)
FILENAME                                                STATE
-------------------------------------------------------
/data/CODM/system01.dbf                                  DEF
/data/CODM/undotbs01.dbf                                 DEF
/data/CODM/sysaux01.dbf                                  DFF
/data/CODM/users01.dbf                                   OFF
/data/CODM/example01.dbf                                 ON
```

Show Cached ODM statistics by using Cached ODM Advisor in SFHA environment

After Cached ODM is enabled on the database files, one can show the effectiveness of turning on cached ODM by displaying the ODM statistics – includes information such as number of read operations, write operations, number of block read, block write, cache hit ratio, etc. User can get some hint on how effective is turning on the Cached ODM by reviewing the hit ratio.

Prerequisites

- Specified files must be database datafile, or an error is returned.
- Specified files must be on a VXFS filesystem, or an error is returned.

Notes

- If a datafile or a list of datafiles is not supplied, it will display odmstats of all the files that are being cached ODM enabled for the specified ORACLE_SID.
- Statistics are reset to zero when the file is no longer opened by anyone.
- ODM statistics are only collected for filesystems where odm_cache_enable has been set. If odm_cache_enable is not set for the filesystem which contain the datafile, information such as CACHE STATISTICS, HIT RATIO will be set to 0.

Following is the command option to display the ODM statistics of one database file. It is mandatory to supply either a datafile name or use the –f list_file option.

```
dbed_codm_adm -S ORACLE_SID -H ORACLE_HOME -o odmstats [  datafile | -f list_file ]
```

For example, to show odm statistics for the datafile /oradata/tb1.dbf , use

```
$ dbed_codm_adm -S prod -H /orahome -o odmstats /oradata/tb1.dbf


FILENAME         NREADS NWRITES RBLOCKS WBLOCKS RTIME WTIME NREQUESTIO NDISKIO HIT-RATIO
-------------------------------------------------------------------------------------------
/oradata/tb1.dbf 1262    10617   51040   287056  11.0  1.0        4728    5918     22.0
```

Displaying ODM I/O statistics by using Cached ODM Advisor in SFHA environment

You can display the ODM I/O statistics. I/O statistics include information such as I/O requests, amount of data processed, for each file type and I/O type combination.

To show ODM I/O statistics for the local host, use:

```
$ dbed_codm_adm -o iostats
```

For displaying ODM I/Statistics:

- `-S ORACLE_SID`, `-H ORACLE_HOME`, datafile, `-f list_file` options are not required as the I/O statistics that is shown is specific to the local node, and it spans all filesystems.
- The statistics shown is not aggregated statistics from all nodes if used on an Oracle RAC cluster.
- Use VxFS command `odmadm getiostats` internally and filter the data.

- The I/O statistics provide only counts of operations and do not indicate whether the operations were cached or not.
- ODM I/O statistics are only collected for filesystems where `odm_cache_enable` has been set.

To display ODM I/O statistics, use

```
$ dbed_codm_adm -o iostats
```

```
FILE-TYPE/IO-TYPE              OPERATIONS        BYTES
----------------------------------------------------------------
data/data_cache_load           33                270336
data/read_data_header          104               851968
data/data_read_seq             2                 16384
olg/redolog_read               98                98283008
```

Chapter 12

Improving database performance with Quick I/O

This chapter includes the following topics:

- About Quick I/O
- Creating Oracle database files as Quick I/O files using qiomkfile
- Preallocating space for Quick I/O files using the setext command
- Accessing regular VxFS files as Quick I/O files
- Converting Oracle files to Quick I/O files
- About sparse files
- Handling Oracle temporary tablespaces and Quick I/O
- Displaying Quick I/O status and file attributes
- Extending a Quick I/O file
- Using Oracle's AUTOEXTEND with Quick I/O files
- Recreating Quick I/O files after restoring a database
- Disabling Quick I/O
- Creating Quick I/O files in Solaris local zone

About Quick I/O

Veritas Quick I/O is a VxFS feature included in Veritas InfoScale Storage Foundation Standard and Enterprise products that enables applications access preallocated VxFS files as raw character devices. Quick I/O provides the administrative benefits of running databases on file systems without the typically associated degradation in performance.

Note: Quick I/O is not supported on Linux.

How Quick I/O improves database performance

The benefits of using Quick I/O are:

- Improved performance and processing throughput by having Quick I/O files act as raw devices.
- Ability to manage Quick I/O files as regular files, which simplifies administrative tasks such as allocating, moving, copying, resizing, and backing up Oracle datafiles.

  Note: Veritas recommends using Oracle Disk Manager.

Note: Quick I/O is not supported on Linux.

Quick I/O's ability to access regular files as raw devices improves database performance by:

Table 12-1

Quick I/O feature	Advantage
Supporting direct I/O	I/O on files using `read`() and `write`() system calls typically results in data being copied twice: once between user and kernel space, and later between kernel space and disk. In contrast, I/O on raw devices is direct. That is, data is copied directly between user space and disk, saving one level of copying. As with I/O on raw devices, Quick I/O avoids extra copying.

Table 12-1 *(continued)*

Quick I/O feature	Advantage
Avoiding kernel write locks on database files	When database I/O is performed using the `write`() system call, each system call acquires and releases a write lock inside the kernel. This lock prevents multiple simultaneous write operations on the same file. Because database systems usually implement their own locking to manage concurrent access to files, per file writer locks unnecessarily serialize I/O operations. Quick I/O bypasses file system per file locking and lets the database server control data access.
Avoiding double buffering	Most database servers maintain their own buffer cache and do not need the file system buffer cache. Database data cached in the file system buffer is therefore redundant and results in wasted memory and extra system CPU utilization to manage the buffer. By supporting direct I/O, Quick I/O eliminates double buffering. Data is copied directly between the relational database management system (RDBMS) cache and disk, which lowers CPU utilization and frees up memory that can then be used by the database server buffer cache to further improve transaction processing throughput.
For AIX: Supporting AIX Fastpath asynchronous I/O	AIX Fastpath asynchronous I/O is a form of I/O that performs non-blocking system level reads and writes, allowing the system to handle multiple I/O requests simultaneously. Operating systems such as AIX provide support for asynchronous I/O on raw devices, but not on regular files. As a result, even if the database server is capable of using asynchronous I/O, it cannot issue asynchronous I/O requests when the database runs on file systems. Lack of asynchronous I/O significantly degrades performance. Quick I/O lets the database server take advantage of kernel-supported asynchronous I/O on file system files accessed using the Quick I/O interface.

Table 12-1 *(continued)*

Quick I/O feature	Advantage
For Solaris: Supporting kernel asynchronous I/O	Solaris kernel asynchronous I/O is a form of I/O that performs non-blocking system level reads and writes, allowing the system to handle multiple I/O requests simultaneously. Operating systems such as Solaris provide kernel support for asynchronous I/O on raw devices, but not on regular files. As a result, even if the database server is capable of using asynchronous I/O, it cannot issue asynchronous I/O requests when the database runs on file systems. Lack of asynchronous I/O significantly degrades performance. Quick I/O enables the database server to take advantage of kernel-supported asynchronous I/O on file system files accessed using the Quick I/O interface.

Creating Oracle database files as Quick I/O files using qiomkfile

The best way to preallocate space for tablespace containers and to make them accessible using the Quick I/O interface is to use the `qiomkfile`. You can use the `qiomkfile` to create the Quick I/O files for either temporary or permanent tablespaces.

Note: Quick I/O is not supported on Linux.

Prerequisites

- You can create Quick I/O files only on VxFS file systems.
- If you are creating database files on an existing file system, run `fsadm` (or similar utility) to report and eliminate fragmentation.
- You must have read/write permissions on the directory in which you intend to create Oracle Quick I/O files.

Usage notes

- The `qiomkfile` command creates two files: a regular file with preallocated, contiguous space, and a file that is a symbolic link pointing to the Quick I/O name extension.
- See the `qiomkfile`(1M) manual page for more information.

`-a` Creates a symbolic link with an absolute path name for a specified file. Use the `-a` option when absolute path names are required. However, the default is to create a symbolic link with a relative path name.

`-e` Extends a file by a specified amount to allow Oracle tablespace resizing.

See "Extending a Quick I/O file" on page 133.

`-h` Specifies the Oracle datafile header size. This option specifies a header that will be allocated in addition to the size specified because Oracle requires one additional database block for all its datafiles. If this option is used, the resulting file can be used as an Oracle datafile. When creating an Oracle datafile, the header size should be equal to the Oracle block size (as determined by the `DB_BLOCK_SIZE` parameter). If the header size is missing when the `-h` option is used, a 32K header will be allocated.

`-r` Increases the file to a specified size to allow Oracle tablespace resizing.

See "Extending a Quick I/O file" on page 133.

`-s` For AIX and Solaris:

Specifies the space to preallocate for a file in bytes, kilobytes, megabytes, gigabytes, or sectors (512 bytes) by adding a `k`, `K`, `m`, `M`, `g`, `G`, `s`, or `S` suffix.

The default is bytes—you do not need to attach a suffix to specify the value in bytes. The size of the file that is preallocated is the total size of the file (including the header) rounded to the nearest multiple of the file system block size.

Warning: Exercise caution when using absolute path names. Extra steps may be required during database backup and restore procedures to preserve symbolic links. If you restore files to directories different from the original paths, you must change the symbolic links that use absolute path names to point to the new path names before the database is restarted.

To create a database file as a Quick I/O file using qiomkfile

1. Create a database file using the `qiomkfile` command:

   ```
   # /opt/VRTS/bin/qiomkfile -h headersize -s file_size
   /mount_point/filename
   ```

2. Change the owner and group permissions on the file:

   ```
   # chown oracle:dba .filename

   # chmod 660 .filename
   ```

3. Create tablespaces on this file using SQL*Plus statements.

   For example:

   ```
   $ sqlplus /nolog

   SQL> connect / as sysdba

   SQL> create tablespace ts1 datafile '/mount_point/filename.dbf'

   size 100M reuse;

   exit;
   ```

An example to show how to create a 100MB database file named `dbfile` on the VxFS file system `/db01` using a relative path name:

```
# /opt/VRTS/bin/qiomkfile -h 32k -s 100m /db01/dbfile

# ls -al
```

For AIX:

```
-rw-r--r--   1 oracle   dba   104857600   Oct 2 13:42    .dbfile
```

For Solaris:

```
-rw-r--r--   1 oracle   dba   104890368   Oct 2 13:42    .dbfile

lrwxrwxrwx   1 oracle   dba          19   Oct 2 13:42    dbfile -> \
.dbfile::cdev:vxfs:
```

In the example, `qiomkfile` creates a regular file named `/db01/.dbfile`, which has the real space allocated. Then, `qiomkfile` creates a symbolic link named `/db01/dbfile`. This symbolic link is a relative link to the Quick I/O interface for `/db01/.dbfile`, that is, to the `.dbfile::cdev:vxfs:` file. The symbolic link allows

`.dbfile` to be accessed by any database or application using its Quick I/O interface.

Preallocating space for Quick I/O files using the setext command

As an alternative to using the `qiomkfile` command, you can also use the VxFS `setext` command to preallocate space for database files.

Before preallocating space with `setext`, make sure the following conditions have been met:

Prerequisites	■ The `setext` command requires superuser (`root`) privileges.
Usage notes	■ You can use the `chown` command to change the owner and group permissions on the file after you create it. See the `setext` (1M) manual page for more information.

To create a Quick I/O database file using setext

1. Access the VxFS mount point and create a file:

```
# cd /mount_point

# touch .filename
```

2. Use the `setext` command to preallocate space for the file:

```
# /opt/VRTS/bin/setext -r size -f noreserve -f chgsize  \
.filename
```

3. Create a symbolic link to allow databases or applications access to the file using its Quick I/O interface:

```
# ln -s .filename::cdev:vxfs: filename
```

4. Change the owner and group permissions on the file:

   For AIX:

```
# chown user:group .filename
```

   For Solaris:

```
# chown oracle:dba .filename

# chmod 660 .filename
```

   An example to show how to access the mount point for Oracle `/db01`, create a datafile, preallocate the space, and change the permissions:

```
# cd /db01
# touch .dbfile
# /opt/VRTS/bin/setext -r 100M -f noreserve -f chgsize .dbfile
# ln -s .dbfile::cdev:vxfs: dbfile

# chown oracle:dba .dbfile

# chmod 660 .dbfile
```

Accessing regular VxFS files as Quick I/O files

You can access regular VxFS files as Quick I/O files using the `::cdev:vxfs:` name extension.

While symbolic links are recommended because they provide easy file system management and location transparency of database files, the drawback of using symbolic links is that you must manage two sets of files (for instance, during database backup and restore).

Usage notes

- When possible, use relative path names instead of absolute path names when creating symbolic links to access regular files as Quick I/O files. Using relative path names prevents copies of the symbolic link from referring to the original file when the directory is copied. This is important if you are backing up or moving database files with a command that preserves the symbolic link.
  However, some applications require absolute path names. If a file is then relocated to another directory, you must change the symbolic link to use the new absolute path. Alternatively, you can put all the symbolic links in a directory separate from the data directories. For example, you can create a directory named `/database` and put all the symbolic links there, with the symbolic links pointing to absolute path names.

To access an existing regular file as a Quick I/O file on a VxFS file system

1. Access the VxFS file system mount point containing the regular files:

```
$ cd /mount_point
```

2. Create the symbolic link:

```
$ mv filename .filename
$ ln -s .filename::cdev:vxfs: filename
```

This example shows how to access the VxFS file `dbfile` as a Quick I/O file:

```
$ cd /db01
$ mv dbfile .dbfile
$ ln -s .dbfile::cdev:vxfs: dbfile
```

This example shows how to confirm the symbolic link was created:

```
$ ls -lo .dbfile dbfile
```

For AIX:

```
lrwxrwxrwx  1 oracle      19     Oct 2 13:42   dbfile ->
.dbfile::vxcdev:vxfs:
```

For Solaris:

```
lrwxrwxrwx  1 oracle      19     Oct 2 13:42   dbfile ->
.dbfile::cdev:vxfs:
```

Converting Oracle files to Quick I/O files

Special commands, available in the `/opt/VRTSdbed/bin` directory, are provided to assist you in converting an existing database to use Quick I/O. You can use the `qio_getdbfiles` command to extract a list of file names from the database system tables and the `qio_convertdbfiles` command to convert this list of database files to use Quick I/O.

Note: It is recommended that you create a Storage Checkpoint before converting to or from Quick I/O.

Before converting database files to Ouick I/O files, the following conditions must be met:

Prerequisites

- Log in as the Database Administrator (typically, the user ID `oracle`) to run the `qio_getdbfiles` and `qio_convertdbfiles` commands.
- You must predefine the Oracle environment variable `$ORACLE_SID`. Change to the ORACLE_SID environment variable must be defined.
- Files you want to convert must be regular files on VxFS file systems or links that point to regular VxFS files

Usage notes

- Converting existing database files to Quick I/O files may not be the best choice if the files are fragmented. Use of the `-f` option to determine the fragmentation levels is not supported for 5.1.
- If you choose to create new files, they will be contiguous. You must then move data from the old files to the new files using the `dd`(1M) command or a database import facility, and then define the new files to the database.
- By default, `qio_getdbfiles` skips any tablespaces marked `TEMPORARY`. Tablespaces marked `TEMPORARY` can be sparse, which means that not all blocks in the file are allocated. Quick I/O files cannot be sparse, as Quick I/O provides a raw type interface to storage. If a sparse file is converted to a Quick I/O file, the Oracle instance can fail if Oracle attempts to write into one of these unallocated blocks.
  See "Handling Oracle temporary tablespaces and Quick I/O" on page 130.
- You may also want to consider creating Quick I/O files for temporary tablespaces.
  See "Creating Oracle database files as Quick I/O files using qiomkfile" on page 118.
- The `qio_convertdbfiles` command exits and prints an error message if any of the database files are not on a VxFS file system. If this happens, you must remove any non-VxFS files from the `mkqio.dat` file before running the `qio_convertdbfiles` command.
- Instead of using the `qio_getdbfiles` command, you can manually create the `mkqio.dat` file containing the Oracle database filenames that you want to convert to Quick I/O files.

Command usage for `qio_getdbfiles` is as follows:

```
qio_getdbfiles SFUA qio_getdbfiles ERROR V-81-4265
```

The following options are available for the `qio_getdbfiles` command:

`-a`	Lets you include all datafiles, including those that are potentially sparse. (Use this option only for debugging purposes, as sparse files are not candidates for use with Quick I/O.)
`-T`	Lets you specify the type of database as `ora`.

Command usage for `qio_getdbfiles` is as follows:

```
qio_getdbfiles [ -T database_type ] [ options ]
```

The following options are available for the `qio_convertdbfiles` command:

`-a`	Changes regular files to Quick I/O files using absolute path names. Use this option when symbolic links need to point to absolute path names (for example, at a site that uses SAP).
`-h`	Displays a help message.
	Creates the extra links for all datafiles and log files in the `/dev` directory to support SAP's `brbackup`.
`-T`	Lets you specify the type of database as `ora`.
`-u`	Changes Quick I/O files back to regular files. Use this option to undo changes made by a previous run of the `qio_convertdbfiles` script.

To extract a list of Oracle files to convert

- With the database instance up and running, run the `qio_getdbfiles` command from a directory for which you have write permission:

  ```
  $ cd /extract_directory
  ```

  ```
  $ /opt/VRTSdbed/bin/qio_getdbfiles -T ora
  ```

  The `qio_getdbfiles` command extracts the list file names from the database system tables and stores the file names and their size in bytes in a file called `mkqio.dat` under the current directory.

  Note: Alternatively, you can manually create the `mkqio.dat` file containing the Oracle database file names that you want to convert to use Quick I/O. You can also manually edit the `mkqio.dat` file generated by `qio_getdbfiles`, and remove files that you do not want to convert to Quick I/O files.

  Note: To run the `qio_getdbfiles` command, you must have permission to access the database and permission to write to the `/extract_directory`.

  The `mkqio.dat` list file should look similar to the following:

  ```
  /data11r1/VRTS11r1/redo01.log 52428800
  /data11r1/VRTS11r1/redo02.log 52428800
  /data11r1/VRTS11r1/redo03.log 52428800
  /data11r1/VRTS11r1/sysaux01.dbf 632553472
  /data11r1/VRTS11r1/system01.dbf 754974720
  /data11r1/VRTS11r1/undotbs01.dbf 47185920
  /data11r1/VRTS11r1/users01.dbf 5242880
  /data11r1/nqio1.dbf 104857600
  ```

To convert the Oracle database files to Quick I/O files

1. Shut down the database.
2. Run the `qio_convertdbfiles` command from the directory containing the `mkqio.dat` file:

   ```
   $ cd /extract_directory
   ```

   ```
   $ /opt/VRTSdbed/bin/qio_convertdbfiles
   ```

   The list of files in the `mkqio.dat` file is displayed. For example:

   ```
   file1 --> .file1::cdev:vxfs:
   file2 --> .file2::cdev:vxfs:
   file3 --> .file3::cdev:vxfs:
   file4 --> .file4::cdev:vxfs:
   file5 --> .file5::cdev:vxfs:
   ```

   Run the `qio_convertdbfiles` command (with no options specified) to rename the file *filename* to `.filename` and creates a symbolic link to `.filename` with the Quick I/O extension. By default, the symbolic link uses a relative path name.

   The `qio_convertdbfiles` script exits and prints an error message if any of the database files are not on a VxFS file system. If this happens, you must remove any non-VxFS files from the `mkqio.dat` file before running the `qio_convertdbfiles` command again.
3. Start up the database.
4. You can now access these database files using the Quick I/O interface.

To undo the previous run of qio_convertdbfiles and change Quick I/O files back to regular VxFS files

1. If the database is running, shut it down.
2. Run the following command from the directory containing the `mkqio.dat` file:

   ```
   $ cd /extract_directory
   ```

   ```
   $ /opt/VRTSdbed/bin/qio_convertdbfiles -u
   ```

   The list of Quick I/O files in the `mkqio.dat` file is displayed. For example:

   ```
   .file1::cdev:vxfs: --> file1
   .file2::cdev:vxfs: --> file2
   .file3::cdev:vxfs: --> file3
   .file4::cdev:vxfs: --> file4
   .file5::cdev:vxfs: --> file5
   ```

   The `qio_convertdbfiles` command with the undo option (`-u`) specified renames the files from <.filename> to <filename> and undoes the symbolic link to `.filename` that was created along with the Quick I/O files.

About sparse files

Support for sparse files lets applications store information (in inodes) to identify data blocks that have only zeroes, so that only blocks containing non-zero data have to be allocated on disk.

For example, if a file is 10KB, it typically means that there are blocks on disk covering the whole 10KB. Assume that you always want the first 9K to be zeroes. The application can go to an offset of 9KB and write 1KB worth of data. Only a block for the 1KB that was written is allocated, but the size of the file is still 10KB.

The file is now sparse. It has a hole from offset 0 to 9KB. If the application reads any part of the file within this range, it will see a string of zeroes.

If the application subsequently writes a 1KB block to the file from an offset of 4KB, for example, the file system will allocate another block.

The file then looks like:

- 0-4KB - hole
- 4-5KB - data block
- 5-9KB - hole
- 9-10KB - data block

So a 1TB file system can potentially store up to 2TB worth of files if there are sufficient blocks containing zeroes. Quick I/O files cannot be sparse and will always have all blocks specified allocated to them.

Handling Oracle temporary tablespaces and Quick I/O

You can create a new temporary tablespace using Quick I/O files. However, you cannot convert existing temporary tablespaces which use regular files to Quick I/O with the `qio_getdbfiles` command on Oracle9.

By default, `qio_getdbfiles` skips any tablespaces marked `TEMPORARY` because they can be sparse, which means that not all blocks in the file are allocated. Quick I/O files cannot be sparse, as Quick I/O provides a raw-type interface to storage. If a sparse file is converted to a Quick I/O file, the Oracle instance can fail if Oracle attempts to write into one of these unallocated blocks. When you initially create a temporary tablespace on Quick I/O files, however, Oracle sees them as raw devices and does not create sparse files.

To convert a temporary tablespace using regular files to Quick I/O files, you can drop your existing temporary tablespaces which use regular files and recreate them using Quick I/O files. You can also leave the temporary tablespaces as regular files.

To obtain a list of file names that are not temporary

- Use the following SQL statements:

```
$ sqlplus /nolog
 SQL> connect / as sysdba;
 SQL> select file_name from dba_data_files a,
 dba_tablespaces b where a.tablespace_name =
 b.tablespace_name and b.contents <> 'TEMPORARY';
```

To drop an existing temporary tablespace and recreate using Quick I/O files

1. Drop the temporary tablespace, including its contents:

```
$ sqlplus /nolog
SQL> connect / as sysdba;
SQL> drop tablespace tablespace_name including contents;
```

2. Create a Quick I/O file on a VxFS file system:

```
# /opt/VRTS/bin/qiomkfile -h header_size -s size \
/mount_point/filename.dbf
```

3 Change the owner and group permissions on the file

```
# chown oracle:dba .filename

# chmod 660 .filename
```

4 Create a new temporary tablespace using the `create temporary tablespace` command.

To use the `create temporary tablespace` command:

```
$ sqlplus /nolog
SQL> connect / as sysdba;
SQL> create temporary tablespace tablespace_name \
tempfile '/mount_point/new_filename.dbf' size size reuse;
```

This example shows how to drop tablespace `tempts`, create a Quick I/O file `temp01.dbf`, change permissions and then create a new temporary tablespace `tempts` using the `create temporary tablespace` command:

```
$ sqlplus /nolog
SQL> connect / as sysdba;
SQL> drop tablespace tempts including contents;
Tablespace dropped.
# /opt/VRTS/bin/qiomkfile -h 32k -s 100M /db01/temp01.dbf

# chown oracle:dba .temp01
# chmod 660 .temp01

$ sqlplus /nolog
SQL> connect / as dba;
SQL> create temporary tablespace tempts \
tempfile '/db01/temp01.dbf' \
size 100M reuse;
Tablespace created.
```

Displaying Quick I/O status and file attributes

You can obtain and display information about Quick I/O status and file attributes using various options of the `ls` command:

-al Lists all files on a file system, including Quick I/O files and their links.

-1L Shows if Quick I/O was successfully installed and enabled.

-a1L Shows how a Quick I/O file name is resolved to that of a raw device.

To list all files on the current file system, including Quick I/O files and their links

- Use the `ls -al` command with the file names:

  ```
  $ ls -al filename .filename
  ```

  The following example shows how to use the `-a` option to display the absolute path name created using `qiomkfile`:

```
$ ls -la /snap_data11r2/FLAS11r2/.qfile01.dbf /snap_data11r2/FLAS11r2/qfile01.dbf
lrwxrwxrwx  1 ora11gr2  dba  24 Jul 15 03:27 /snap_data11r2/FLAS11r2/.qfile01.dbf -> \
.qfile01.dbf::cdev:vxfs:
lrwxrwxrwx  1 ora11gr2  dba  48 Jul 15 05:17 /snap_data11r2/FLAS11r2/qfile01.dbf -> \
/snap_data11r2/FLAS11r2/.qfile01.dbf::cdev:vxfs:
```

To determine if an Oracle datafile has been converted to Quick I/O

- Use the `ls` command as follows:

  ```
  $ ls -lL filename
  ```

  The following example shows how to determine if Quick I/O is installed and enabled:

  ```
  $ ls -lL dbfile
  ```

  ```
  crw-r--r--  1 oracle  dba   45, 1        Oct 2 13:42  dbfile
  ```

  For AIX:

  where the first character, `c`, indicates it is a raw character device file, and the major and minor device numbers are displayed in the size field. If you see a `No such file or directory` message, Quick I/O did not install properly or does not have a valid license key.

To show a Quick I/O file resolved to a raw device

- Use the `ls` command with the file names as follows:

  ```
  $ ls -alL filename .filename
  ```

  The following example shows how the Quick I/O file name `dbfile` is resolved to that of a raw device:

  For AIX:

  ```
  crw-r--r--   1 oracle   dba      45,  1        Oct 2 13:42    dbfile

  $ ls -alL d* .d*
  ```

  For Solaris:

  ```
  crw-r--r--   1 oracle   dba      45,  1        Oct 2 13:42    dbfile
  ```

Extending a Quick I/O file

Although Quick I/O files must be preallocated, they are not limited to the preallocated sizes. You can grow or “extend” a Quick I/O file by a specific amount or to a specific size, using options to the `qiomkfile` command. Extending Quick I/O files is a fast, online operation and offers a significant advantage over using raw devices.

Before extending a Quick I/O file, make sure the following conditions have been met:

Prerequisites

- You must have sufficient space on the file system to extend the Quick I/O file.

Usage notes

- You can also grow VxFS file systems online (provided the underlying disk or volume can be extended) using the `fsadm` command. You can expand the underlying volume and the filesystem with the `vxresize` command.
- You must have superuser (`root`) privileges to resize VxFS file systems using the `fsadm` command.
- See the `fsadm_vxfs` (1M) and `qiomkfile` (1M) manual pages for more information.

The following options are available with the `qiomkfile` command:

`-e`	Extends the file by a specified amount to allow Oracle tablespace resizing.
`-r`	Increases the file to a specified size to allow Oracle tablespace resizing.

To extend a Quick I/O file

1. If required, ensure the underlying storage device is large enough to contain a larger VxFS file system (see the `vxassist`(1M) manual page for more information), and resize the VxFS file system using `fsadm` command:
2. Extend the Quick I/O file using the `qiomkfile` command:

   ```
   $ /opt/VRTS/bin/qiomkfile -e extend_amount /mount_point/filename
   ```

   or

   ```
   $ /opt/VRTS/bin/qiomkfile -r newsize /mount_point/filename
   ```

   An example to show how to grow VxFS file system:

   `/db01` to 500MB and extend the `emp.dbf` Quick I/O file by 20MB:

   ```
   $ /opt/VRTS/bin/qiomkfile -e 20M /db01/emp.dbf
   ```

   ```
   # /opt/VRTS/bin/fsadm -b 500M /db01
   ```

   An example to show how to grow VxFS file system:

   `/db01` to 500MB and resize the `emp.dbf` Quick I/O file to 300MB:

   ```
   $ /opt/VRTS/bin/qiomkfile -r 300M /db01/emp.dbf
   ```

   ```
   # /opt/VRTS/bin/fsadm -b 500M /db01
   ```

Using Oracle's AUTOEXTEND with Quick I/O files

Oracle supports an automatic extend feature that automatically grows a database file by a prespecified amount, up to a prespecified maximum size.

For regular file system files, `AUTOEXTEND` works transparently, provided the underlying file system has enough space. For example, suppose the current size of a database file `emp.dbf` is 100MB, but this file is expected to triple in size over time. To

accommodate this growth using AUTOEXTEND feature, you can specify the next size at 20MB and maxsize at 300MB. This will automatically grow the file by 20MB until its size reaches 300MB. For example:

```
alter database datafile 'emp.dbf' autoextend on next 20m \
maxsize 300m;
```

(See the Oracle Server SQL Reference Guide for more information about the alter database command, as well as the next and maxsize parameters.)

Note: You must have sufficient space on the underlying file system to AUTOEXTEND a file, and the underlying storage device must be large enough to contain the new, larger file system.

For Quick I/O files or raw devices, AUTOEXTEND does not know how to grow the underlying Quick I/O files or devices. Therefore, the Quick I/O file size must be large enough to accommodate the new size before AUTOEXTEND can grow the datafile.

You can use AUTOEXTEND with Quick I/O files in the following ways:

- Preallocate the Quick I/O file to a size at least as big as the maximum growth size expected for this database file.
  Using this method, you would need to preallocate the Quick I/O file emp.dbf for the entire 300MB. The drawback is that this can unnecessarily lock up excess disk space. Raw devices have a similar requirement.
- Monitor the free space available in the Quick I/O file, and grow the file as necessary with the qiomkfile command.
  Unlike raw devices, you can easily extend Quick I/O files online. Using this method, you can monitor the free space available in the Oracle datafiles and use the qiomkfile command to grow the Quick I/O files online as and when needed (typically when the file is about 80 to 90 percent full). This method does not require you to lock out unused disk space for Quick I/O files. The free space on the file system is available for use by other applications.

The following options are available for the qiomkfile command:

-e	Extends the file by a specified amount to allow Oracle tablespace resizing.
-r	Increases the file to a specified size to allow Oracle tablespace resizing.

You can grow underlying VxFS file systems online (provided the underlying disk or volume can be extended) using the `fsadm` command. See the `fsadm_vxfs`(1M) manual page for more information.

To monitor the free space available in an Oracle tablespace

- Check the free space currently available in the Oracle tablespace using the following Oracle SQL command:

```
$ sqlplus /nolog
SQL> connect / as sysdba;
SQL> select * from dba_free_space where \
tablespace_name = 'tablespace_name';
SQL> exit
```

To extend a Quick I/O file using qiomkfile

- If the datafile is running low on free blocks, use the `qiomkfile` command to extend the Quick I/O file:

```
$ /opt/VRTS/bin/qiomkfile -e extend_amount \
  /mount_point/filename
```

The following example shows how to monitor the free space on the tablespace `EMP` on file system `/db01`:

```
$ sqlplus /nolog
SQL> connect / as sysdba;
SQL> select * from dba_free_space where tablespace_name = 'EMP';
SQL> exit
```

The following example shows how to extend the Oracle datafile `emp.dbf` by 20MB (the specified `next` size) using the `qiomkfile` command:

```
$ /opt/VRTS/bin/qiomkfile -e 20M /db01/emp.dbf
```

Recreating Quick I/O files after restoring a database

If you need to restore your database and were using Quick I/O files, you can use the `qio_recreate` command to automatically recreate the Quick I/O files after you have performed a full database recovery. The `qio_recreate` command uses the `mkqio.dat` file, which contains a list of the Quick I/O files used by the database and the file sizes.

For information on recovering your database, refer to the documentation that came with your database software.

Before recreating Quick I/O with the `qio_recreate` command, make sure the following conditions have been met:

Oracle Prerequisites

- Recover your database before attempting to recreate the Quick I/O files.
- You may be logged in as either the database administrator or `root` to run the `qio_recreate` command.
- In the directory from which you run the `qio_recreate` command, you must have an existing `mkqio.dat` file.
- The ORACLE_SID environment variable must be set.
  See "Converting Oracle files to Quick I/O files" on page 124.

Usage notes

- The `qio_recreate` command supports only conventional Quick I/O files.
- Refer to the `qio_recreate`(1M) manual page for more information.

To recreate Quick I/O files after recovering a database

◆ As Oracle DBA, use the `qio_recreate` command as follows:

```
$ /opt/VRTSdbed/bin/qio_recreate -T ora
```

You will not see any output if the command is successful.

When you run the `qio_recreate` command, the following actions occur:

If...	Then...
a Quick I/O file is missing	the Quick I/O file is recreated.
a symbolic link from a regular VxFS file to a Quick I/O file is missing	the symbolic link is recreated.
a symbolic link and its associated Quick I/O file are missing	both the link and the Quick I/O file are recreated.
a Quick I/O file is missing and the regular VxFS file that it is symbolically linked to is not the original VxFS file	the Quick I/O file is not recreated and a warning message is displayed.
a Quick I/O file is smaller than the size listed in the `mkqio.dat` file	the Quick I/O file is not recreated and a warning message is displayed.

Disabling Quick I/O

If you need to disable the Quick I/O feature, you first need to convert any Quick I/O files back to regular VxFS files. Then, remount the VxFS file system using a special mount option.

Before disabling Quick I/O, make sure the following condition has been met:

Prerequisite

For AIX:

The file system you are planning to remount must be located in the `/etc/filesystems` file.

For Solaris:

The file system you are planning to remount must be located in the `/etc/vfstab` file.

To disable Quick I/O

1. If the database is running, shut it down.
2. To change Quick I/O files back to regular VxFS files, run the following command from the directory containing the `mkqio.dat` list:

   ```
   $ /opt/VRTSdbed/bin/qio_convertdbfiles -u
   ```

   The list of Quick I/O files in the `mkqio.dat` file is displayed. For example:

   ```
   .file1::cdev:vxfs: --> file1
   .file2::cdev:vxfs: --> file2
   .file3::cdev:vxfs: --> file3
   .file4::cdev:vxfs: --> file4
   .file5::cdev:vxfs: --> file5
   ```

   The `qio_convertdbfiles` command with the undo option (`-u`) renames the files from .filename to filename and removes the symbolic link to .filename that was created along with the Quick I/O files.
3. To remount the file system with Quick I/O disabled, use the `mount -o noqio` command as follows:

   For AIX:

   ```
   # /opt/VRTS/bin/mount -V vxfs -o remount,noqio special
        /mount_point
   ```

   For Solaris:

   ```
   # /opt/VRTS/bin/mount -F vxfs -o remount,noqio /mount_point
   ```

Creating Quick I/O files in Solaris local zone

Quick I/O files cannot be created in the Solaris local zones. This is because the VxFS command `qiomkfile` cannot run in the Solaris local zone.

You must use the following workaround procedure to create Quick I/O files in Solaris local zone. You must perform the following commands as **root** unless otherwise stated.

To create Quick I/O files in Solaris local zone

1. Install `VRTSodm` on the Solaris global zone.

   You have to perform this step once on your system. You do not have to repeat it for each local zone.

2. Install your Veritas InfoScale Storage Foundation product license on the Solaris global zone.

   You need to perform this step once on your system. You do not need to repeat it for each local zone.

3. Export the Veritas InfoScale license from the Solaris global zone to each Solaris local zone where you plan to use Quick I/O.

   To do so, enter the following sequence of commands:

   ```
   zonecfg -z <zone_name>
   ```

   For example, if the Solaris local zone name is **zone2**, then:

   ```
   # zonecfg -z zone2
     zonecfg:zone2> add fs
     zonecfg:zone2:fs> set dir=/etc/vx/licenses/lic
     zonecfg:zone2:fs> set special=/etc/vx/licenses/lic
     zonecfg:zone2:fs> set type=lofs
     zonecfg:zone2:fs> end
     zonecfg:zone2> verify
     zonecfg:zone2> commit
     zonecfg:zone2> exit
   ```

4 In each local zone, mount the **odm** device to `/dev/odm`.

To do so, use the following command in the Solaris global zone:

```
# mkdir /zones/<zone_name>/dev/odm
```

Then use the following command in the Solaris local zone:

```
# mount -F odm /dev/odm /dev/odm
```

For example, if the Solaris local zone name is **zone2**, you must use the following commands:

```
# mkdir /zones/zone2/dev/odm
# mount -F odm /dev/odm /dev/odm
```

5 As Oracle user, create a Quick I/O file in a Solaris local zone utilizing `/opt/VRTS/bin/odmmkfile`. Rename the file to a Quick I/O file.

To do so, enter the following sequence of commands:

```
# su - <oracle_dba>
# cd <target_dir>
# /opt/VRTS/bin/odmmkfile -s <size> filename
# mv filename .filename
# ln -s .filename::cdev:vxfs: filename
# chown <Oracle_dba> .filename
```

Chapter 13

Improving database performance with Cached Quick I/O

This chapter includes the following topics:

- About Cached Quick I/O
- Tasks for setting up Cached Quick I/O
- Enabling Cached Quick I/O on a file system
- Determining candidates for Cached Quick I/O
- Enabling and disabling Cached Quick I/O for individual files

About Cached Quick I/O

Veritas Cached Quick I/O maintains and extends the database performance benefits of Veritas Quick I/O by making more efficient use of large, unused system memory through a selective buffering mechanism. Cached Quick I/O also supports features that support buffering behavior, such as file system read-ahead.

Enabling Cached Quick I/O on suitable Quick I/O files improves database performance by using the file system buffer cache to store data. This data storage speeds up system reads by accessing the system buffer cache and avoiding disk I/O when searching for information.

Having data at the cache level improves database performance in the following ways:

- For read operations, Cached Quick I/O caches database blocks in the system buffer cache, which can reduce the number of physical I/O operations and therefore improve read performance.
- For write operations, Cached Quick I/O uses a direct-write, copy-behind technique to preserve its buffer copy of the data. After the direct I/O is scheduled and while it is waiting for the completion of the I/O, the file system updates its buffer to reflect the changed data being written out. For online transaction processing, Cached Quick I/O achieves better than raw device performance in database throughput on large platforms with very large physical memories.
- For sequential table scans, Cached Quick I/O can significantly reduce the query response time because of the read-ahead algorithm used by Veritas File System. If a user needs to read the same range in the file while the data is still in cache, the system is likely to return an immediate cache hit rather than scan for data on the disk.

Note: Cached Quick I/O is not supported on Linux.

Tasks for setting up Cached Quick I/O

To set up and use Cached Quick I/O, you should do the following in the order in which they are listed:

- Enable Cached Quick I/O on the underlying file systems used for your database.
- Exercise the system in your production environment to generate file I/O statistics.
- Collect the file I/O statistics while the files are in use.
- Analyze the file I/O statistics to determine which files benefit from Cached Quick I/O.
- Disable Cached Quick I/O on files that do not benefit from caching.

Note: Cached Quick I/O is not supported on Linux.

Enabling Cached Quick I/O on a file system

Cached Quick I/O depends on Veritas Quick I/O running as an underlying system enhancement in order to function correctly. Follow the procedures listed here to ensure that you have the correct setup to use Cached Quick I/O successfully.

Prerequisites

- You must have permission to change file system behavior using the `vxtunefs` command to enable or disable Cached Quick I/O. By default, you need superuser (`root`) permissions to run the `vxtunefs` command, but other system users do not. Superuser (`root`) must specifically grant database administrators permission to use this command as follows:
  For AIX and Solaris:

  ```
  # chown root:dba opt/VRTS/bin/vxtunefs
  ```

  ```
  # chmod 4550 /opt/VRTS/bin/vxtunefs
  ```

  where users belonging to the `dba` group are granted permission to run the `vxtunefs` command. Symantec recommends this selective, more secure approach for granting access to powerful commands.
- You must enable Quick I/O on the file system. Quick I/O is enabled automatically at file system mount time.

Usage notes

- If Cached Quick I/O is enabled, it is recommended that you monitor any paging activity to the swap device on your database servers. You can use the `vmstat -I` command to monitor swap device paging. If swap device paging is observed, proper AIX Virtual Memory Manager (VMM) tuning is required to improve database performance.

Usage notes

For AIX:

- If Cached Quick I/O is enabled, it is recommended that you monitor any paging activity to the swap device on your database servers. You can use the `vmstat -I` command to monitor swap device paging. If swap device paging is observed, proper AIX Virtual Memory Manager (VMM) tuning is required to improve database performance.

Enabling and disabling the qio_cache_enable flag

As superuser (`root`), set the `qio_cache_enable` flag using the `vxtunefs` command after you mount the file system.

To enable the qio_cache_enable flag for a file system

- Use the `vxtunefs` command as follows:

  For AIX and Solaris:

  ```
  # /opt/VRTS/bin/vxtunefs -s -o qio_cache_enable=1 /mount_point
  ```

  For example:

  For AIX and Solaris:

  ```
  # /opt/VRTS/bin/vxtunefs -s -o qio_cache_enable=1 /db02
  ```

  where `/db02` is a VxFS file system containing the Quick I/O files and setting the `qio_cache_enable` flag to “1” enables Cached Quick I/O. This command enables caching for all the Quick I/O files on this file system.

To disable the flag on the same file system

- Use the `vxtunefs` command as follows:

  For AIX and Solaris:

  ```
  # /opt/VRTS/bin/vxtunefs -s -o qio_cache_enable=0 /mount_point
  ```

  For example:

  For AIX and Solaris:

  ```
  # /opt/VRTS/bin/vxtunefs -s -o qio_cache_enable=0 /db02
  ```

  where `/db02` is a VxFS file system containing the Quick I/O files and setting the `qio_cache_enable` flag to “0” disables Cached Quick I/O. This command disables caching for all the Quick I/O files on this file system.

Making Cached Quick I/O settings persistent across reboots and mounts

You can make the Cached Quick I/O system setting persistent across reboots and mounts by adding a file system entry in the `/etc/vx/tunefstab` file.

Note: The `tunefstab` file is a user-created file. For information on how to create the file and add tuning parameters, see the `tunefstab` (4) manual page.

To enable a file system after rebooting

- Put the file system in the `/etc/vx/tunefstab` file and set the flag entry:

```
/dev/vx/dsk/dgname/volname qio_cache_enable=1
```

where:

- `/dev/vx/dsk/dgname/volname` is the name of a block device
- *dgname* is the name of the disk group
- *volname* is the name of the volume

For example:

```
/dev/vx/dsk/PRODdg/db01 qio_cache_enable=1
/dev/vx/dsk/PRODdg/db02 qio_cache_enable=1
```

where `/dev/vx/dsk/PRODdg/db01` is the block device on which the file system resides.

The `tunefstab` (4) manual pages contain information on how to add tuning parameters.

See the `tunefstab` (4) manual page.

Note: `vxtunefs` can specify a mount point or a block device; `tunefstab` must always specify a block device only.

Using vxtunefs to obtain tuning information

Check the setting of the `qio_cache_enable` flag for each file system using the `vxtunefs` command.

To obtain information on only the qio_cache_enable flag setting

◆ Use the `grep` command with `vxtunefs`:

For AIX and Solaris:

```
# /opt/VRTS/bin/vxtunefs /mount_point | grep qio_cache_enable
```

For example:

For AIX and Solaris:

```
# /opt/VRTS/bin/vxtunefs /db01 | grep qio_cache_enable
```

where `/db01` is the name of the file system. This command displays only the `qio_cache_enable` setting as follows:

```
qio_cache_enable = 0
```

You can also use the `vxtunefs` command to obtain a more complete list of I/O characteristics and tuning statistics.

See the `vxtunefs` (1) manual page.

To obtain information on all vxtunefs system parameters

◆ Use the `vxtunefs` command without `grep`:

```
# /opt/VRTS/bin/vxtunefs /mount_point
```

For example:

```
# /opt/VRTS/bin/vxtunefs /db01
```

The `vxtunefs` command displays output similar to the following:

```
 Filesystem i/o parameters for /db01
read_pref_io = 65536
read_nstream = 1
read_unit_io = 65536
write_pref_io = 65536
write_nstream = 1
write_unit_io = 65536
pref_strength = 10
buf_breakup_size = 131072
discovered_direct_iosz = 262144
max_direct_iosz = 1048576
default_indir_size = 8192
qio_cache_enable = 0
```

```
odm_cache_enable = 0
write_throttle = 0
max_diskq = 1048576
initial_extent_size = 1
max_seqio_extent_size = 2048
max_buf_data_size = 8192
hsm_write_prealloc = 0
read_ahead = 1
inode_aging_size = 0
inode_aging_count = 0
fcl_maxalloc = 887660544
fcl_keeptime = 0
fcl_winterval = 3600
fcl_ointerval = 600
oltp_load = 0
delicache_enable = 1
thin_friendly_alloc = 0
dalloc_enable = 1
dalloc_limit = 90
```

The `vxtunefs`(1) manual pages contain a complete description of `vxtunefs` parameters and the tuning instructions.

See the `vxtunefs`(1) manual page.

Determining candidates for Cached Quick I/O

Determining which files can benefit from Cached Quick I/O is an iterative process that varies with each application. For this reason, you may need to complete the following steps more than once to determine the best possible candidates for Cached Quick I/O.

Before determining candidate files for Quick I/O, make sure the following conditions have been met:

Prerequisites
- You must enable Cached Quick I/O for the file systems.

Usage notes
- See the `qiostat` (1M) manual page for more information.

Collecting I/O statistics

Once you have enabled Cached Quick I/O on a file system, you need to collect statistics to determine and designate the files that can best take advantage of its benefits.

To collect statistics needed to determine files that benefit from Cached Quick I/O

1. Reset the `qiostat` counters by entering:

   ```
   $ /opt/VRTS/bin/qiostat -r /mount_point/filenames
   ```

2. Run the database under full normal load and through a complete cycle (24 to 48 hours in most cases) to determine your system I/O patterns and database traffic in different usage categories (for example, OLTP, reports, and backups) at different times of the day.

3. While the database is running, run `qiostat -l` to report the caching statistics as follows:

   ```
   $ /opt/VRTS/bin/qiostat -l /mount_point/filenames
   ```

   or, use the `-i` option to see statistic reports at specified intervals:

   ```
   $ /opt/VRTS/bin/qiostat -i n /mount_point/filenames
   ```

   where n is time in seconds

   For example:

   To collect I/O statistics from all database files on file system `/db01`:

   ```
   $ /opt/VRTS/bin/qiostat -l /db01/*.dbf
   ```

About I/O statistics for Oracle

The output of the `qiostat` command is the primary source of information to use in deciding whether to enable or disable Cached Quick I/O on specific files. Statistics are printed in two lines per object.

The second line of information is defined as follows:

- `CREAD` is the number of reads from the VxFS cache (or total number of reads to Quick I/O files with cache advisory on)
- `PREAD` is the number of reads going to the disk for Quick I/O files with the cache advisory on
- `HIT RATIO` is displayed as a percentage and is the number of `CREADS` minus the number of `PREADS` times 100 divided by the total number of `CREADS`. The formula looks like this:

```
(CREADs - PREADs) * 100/ CREADs
```

The `qiostat -l` command output looks similar to the following:

```
                 OPERATIONS  FILE BLOCKS  AVG TIME(ms)
CACHE_STATISTICS

FILE NAME       READ  WRITE   READ WRITE    READ WRITE
CREAD  PREAD  HIT RATIO

/db01/cust.dbf   17128 9634   68509  38536   24.8  0.4
17124  15728  8.2

/db01/system.dbf   6     1      21    4      10.0  0.0
  6     6     0.0

/db01/stk.dbf    62552 38498  250213 153992  21.9  0.4
62567  49060  21.6

                 OPERATIONS  FILE BLOCKS  AVG TIME(ms)
CACHE_STATISTICS

FILE NAME       READ  WRITE   READ WRITE    READ WRITE
CREAD  PREAD  HIT RATIO

                 OPERATIONS  FILE BLOCKS  AVG TIME(ms)
CACHE_STATISTICS

FILE NAME       READ  WRITE   READ WRITE    READ WRITE
CREAD  PREAD  HIT RATIO
```

Analyze the output to find out where the cache-hit ratio is above a given threshold. A cache-hit ratio above 20 percent on a file for a given application may be sufficient to justify caching on that file. For systems with larger loads, the acceptable ratio may be 30 percent or above. Cache-hit-ratio thresholds vary according to the database type and load.

Using the sample output above as an example, the file `/db01/system.dbf` does not benefit from the caching because the cache-hit ratio is zero. In addition, the file receives very little I/O during the sampling duration.

However, the file `/db01/stk.dbf` has a cache-hit ratio of 21.6 percent. If you have determined that, for your system and load, this figure is above the acceptable threshold, it means the database can benefit from caching. Also, study the numbers reported for the read and write operations. When you compare the number of reads and writes for the `/db01/stk.dbf` file, you see that the number of reads is roughly twice the number of writes. You can achieve the greatest performance gains with Cached Quick I/O when using it for files that have higher read than write activity.

Based on these two factors, `/db01/stk.dbf` is a prime candidate for Cached Quick I/O.

See "Enabling and disabling Cached Quick I/O for individual files" on page 150.

Effects of read-aheads on I/O statistics

The number of `CREAD`s in the `qiostat` output is the total number of reads performed, including Cached Quick I/O, and the number of `PREAD`s is the number of physical reads. The difference between `CREAD`s and `PREAD`s (`CREADS` - `PREADS`) is the number of reads satisfied from the data in the file system cache. Thus, you expect that the number of `PREAD`s would always be equal to or lower than the number of `CREAD`s.

However, the `PREAD`s counter also increases when the file system performs read-aheads. These read-aheads occur when the file system detects sequential reads. In isolated cases where cache hits are extremely low, the output from `qiostat` could show that the number of `CREAD`s is lower than the number of `PREAD`s. The cache-hit ratio calculated against these `CREAD`/`PREAD` values is misleading when used to determine whether Cached Quick I/O should be enabled or disabled.

Under these circumstances, you can make a more accurate decision based on a collective set of statistics by gathering multiple sets of data points. Consequently, you might want to enable Cached Quick I/O for all the data files in a given tablespace, even if just one of the files exhibited a high cache-hit ratio.

Other tools for analysis

While the output of the `qiostat` command is the primary source of information to use in deciding whether to enable Cached Quick I/O on specific files, we also recommend using other tools in conjunction with `qiostat`. For example, benchmarking software that measures database throughput is also helpful. If a benchmark test in which Cached Quick I/O was enabled for a certain set of data files resulted in improved performance, you can also use those results as the basis for enabling Cached Quick I/O.

Enabling and disabling Cached Quick I/O for individual files

After using `qiostat` or other analysis tools to determine the appropriate files for Cached Quick I/O, you need to disable Cached Quick I/O for those individual files that do not benefit from caching using the `qioadmin` command.

Prerequisites

- Enable Cached Quick I/O for the file system before enabling or disabling Cached Quick I/O at the individual file level.

Usage notes

- You can enable or disable Cached Quick I/O for individual files while the database is online.
- You should monitor files regularly using `qiostat` to ensure that a file's cache-hit ratio has not changed enough to reconsider enabling or disabling Cached Quick I/O for the file.
- Enabling or disabling Cached Quick I/O for an individual file is also referred to as setting the cache advisory on or off.
- See the `qioadmin` (1) manual page.

Setting cache advisories for individual files

You can enable and disable Cached Quick I/O for individual files by changing the cache advisory settings for those files.

To disable Cached Quick I/O for an individual file

◆ Use the `qioadmin` command to set the cache advisory to `OFF` as follows:

```
$ /opt/VRTS/bin/qioadmin -S filename=OFF /mount_point
```

For example, to disable Cached Quick I/O for the file `/db01/system.dbf`, set the cache advisory to `OFF`:

```
$ /opt/VRTS/bin/qioadmin -S system.dbf=OFF /db01
```

To enable Cached Quick I/O for an individual file

◆ Use the `qioadmin` command to set the cache advisory to `ON` as follows:

```
$ /opt/VRTS/bin/qioadmin -S filename=ON /mount_point
```

For example, running `qiostat` shows the cache hit ratio for the file `/db01/system.dbf` reaches a level that would benefit from caching. To enable Cached Quick I/O for the file `/db01/system.dbf`, set the cache advisory to `ON`:

```
$ /opt/VRTS/bin/qioadmin -S system.dbf=ON /db01
```

Making individual file settings for Cached Quick I/O persistent

You can make the enable or disable individual file settings for Cached Quick I/O persistent across reboots and mounts by adding cache advisory entries in the `/etc/vx/qioadmin` file.

Cache advisories set using the `qioadmin` command are stored as extended attributes of the file in the inode. These settings persist across file system remounts and system reboots, but these attributes are not backed up by the usual backup methods, so they cannot be restored. Therefore, always be sure to reset cache advisories after each file restore. This is not necessary if you maintain the cache advisories for Quick I/O files in the `/etc/vx/qioadmin` file.

To enable or disable individual file settings for Cached Quick I/O automatically after a reboot or mount

- Add cache advisory entries in the `/etc/vx/qioadmin` file as follows:

```
 device=/dev/vx/dsk/<diskgroup>/<volume>

filename1,OFF

filename2,OFF

filename3,OFF

filename4,ON
```

  For example, to make the Cached Quick I/O settings for individual files in the `/db01` file system persistent, edit the `/etc/vx/qioadmin` file similar to the following:

```
 #
 # List of files to cache in /db01 file system
 #
 device=/dev/vx/dsk/PRODdg/db01

 cust.dbf,OFF
 system.dbf,OFF
 stk.dbf,ON
```

Determining individual file settings for Cached Quick I/O using qioadmin

You can determine whether Cached Quick I/O is enabled or disabled for individual files by displaying the file's cache advisory setting using the `qioadmin` command.

Note: To verify caching, always check the setting of the flag `qio_cache_enable` using `vxtunefs`, along with the individual cache advisories for each file.

To display the current cache advisory settings for a file

- Use the `qioadmin` command with the `-P` option as follows:

  ```
  $ /opt/VRTS/bin/qioadmin -P filename /mount_point
  ```

  For example, to display the current cache advisory setting for the file `cust.dbf` in the `/db01` file system:

  ```
  $ /opt/VRTS/bin/qioadmin -P cust.dbf /db01
  ```

  ```
  cust.dbf,OFF
  ```

Section 5

Using point-in-time copies

- Chapter 14. Understanding point-in-time copy methods
- Chapter 15. Considerations for Oracle point-in-time copies
- Chapter 16. Administering third-mirror break-off snapshots
- Chapter 17. Administering space-optimized snapshots
- Chapter 18. Administering Storage Checkpoints
- Chapter 19. Administering FileSnap snapshots
- Chapter 20. Backing up and restoring with Netbackup in an SFHA environment

Chapter 14

Understanding point-in-time copy methods

This chapter includes the following topics:

- About point-in-time copies
- When to use point-in-time copies
- About Storage Foundation point-in-time copy technologies
- Point-in-time copy solutions supported by SFDB tools
- About snapshot modes supported by Storage Foundation for Databases (SFDB) tools
- Volume-level snapshots
- About Reverse Resynchronization in volume-level snapshots (FlashSnap)
- Storage Checkpoints
- About FileSnaps

About point-in-time copies

Storage Foundation offers a flexible and efficient means of managing business-critical data. Storage Foundation lets you capture an online image of an actively changing database at a given instant, called a point-in-time copy.

More and more, the expectation is that the data must be continuously available (24x7) for transaction processing, decision making, intellectual property creation,

and so forth. Protecting the data from loss or destruction is also increasingly important. Formerly, data was taken out of service so that the data did not change while data backups occured; however, this option does not meet the need for minimal down time.

A point-in-time copy enables you to maximize the online availability of the data. You can perform system backup, upgrade, or perform other maintenance tasks on the point-in-time copies. The point-in-time copies can be processed on the same host as the active data, or a different host. If required, you can offload processing of the point-in-time copies onto another host to avoid contention for system resources on your production server. This method is called off-host processing. If implemented correctly, off-host processing solutions have almost no impact on the performance of the primary production system.

When to use point-in-time copies

The following typical activities are suitable for point-in-time copy solutions implemented using Veritas InfoScale FlashSnap:

- Data backup —Many enterprises require 24 x 7 data availability. They cannot afford the downtime involved in backing up critical data offline. By taking snapshots of your data, and backing up from these snapshots, your business-critical applications can continue to run without extended downtime or impacted performance.
- Providing data continuity —To provide continuity of service in the event of primary storage failure, you can use point-in-time copy solutions to recover application data. In the event of server failure, you can use point-in-time copy solutions in conjunction with the high availability cluster functionality of SFCFSHA or SFHA.
- Decision support analysis and reporting—Operations such as decision support analysis and business reporting may not require access to real-time information. You can direct such operations to use a replica database that you have created from snapshots, rather than allow them to compete for access to the primary database. When required, you can quickly resynchronize the database copy with the data in the primary database.
- Testing and training—Development or service groups can use snapshots as test data for new applications. Snapshot data provides developers, system testers and QA groups with a realistic basis for testing the robustness, integrity and performance of new applications.
- Database error recovery—Logic errors caused by an administrator or an application program can compromise the integrity of a database. You can recover a database more quickly by restoring the database files by using Storage

Checkpoints or a snapshot copy than by full restoration from tape or other backup media.

Use Storage Checkpoints to quickly roll back a database instance to an earlier point in time.

- Cloning data—You can clone your file system or application data. This functionality enable you to quickly and efficiently provision virtual desktops.

All of the snapshot solutions mentioned above are also available on the disaster recovery site, in conjunction with Volume Replicator.

For more information about snapshots with replication, see the *Veritas InfoScale 7.4.2 Replication Administrator's Guide*.

Storage Foundation provides several point-in-time copy solutions that support your needs, including the following use cases:

- Creating a replica database for decision support.
- Backing up and recovering a database with snapshots.
- Backing up and recovering an off-host cluster file system
- Backing up and recovering an online database.

About Storage Foundation point-in-time copy technologies

This topic introduces the point-in-time copy solutions that you can implement using the Veritas FlashSnap™ technology. Veritas FlashSnap technology requires a Veritas InfoScale Enterprise or Storage licenses.

Veritas InfoScale FlashSnap offers a flexible and efficient means of managing business critical data. It allows you to capture an online image of actively changing data at a given instant: a point-in-time copy. You can perform system backup, upgrade and other maintenance tasks on point-in-time copies while providing continuous availability of your critical data. If required, you can offload processing of the point-in-time copies onto another host to avoid contention for system resources on your production server.

The following kinds of point-in-time copy solution are supported by the FlashSnap license:

- Volume-level solutions. There are several types of volume-level snapshots. These features are suitable for solutions where separate storage is desirable to create the snapshot. For example, lower-tier storage. Some of these techniques provided exceptional offhost processing capabilities.

- File system-level solutions use the Storage Checkpoint feature of Veritas File System. Storage Checkpoints are suitable for implementing solutions where storage space is critical for:
  - File systems that contain a small number of mostly large files.
  - Application workloads that change a relatively small proportion of file system data blocks (for example, web server content and some databases).
  - Applications where multiple writable copies of a file system are required for testing or versioning.

  See "Storage Checkpoints" on page 169.
- File level snapshots.
  The FileSnap feature provides snapshots at the level of individual files.

Point-in-time copy solutions supported by SFDB tools

Storage Foundation for Databases (SFDB) tools provide a database-specific command line to create point-in-time copies of your Oracle database. SFDB tools use the underlying features of Storage Foundation to perform these operations.. For ease of use, the SFDB command line enables you to perform the point-in-time copy operations on the Oracle database with fewer steps. Also, the SFDB command line enables you to perform functions specific to Oracle databases.

Table 14-1 provides a comparison of the point-in-time copy solutions supported by SFDB tools.

Table 14-1 Comparison of Point-in-time Copy Solutions

	FlashSnap	Space-optimized snapshots	Database Storage Checkpoints	FileSnap
Underlying technology	Volume snapshots (third-mirror break-off snapshots)	Volume snapshots (space-optimized instant snapshots)	File system checkpoints	File-level snapshots (VxFS FileSnap)
Possibility of off-host processing	Yes	Yes (requires Cluster Volume Manager)	Yes (requires Cluster File System)	Yes (requires Cluster File System)
Additional storage requirements	Additional mirror plexes are required. Plexes are full copies of the original data.	Minimal (uses copy-on-write and requires cache volume)	Minimal (uses copy-on-write)	Minimal (uses shared extents)

Table 14-1 Comparison of Point-in-time Copy Solutions *(continued)*

	FlashSnap	Space-optimized snapshots	Database Storage Checkpoints	FileSnap
Performance impact after taking the point-in-time copy	None	Copy-on-write penalty	Copy-on-write penalty	Writes cause allocation.
Support for multiple clones from a single point-in-time copy	No. However, different mirrors with different snapshots can be used to create multiple clones.	No. However, multiple snapshots can be used to create multiple clones.	Yes	Yes
Supported snapshot modes	Online, Offline, Instant	Online, Offline, Instant	Online, Offline, Instant	Online
Support for replication environment (with Veritas Replicator)	No	Yes (Instant snapshot mode only)	No	No

About snapshot modes supported by Storage Foundation for Databases (SFDB) tools

The snapshot modes supported by SFDB tools are:

- Online
- Instant
- Offline

Certain snapshot modes are supported conditionally, as follows:

- FlashSnap, Storage Checkpoint, and SOS support all the modes.
- FileSnap supports the Online mode only.
- The Online mode is the default and recommended snapshot mode for all of the snapshot methods.

Table 14-2 Descriptions of Snapshot Modes

Snapshot mode	Description
Online	The online snapshot mode: ■ Is like an online or a hot backup of the application or the database and it is suitable as a backup image. ■ Requires the application to support online or hot backup mode. The Oracle database is put in the hot backup mode until the snapshot is taken. ■ Requires the database to be open. ■ Requires the database to be in the archivelog mode. **Note:** Most production databases are in the archivelog mode. ■ Has the least performance impact on the application or the database. ■ Allows restore of the primary application data from the snapshot image. ■ Allows the user to perform manual point-in-time recovery of a clone based on the snapshot or of the primary application after a restore operation.
Instant	The instant snapshot mode: ■ Is like a crash-consistent image of the application or the database and is not suitable as a backup image. ■ Freezes the application I/O so that a crash-consistent snapshot can be taken. Application I/O is frozen by freezing all the file systems holding the application files or data. Application I/O is frozen for the duration of the snapshot process. ■ Does not require the database to be in the archivelog mode. ■ Has some performance impact on the application or the database, since application I/O is frozen while the snapshot is being taken. However, the snapshot process itself may be faster, especially under heavy transactional load on the database. ■ Is not supported for standby databases. ■ Is supported for RAC databses.
Offline	The offline snapshot mode: ■ Is like a cold backup of the application or the database and it is suitable as a backup image. ■ Requires the application to be offline. ■ Requires the snapshot configuration to be validated when the application is online. ■ Is fastest amongst the snapshot modes. ■ Does not require the database to be in the archivelog mode. ■ Requires that database instance(s) should not be running, in case of Oracle.

Volume-level snapshots

A volume snapshot is an image of a Veritas Volume Manager (VxVM) volume at a given point in time. You can also take a snapshot of a volume set.

Volume snapshots allow you to make backup copies of your volumes online with minimal interruption to users. You can then use the backup copies to restore data that has been lost due to disk failure, software errors or human mistakes, or to create replica volumes for the purposes of report generation, application development, or testing.

Volume snapshots can also be used to implement off-host online backup.

Physically, a snapshot may be a full (complete bit-for-bit) copy of the data set, or it may contain only those elements of the data set that have been updated since snapshot creation. The latter are sometimes referred to as allocate-on-first-write snapshots, because space for data elements is added to the snapshot image only when the elements are updated (overwritten) for the first time in the original data set. Storage Foundation allocate-on-first-write snapshots are called space-optimized snapshots.

Persistent FastResync of volume snapshots

If persistent FastResync is enabled on a volume, VxVM uses a FastResync map to keep track of which blocks are updated in the volume and in the snapshot.

When snapshot volumes are reattached to their original volumes, persistent FastResync allows the snapshot data to be quickly refreshed and re-used. Persistent FastResync uses disk storage to ensure that FastResync maps survive both system and cluster crashes. If persistent FastResync is enabled on a volume in a private disk group, incremental resynchronization can take place even if the host is rebooted.

Persistent FastResync can track the association between volumes and their snapshot volumes after they are moved into different disk groups. After the disk groups are rejoined, persistent FastResync allows the snapshot plexes to be quickly resynchronized.

Data integrity in volume snapshots

A volume snapshot captures the data that exists in a volume at a given point in time. As such, VxVM does not have any knowledge of data that is cached in memory by the overlying file system, or by applications such as databases that have files open in the file system. Snapshots are always crash consistent, that is, the snapshot can be put to use by letting the application perform its recovery. This is similar to how the application recovery occurs after a server crash. If the `fsgen` volume usage type is set on a volume that contains a mounted Veritas File System (VxFS), VxVM

coordinates with VxFS to flush data that is in the cache to the volume. Therefore, these snapshots are always VxFS consistent and require no VxFS recovery while mounting.

For databases, a suitable mechanism must additionally be used to ensure the integrity of tablespace data when the volume snapshot is taken. The facility to temporarily suspend file system I/O is provided by most modern database software. The examples provided in this document illustrate how to perform this operation. For ordinary files in a file system, which may be open to a wide variety of different applications, there may be no way to ensure the complete integrity of the file data other than by shutting down the applications and temporarily unmounting the file system. In many cases, it may only be important to ensure the integrity of file data that is not in active use at the time that you take the snapshot. However, in all scenarios where application coordinate, snapshots are crash-recoverable.

Third-mirror break-off snapshots

A plex break-off snapshot uses an additional mirror to create the snapshot. Although you can create a plex break-off snapshot for a single plex volume, typically you take a snapshot of a mirrored volume. A mirrored volume has more than one plex or mirror, each of which is a copy of the data. The snapshot operation "breaks off" the plex, which becomes the snapshot volume. You can break off an existing plex or add a new plex specifically to serve as the snapshot mirror. Generally, you want to maintain redundancy for the original volume. If the original volume is a mirrored volume with two plexes, you add a third mirror for the snapshot. Hence, this type of snapshot is also known as a third-mirror snapshot.

The snapshot plex must be on a different disk from the existing plexes in the volume, within the same disk group. The disk must have enough disk space to contain the contents of the existing volume. If you have a one terabyte volume, you must have an additional one terabyte of disk space.

When you create the snapshot, the plexes are separated into two volumes. The original volume retains its original plex or plexes. The snapshot volume contains the snapshot plex. The original volume continues to take on I/O. The snapshot volume retains the data at the point of time when the snapshot was created, until you choose to perform processing on that volume.

You can make multiple snapshots, so you can have multiple copies of the original data.

Third-mirror break-off snapshots are suitable for write-intensive volumes (such as for database redo logs) where the copy-on-write mechanism of space-optimized or full-sized instant snapshots might degrade performance.

Space-optimized instant volume snapshots

Space-optimized snapshots do not contain complete physical images of the original data objects they represent. Space-optimized instant snapshots record changed regions in the original volume to a storage cache. As the original volume is written to, VxVM preserves its data in the cache before the write is committed. As the storage cache typically requires much less storage than the original volume, it is referred to as space-optimized. Space-optimized snapshots consume storage and I/O bandwidth in proportion to how much data on the original volume is updated during the life of the snapshot.

The benefits of space-optimized instant snapshots include immediate availability for use, quick refreshment, and easier configuration and administration. Because space-optimized snapshots consume less storage and I/O bandwidth than full-copy snapshots, you can take the snapshots much more frequently. This makes them well-suited for recovering from data corruption.

Space-optimized snapshots naturally tend to grow with age, as more of the data in the original objects changes, so they are inherently better-suited for shorter lifetimes.

Space-optimized snapshots cannot be taken off-host for auxiliary processing.

How space-optimized instant snapshots work

Space-optimized snapshots use a copy-on-write mechanism to make them immediately available for use when they are first created, or when their data is refreshed.

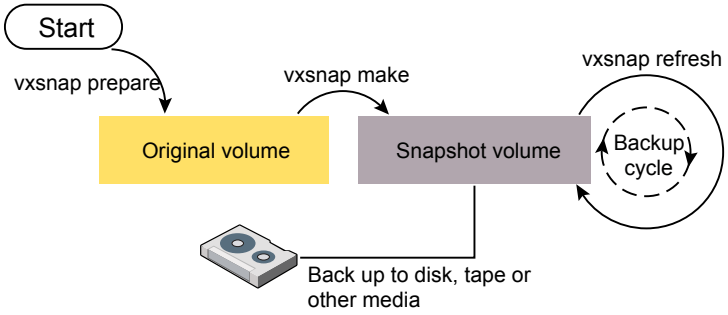
You can configure a single storage cache in a disk group that can be shared by all the volumes in that disk group. If so, the name of the cache that is declared must be the same for each volume's space-optimized snapshot. The cache is stored on disk and is persistent.

If the cache approaches full, configure VxVM to grow the cache automatically using any available free space in the disk group.

See "Creating a shared cache object" on page 208.

Figure 14-1 shows the instant space-optimized snapshot model.

Figure 14-1 Space-optimized instant snapshot creation and usage in a backup cycle



About Reverse Resynchronization in volume-level snapshots (FlashSnap)

FlashSnap service helps to create snapshots of Oracle database, which can be cloned and used for backup. When the primary Oracle database faces some kind of corruption, which renders the database non-functional, you can use volume snapshots to recover from such errors and return back to the point in time when the snapshots were created. This process of recovering a database from its volume snapshots created using FlashSnap service is referred to as Reverse Resynchronization or Reverse Resync.

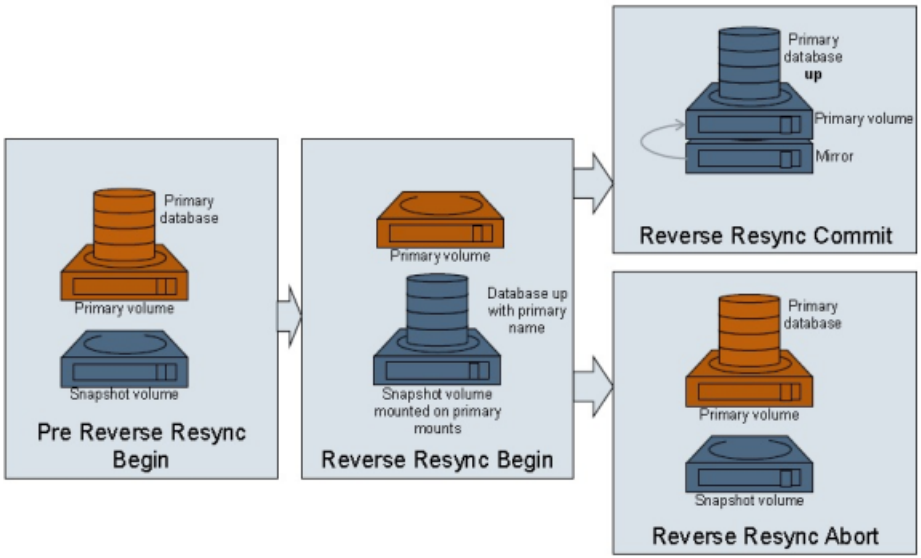
You can use the Storage Foundation Database Flashsnap feature to reverse resynchronize an online or offline point-in-time image of a database in an Oracle environment.

Reverse resync means restoring snapshot mirror volumes to primary data volumes. You can use reverse resync in cases where the primary data volumes are logically corrupted, and you want to replace the contents from the snapshot volumes.

Reverse resync operation is split into three distinct operations. The three operations are as follows:

- *Reverse Resync Begin*
- *Reverse Resync Abort*
- *Reverse Resync Commit*

Figure 14-2 Overview of Reverse Resynchronization



Types of Reverse Resync operations

Reverse resync provides the following three operations:

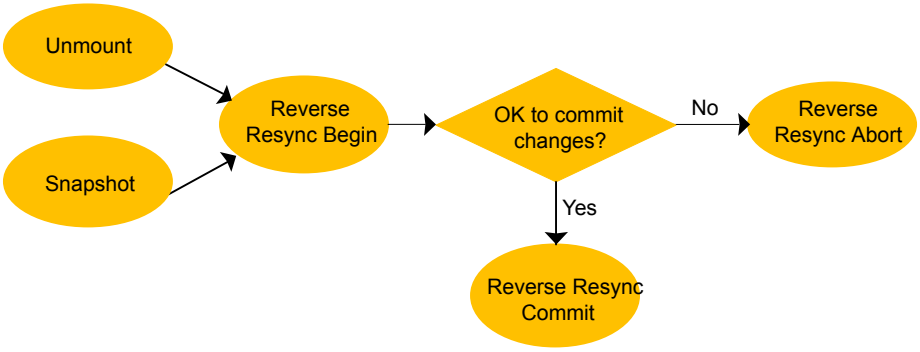
- **ReverseResyncBegin (alias: RRBegin, RRStart, ReverseResyncStart)**: You can use *RRBegin* operation to mount the snapshot volumes and the Oracle instance.

  It lets you preview the Oracle instance from the snapshot volumes.

  ReverseResync Begin operation supports three types of recovery methods: *manual*, *auto,* and *auto_until_scn*.
- **ReverseResyncAbort (alias: RRAbort, RRCancel, ReverseResyncCancel)**: You can use this operation to end reverse resync preview session and get back to the original data volumes.
- **ReverseResyncCommit (alias: RRCommit, RRFinish, ReverseResyncFinish)**: You can use this operation to commit the reverse resync changes that are previewed in *RRBegin* operation.

Note: After this operation finishes, you cannot go back to original data volumes. The snapshot volumes are permanently restored on to the data volumes. Storage checkpoints or the filesnaps that are created after the snapshots, which are used for reverse resync, are not available after the completion of RRCommit operation.

Figure 14-3 Reverse Resync operations and its working



See Figure 32-4 on page 398.

Limitations of Reverse Resync

Reverse Resync has the following limitations:

- The reverse resync is allowed from online and offline snapshots only. Instant snapshot mode cannot be used with Reverse Resync
- Not supported for RAC databases
- Not allowed if snapshot has been cloned
- If the archive log volume is excluded from snapshot operation, then the automatic recovery option is not available
- Reverse resynchronization of archive log volume cannot be performed for offline mode snapshots
- Reverse resynchronization is not supported in Cluster FileSystem (CFS) environment
- Reverse resynchronization is not supported for Data Guard configurations

- Reverse resynchronization is not supported for Veritas Volume Replicator (VVR) configurations

Pre-requisites for performing Reverse Resync operation

You have to ensure the following before performing Reverse Resync operation.

Reverse Resync in HA Configuration

To perform reverse resync in High Availability (HA) environment, the database service group must to be frozen and the `VxFSMountLock` attribute of mount resources must be set to zero.

For example,

- `# hares -modify Mount_data VxFSMountLock 0`
- `# hares -modify Mount_arch VxFSMountLock 0`
- `# hagrp -freeze <DB_service_grp>`

After reverse resync completion, you can unfreeze database service group and reset the attribute to the original value.

Recommendations for performing reverse resync

- Create a back up repository using `sfua_rept_util` tool before performing reverse resynchronization.
- Create a back up using FlashSnap snapshot commands if you make any structural changes to the database after you have taken a snapshot (e.g. adding a new datafile).
- Consider the database incarnation number, if you are planning to use RMAN backup for database recovery. The database incarnation number changes after RRcommit operation.

Changes introduced in the implementation of Reverse Resync from 5.X

Following are the changes introduced in the implementation of Reverse Resync:

- You can perform ReverseResyncBegin operation after ReverseResyncAbort operation

- You can control the database recovery in ReverseResyncBegin operation using the new parameters:

  `Reverse_Resync_Recovery`

  `Reverse_Resync_Archive_Log`

Configuration parameters for Reverse Resync operation

Table 14-3 lists the new parameters for reverse resync operation.

Table 14-3 New parameters for Reverse Resync operation

Parameter	Value	Description	Command line options
`REVERSE_RESYNC_RECOVERY`	Auto Manual Auto_Until_SCN	Specifies the method for recovering database during reverse resynchronization. The default is AUTO. Use AUTO recovery option to attempt automatic database recovery. Use Manual recovery option to mount the database during `-reverse-resync-begin` and `-reverse-resync-commit` steps. Use AUTO_UNTIL_SCN option to automatically recover the database until the point in time when snapshot was taken.	`--reverse_resync_recovery <value>` OR `--rr_recovery <value>`

Table 14-3 New parameters for Reverse Resync operation *(continued)*

Parameter	Value	Description	Command line options
REVERSE_RESYNC_ ARCHIVE_LOG	No Yes	Specifies whether the archivelog volume is to be reverse resynchronized. The default value is NO. When the value is set to NO, the archive log volume is excluded from reverse resync operation. When the value is set to YES, the archive log volume is restored from the corresponding snapshot volume.*	--reverse_resync_ archive_log <value> OR --rr_archive_log <value>

Note: *Reverse_Resync_Archive_Log = Yes option is useful when the archived logs are corrupted and need to be restored from backup.

Note: *Reverse_Resync_Archive_Log = No option lets you perform database recovery using the primary database archived logs for roll-forward recovery.

Storage Checkpoints

A Storage Checkpoint is a persistent image of a file system at a given instance in time. Storage Checkpoints use a copy-on-write technique to reduce I/O overhead by identifying and maintaining only those file system blocks that have changed since a previous Storage Checkpoint was taken. Storage Checkpoints have the following important features:

- Storage Checkpoints persist across system reboots and crashes.

- A Storage Checkpoint can preserve not only file system metadata and the directory hierarchy of the file system, but also user data as it existed when the Storage Checkpoint was taken.
- After creating a Storage Checkpoint of a mounted file system, you can continue to create, remove, and update files on the file system without affecting the image of the Storage Checkpoint.
- Unlike file system snapshots, Storage Checkpoints are writable.
- To minimize disk space usage, Storage Checkpoints use free space in the file system.

Storage Checkpoints and the Storage Rollback feature of Storage Foundation for Databases enable rapid recovery of databases from logical errors such as database corruption, missing files and dropped table spaces. You can mount successive Storage Checkpoints of a database to locate the error, and then roll back the database to a Storage Checkpoint before the problem occurred.

How Storage Checkpoints differ from snapshots

Storage Checkpoints differ from Veritas File System snapshots in the following ways because they:

- Allow write operations to the Storage Checkpoint itself.
- Persist after a system reboot or failure.
- Share the same pool of free space as the file system.
- Maintain a relationship with other Storage Checkpoints by identifying changed file blocks since the last Storage Checkpoint.
- Can have multiple, read-only Storage Checkpoints that reduce I/O operations and required storage space because the most recent Storage Checkpoint is the only one that accumulates updates from the primary file system.
- Can restore the file system to its state at the time that the Storage Checkpoint was taken.

Various backup and replication solutions can take advantage of Storage Checkpoints. The ability of Storage Checkpoints to track the file system blocks that have changed since the last Storage Checkpoint facilitates backup and replication applications that only need to retrieve the changed data. Storage Checkpoints significantly minimize data movement and may promote higher availability and data integrity by increasing the frequency of backup and replication solutions.

Storage Checkpoints can be taken in environments with a large number of files, such as file servers with millions of files, with little adverse impact on performance. Because the file system does not remain frozen during Storage Checkpoint creation,

applications can access the file system even while the Storage Checkpoint is taken. However, Storage Checkpoint creation may take several minutes to complete depending on the number of files in the file system.

How a Storage Checkpoint works

The Storage Checkpoint facility freezes the mounted file system (known as the primary fileset), initializes the Storage Checkpoint, and thaws the file system. Specifically, the file system is first brought to a stable state where all of its data is written to disk, and the freezing process momentarily blocks all I/O operations to the file system. A Storage Checkpoint is then created without any actual data; the Storage Checkpoint instead points to the block map of the primary fileset. The thawing process that follows restarts I/O operations to the file system.

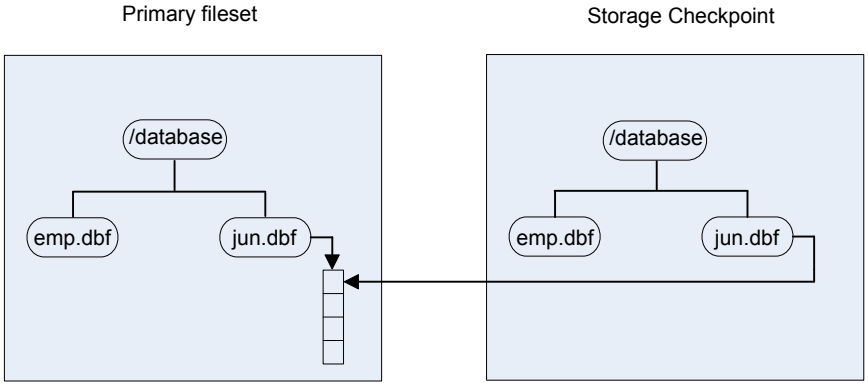
You can create a Storage Checkpoint on a single file system or a list of file systems. A Storage Checkpoint of multiple file systems simultaneously freezes the file systems, creates a Storage Checkpoint on all of the file systems, and thaws the file systems. As a result, the Storage Checkpoints for multiple file systems have the same creation timestamp. The Storage Checkpoint facility guarantees that multiple file system Storage Checkpoints are created on all or none of the specified file systems, unless there is a system crash while the operation is in progress.

Note: The calling application is responsible for cleaning up Storage Checkpoints after a system crash.

A Storage Checkpoint of the primary fileset initially contains only pointers to the existing data blocks in the primary fileset, and does not contain any allocated data blocks of its own.

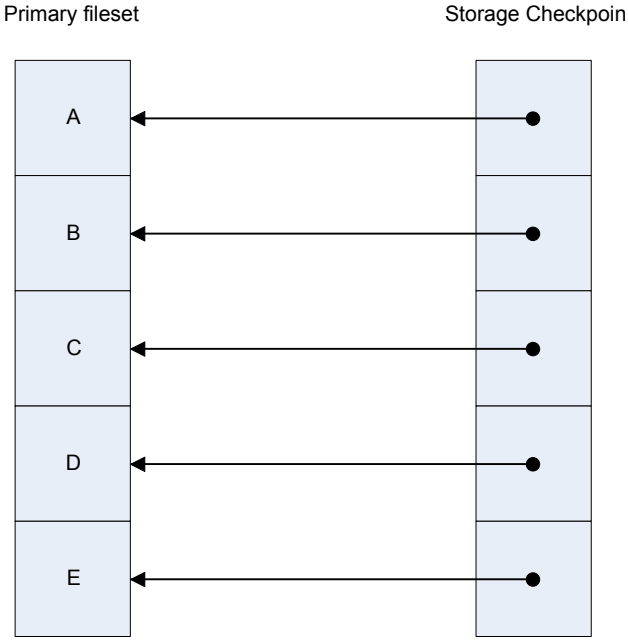
Figure 14-4 shows the file system `/database` and its Storage Checkpoint. The Storage Checkpoint is logically identical to the primary fileset when the Storage Checkpoint is created, but it does not contain any actual data blocks.

Figure 14-4 Primary fileset and its Storage Checkpoint



In Figure 14-5, a square represents each block of the file system. This figure shows a Storage Checkpoint containing pointers to the primary fileset at the time the Storage Checkpoint is taken, as in Figure 14-4.

Figure 14-5 Initializing a Storage Checkpoint



The Storage Checkpoint presents the exact image of the file system by finding the data from the primary fileset. VxFS updates a Storage Checkpoint by using the copy-on-write technique.

See "Copy-on-write" on page 173.

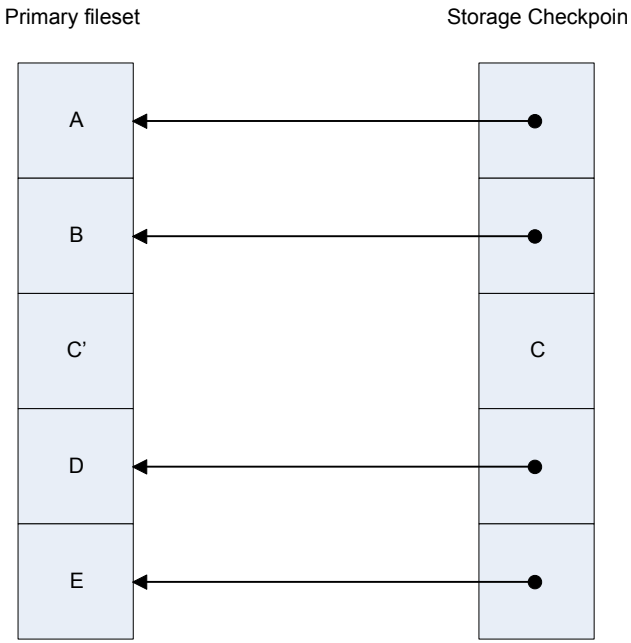
Copy-on-write

In Figure 14-6, the third data block in the primary fileset originally containing C is updated.

Before the data block is updated with new data, the original data is copied to the Storage Checkpoint. This is called the copy-on-write technique, which allows the Storage Checkpoint to preserve the image of the primary fileset when the Storage Checkpoint is taken.

Every update or write operation does not necessarily result in the process of copying data to the Storage Checkpoint because the old data needs to be saved only once. As blocks in the primary fileset continue to change, the Storage Checkpoint accumulates the original data blocks. In this example, subsequent updates to the third data block, now containing C', are not copied to the Storage Checkpoint because the original image of the block containing C is already saved.

Figure 14-6 Updates to the primary fileset



Storage Checkpoint visibility

With the `ckptautomnt` mount option, all Storage Checkpoints are made accessible automatically through a directory in the root directory of the file system that has the special name `.checkpoint`, which does not appear in directory listings. Inside this directory is a directory for each Storage Checkpoint in the file system. Each of these directories behave as a mount of the corresponding Storage Checkpoint, with the following exceptions:

- External applications, such as NFS, see the files as part of the original mount point. Thus, no additional NFS exports are necessary.
- Inode numbers exposed to applications can be made unique, depending on a mount option.

The Storage Checkpoints are automounted internally, but the operating system does not know about the automounting. This means that Storage Checkpoints cannot be mounted manually, and they do not apear in the list of mounted file systems. When Storage Checkpoints are created or deleted, entries in the Storage Checkpoint directory are automatically updated. If a Storage Checkpoint is removed with the `-f` option while a file in the Storage Checkpoint is still in use, the Storage Checkpoint is force unmounted, and all operations on the file fail with the EIO error.

If there is already a file or directory named `.checkpoint` in the root directory of the file system, such as a directory created with an older version of Veritas File System (VxFS) or when Storage Checkpoint visibility feature was disabled, the fake directory providing access to the Storage Checkpoints is not accessible. With this feature enabled, attempting to create a file or directory in the root directory with the name `.checkpoint` fails with the EEXIST error.

Note: If an auto-mounted Storage Checkpoint is in use by an NFS mount, removing the Storage Checkpoint might succeed even without the forced (`-f`) option.

Storage Checkpoints and 64-bit inode numbers

The inode number of a file is the same across Storage Checkpoints. For example, if the file `file1` exists in a file system and a Storage Checkpoint is taken of that file system, running the `stat` command on `file1` in the original file system and in the Storage Checkpoint returns the same value in `st_ino`. The combination of `st_ino` and `st_dev` should uniquely identify every file in a system. This is usually not a problem because Storage Checkpoints get mounted separately, so `st_dev` is different. When accessing files in a Storage Checkpoint through the Storage Checkpoint visibility extension, `st_dev` is the same for all Storage Checkpoints as well as for the original file system. This means files can no longer be identified uniquely by `st_ino` and `st_dev`.

In general, uniquely identifying all files in a system is not necessary. However, there can be some applications that rely on unique identification to function properly. For example, a backup application might check if a file is hard-linked to another file by calling `stat` on both and checking if `st_ino` and `st_dev` are the same. If a backup application were told to back up two clones through the Storage Checkpoint visibility extension at the same time, the application can erroneously deduce that two files are the same even though the files contain different data.

By default, Storage Foundation (SF) does not make inode numbers unique. However, you can specify the `uniqueino` mount option to enable the use of unique 64-bit inode numbers. You cannot change this option during a remount.

About Database Rollbacks using Storage Checkpoints

Each Storage Checkpoint is a consistent, point-in-time image of a file system, and Storage Rollback is the restore facility for these on-disk backups. Storage Rollback rolls back changed blocks contained in a Storage Checkpoint into the primary file system for faster database restoration.

Storage Checkpoints and Rollback process

A Storage Checkpoint is a disk and I/O efficient snapshot technology for creating a "clone" of a currently mounted file system (the primary file system). Like a snapshot file system, a Storage Checkpoint appears as an exact image of the snapped file system at the time the Storage Checkpoint was made. However, unlike a snapshot file system that uses separate disk space, all Storage Checkpoints share the same free space pool where the primary file system resides.

Note: A database Storage Checkpoint can be mounted as read only or read-write, allowing access to the files as if it were a regular file system.

Initially, a Storage Checkpoint contains no data. The Storage Checkpoint only contains the inode list and the block map of the primary fileset. This block map points to the actual data on the primary file system. Because only the inode list and block map are required and no data is copied, creating a Storage Checkpoint takes only a few seconds and very little space.

A Storage Checkpoint initially satisfies read requests by finding the data on the primary file system, using its block map copy, and returning the data to the requesting process. When a write operation changes a data block in the primary file system, the old data is first copied to the Storage Checkpoint, and then the primary file system is updated with the new data. The Storage Checkpoint maintains the exact view of the primary file system at the time the Storage Checkpoint was

taken. Subsequent writes to block n on the primary file system do not result in additional copies to the Storage Checkpoint because the old data only needs to be saved once. As data blocks are changed on the primary file system, the Storage Checkpoint gradually fills with the original data copied from the primary file system, and less and less of the block map in the Storage Checkpoint points back to blocks on the primary file system.

Database Storage Rollback restores a database, a tablespace, or datafiles on the primary file systems to the point-in-time image created during a Storage Checkpoint.

Database Storage Rollback is accomplished by copying the "before" images from the appropriate Storage Checkpoint back to the primary file system. As with Storage Checkpoints, Database Storage Rollback restores at the block level, rather than at the file level. Database Storage Rollback is executed using the `vxsfadm` command with the `-o restore` operation.

For example:

```
$ $ /opt/VRTSdbed/bin/vxsfadm -s checkpoint /
 -a oracle -o restore --checkpoint_name checkpoint1
```

Mountable Storage Checkpoints can be used for a wide range of application solutions including the following:

- Backups
- Investigations into data integrity
- Staging upgrades
- Database modifications
- Data replication solutions

If you mount a Storage Checkpoint as read-write, the command will not allow you to roll back to this Storage Checkpoint. This ensures that any Storage Checkpoint data that has been modified incorrectly cannot be a source of any database corruption. When a Storage Checkpoint is mounted as read-write, then a "shadow" Storage Checkpoint of the original Storage Checkpoint gets created, and this "shadow" Storage Checkpoint is mounted as read-write. This allows the database to still be rolled back to the original Storage Checkpoint.

Types of Storage Checkpoints

You can create the following types of Storage Checkpoints:

- Data Storage Checkpoints
- Removable Storage Checkpoints

- Non-mountable Storage Checkpoints

Data Storage Checkpoints

A data Storage Checkpoint is a complete image of the file system at the time the Storage Checkpoint is created. This type of Storage Checkpoint contains the file system metadata and file data blocks. You can mount, access, and write to a data Storage Checkpoint just as you would to a file system. Data Storage Checkpoints are useful for backup applications that require a consistent and stable image of an active file system. Data Storage Checkpoints introduce some overhead to the system and to the application performing the write operation. For best results, limit the life of data Storage Checkpoints to minimize the impact on system resources.

Removable Storage Checkpoints

A removable Storage Checkpoint can self-destruct under certain conditions when the file system runs out of space.

See "Storage Checkpoint space management considerations" on page 178.

During user operations such as `create` or `mkdir`, if the file system runs out of space, removable Storage Checkpoints are deleted, even if the Storage Checkpoints are mounted. This ensures that applications can continue without interruptions due to lack of disk space. Non-removable Storage Checkpoints are not automatically removed under such `ENOSPC` conditions. Veritas recommends that you create only removable Storage Checkpoints. However, during certain administrative operations, such as `fsadm`, even if the file system runs out of space, removable Storage Checkpoints are not deleted.

Storage Checkpoints are created as non-removable by default. The default behavior can be changed so that VxFS creates removable Storage Checkpoints by using the `vxtunefs -D ckpt_removable=1` command. With the default set to create removable Storage Checkpoints, non-removable Storage Checkpoints can be created using `fsckptadm -R create` *`ckpt_name mount_point`* command.

See the `vxtunefs`(1M) and `fsckptadm`(1M) manual pages.

Non-mountable Storage Checkpoints

Use non-mountable Storage Checkpoints as a security feature. This prevents other applications from accessing and modifying the Storage Checkpoint.

Storage Checkpoint space management considerations

Several operations, such as removing or overwriting a file, can fail when a file system containing Storage Checkpoints runs out of space. If the system cannot allocate sufficient space, the operation will fail.

Database applications usually preallocate storage for their files and may not expect a write operation to fail. During user operations such as `create` or `mkdir`, if the file system runs out of space, removable Storage Checkpoints are deleted. This ensures that applications can continue without interruptions due to lack of disk space. Non-removable Storage Checkpoints are not automatically removed under such `ENOSPC` conditions. Veritas recommends that you create only removable Storage Checkpoints. However, during certain administrative operations, such as using the `fsadm` command, using the `qiomkfile` command, and creating a Storage Checkpoint with the `fsckptadm` command, even if the file system runs out of space, removable Storage Checkpoints are not deleted.

When the kernel automatically removes the Storage Checkpoints, it applies the following policies:

- Remove as few Storage Checkpoints as possible to complete the operation.
- Never select a non-removable Storage Checkpoint.
- Select a nodata Storage Checkpoint only when data Storage Checkpoints no longer exist.
- Remove the oldest Storage Checkpoint first.
- Remove a Storage Checkpoint even if it is mounted. New operations on such a removed Storage Checkpoint fail with the appropriate error codes.
- If the oldest Storage Checkpoint is non-removable, then the oldest removable Storage Checkpoint is selected for removal. In such a case, data might be required to be pushed to a non-removable Storage Checkpoint, which might fail and result in the file system getting marked for a FULLFSCK. To prevent this occurrence, Veritas recommends that you only create removable Storage Checkpoints.

About FileSnaps

A FileSnap is an atomic space-optimized copy of a file in the same name space, stored in the same file system. Veritas File System (VxFS) supports snapshots on file system disk layout Version 8 and later.

FileSnaps provide an ability to snapshot objects that are smaller in granularity than a file system or a volume. The ability to snapshot parts of a file system name space is required for application-based or user-based management of data stored in a file

system. This is useful when a file system is shared by a set of users or applications or the data is classified into different levels of importance in the same file system.

All regular file operations are supported on the FileSnap, and VxFS does not distinguish the FileSnap in any way.

Properties of FileSnaps

FileSnaps provide non-root users the ability to snapshot data that they own, without requiring administrator privileges. This enables users and applications to version, backup, and restore their data by scheduling snapshots at appropriate points of their application cycle. Restoring from a FileSnap is as simple as specifying a snapshot as the source file and the original file as the destination file as the arguments for the `vxfilesnap` command.

FileSnap creation locks the source file as read-only and locks the destination file exclusively for the duration of the operation, thus creating the snapshots atomically. The rest of the files in the file system can be accessed with no I/O pause while FileSnap creation is in progress. Read access to the source file is also uninterrupted while the snapshot creation is in progress. This allows for true sharing of a file system by multiple users and applications in a non-intrusive fashion.

The name space relationship between source file and destination file is defined by the user-issued `vxfilesnap` command by specifying the destination file path. Veritas File System (VxFS) neither differentiates between the source file and the destination file, nor does it maintain any internal relationships between these two files. Once the snapshot is completed, the only shared property between the source file and destination file are the data blocks and block map shared by them.

The number of FileSnaps of a file is practically unlimited. The technical limit is the maximum number of files supported by the VxFS file system, which is one billion files per file set. When thousands of FileSnaps are created from the same file and each of these snapshot files is simultaneously read and written to by thousands of threads, FileSnaps scale very well due to the design that results in no contention of the shared blocks when unsharing happens due to an overwrite. The performance seen for the case of unsharing shared blocks due to an overwrite with FileSnaps is closer to that of an allocating write than that of a traditional copy-on-write.

In disk layout version 8 and later, to support block or extent sharing between the files, reference counts are tracked for each shared extent. VxFS processes reference count updates due to sharing and unsharing of extents in a delayed fashion. Also, an extent that is marked shared once will not go back to unshared until all the references are gone. This is to improve the FileSnap creation performance and performance of data extent unsharing. However, this in effect results in the shared block statistics for the file system to be only accurate to the point of the processing

of delayed reclamation. In other words, the shared extent statistics on the file system and a file could be stale, depending on the state of the file system.

Concurrent I/O to FileSnaps

FileSnaps design and implementation ensures that concurrent reads or writes to different snapshots of the same file perform as if these were independent files. Even though the extents are shared between snapshots of the same file, the sharing has no negative impact on concurrent I/O.

Copy-on-write and FileSnaps

Veritas File System (VxFS) supports an option to do lazy copy-on-write when a region of a file referred to by a shared extent is overwritten. A typical copy-on-write implementation involves reading the old data, allocating a new block, copying or writing the old data to the new block synchronously, and writing the new data to the new block. This results in a worst case possibility of one or more allocating transactions, followed by a read, followed by a synchronous write and another write that conforms to the I/O behavior requested for the overwrite. This sequence makes typical copy-on-write a costly operation. The VxFS lazy copy-on-write implementation does not copy the old data to the newly allocated block and hence does not have to read the old data either, as long as the new data covers the entire block. This behavior combined with delayed processing of shared extent accounting makes the lazy copy-on-write complete in times comparable to that of an allocating write. However, in the event of a server crash, when the server has not flushed the new data to the newly allocated blocks, the data seen on the overwritten region would be similar to what you would find in the case of an allocating write where the server has crashed before the data is flushed. This is not the default behavior and with the default behavior the data that you find in the overwritten region will be either the new data or the old data.

Reading from FileSnaps

For regular read requests, Veritas File System (VxFS) only caches a single copy of a data page in the page cache for a given shared data block, even though the shared data block could be accessed from any of the FileSnaps or the source file. Once the shared data page is cached, any subsequent requests via any of the FileSnaps or the source file is serviced from the page cache. This eliminates duplicate read requests to the disk, which results in lower I/O load on the array. This also reduces the page cache duplication, which results in efficient usage of system page cache with very little cache churning when thousands of FileSnaps are accessed.

Block map fragmentation and FileSnaps

The block map of the source file is shared by the snapshot file. When data is overwritten on a previously shared region, the block map of the file to which the write happens gets changed. In cases where the shared data extent of a source file is larger than the size of the overwrite request to the same region, the block map of the file that is written to becomes more fragmented.

Backup and FileSnaps

A full backup of a VxFS file system that has shared blocks may require as much space in the target as the number of total logical references to the physical blocks in the source file system. For example, if you have a 20 GB file from which one thousand FileSnaps were created, the total number of logical block references is approximately 20 TB. While the VxFS file system only requires a little over 20 GB of physical blocks to store the file and the file's one thousand snapshots, the file system requires over 20 TB of space on the backup target to back up the file system, assuming the backup target does not have deduplication support.

Chapter 15

Considerations for Oracle point-in-time copies

This chapter includes the following topics:

- Considerations for database layouts
- Supported Oracle configurations
- Considerations for multitenant database support
- Behavior of clone databases

Considerations for database layouts

The following considerations for database layouts apply if you are using Storage Foundation for Databases (SFDB) tools:

- All database files must be on VxFS file systems. These include all copies of control files, data files, temp files, and all copies of online redo log files. All the underlying volumes must be VxVM volumes.
- At least one archive log destination must be on VxFS on a VxVM volume.
- For the online snapshot mode, the archive log destination must not contain any control files, data files, temp files, or online redo log files.
- The database PFILE or SPFILE can reside anywhere. If your database is using a pfile and the pfile resides in a non-default location (other than `$ORACLE_HOME/dbs`), then you must specify the pfile location explicitly using the PFILE parameter.
- The Oracle home directory can reside anywhere.
- Databases using OMF naming conventions are fully supported.

- Databases on Oracle ASM storage are not supported.
- The `vxsfadm` command must be enabled to use the operating system authentication to access the Oracle Database (/ as sysdba).
- When performing `vxsfadm` operations, certain operations may fail, and when `vxsfadm` attempts to recover from this failure the recovery may also fail. Due to this, the configuration would be in an error state, and some of the application changes would not be reverted. In this scenario, you may need to revert the changes manually followed by an error recovery.
  For example, when creating a checkpoint for an Oracle database, the database is put into the backup mode, and then a checkpoint is created following which the database is removed from the backup mode. If the checkpoint creation fails, and the recovery operation is unsuccessful, then the database would remain in the backup mode.

Supported Oracle configurations

For information on supported Oracle versions, refer to the database support matrix at:

https://www.veritas.com/bin/support/docRepoServlet?bookId=112632971-112632974-1&requestType=pdf

SFDB tools fully support Oracle single instance and Oracle RAC databases except for the following restrictions

- If you are using Storage Foundation for Databases (SFDB) tools with physical standby databases, the following restrictions apply:
  - Oracle RAC standby databases are not supported.
  - Instant snapshots are not supported. Only offline and online snapshots are supported for standby databases.
  - The standby database must be open either in the read-only mode (active standby) or it must be activated in the read-write mode.

    Note: You might need an active Oracle Data Guard license for this requirement.

  - If the standby database is opened in the read-only mode, then redo apply services must be stopped temporarily before SFDB tools can be used to perform operations, such as cloning or taking a snapshot, on a physical standby database.

- Logical standby databases are not supported. Snapshot or cloning operations cannot be performed on a database functioning as a logical standby.
- Oracle databases using Oracle ASM storage are not supported.
- Oracle RAC databases are not supported for taking space-optimized snapshots on a Veritas Volume Replicator (VVR) secondary site.
  Oracle disaster recovery configuration with VVR is supported. You can use SFDB tools to take space-optimized snapshots on the secondary site also.
- SFDB tools do not support taking point-in-time copies while structural changes to the database are in progress, such as adding or dropping tablespaces and adding or dropping data files. However, once a point-in-time copy is taken, a clone can be created at any time, regardless of the status of the database.
- SFDB tools support `vxsfadm` operations (point-in-time copy) on supported versions of the Oracle multitenant databases. The SFDB tools do not support operations on individual Pluggable Databases (PDB).

Considerations for multitenant database support

For the multitenant databases, the following considerations are applicable:

- SFDB tools do not support offline tablespaces within PDBs.
- Reverse resync operations are not allowed if some PDBs are not in the open state during a point-in-time copy operation.
- Checkpoint rollback with the **tbslist** option rolls back all tablespaces in a CDB and/or PDB with the specified tbsname.
- A point-in-time copy of PDBs can be captured only in the open state. When the PDBs are cloned, they are visible in the cloned CDB but operations on these PDBs fail.
- CDB cloning for instant snapshot are not supported when any of the PDBs are not in the open state.
- Cloning operation fails if some of the PDBs are opened in the read-only mode. You may not observe this issue in Oracle Database version 12.1.0.2.
- Cloning operation fails if some of the tablespaces are in the read-only mode. You may not observe this issue in Oracle Database version 12.1.0.2.
- The cloning operation may fail if performed after a reverse resync operation.
- The operations that are performed on the CDB may fail when any of its PDBs are opened in the restricted mode.

Behavior of clone databases

For clone databases, the following considerations apply for tablespaces and data files.

- In a clone database, if some data files are offline, but their associated tablespaces are online, the data files normally appear in the RECOVER status. Therefore, an additional, manual recovery might be required for such files to bring them online in the clone database.
- If some tablespaces or data files are under recovery, the snapshot process fails.
- Tablespaces that are offline are retained offline in a clone database.
- Tablespaces that are read-only are retained read-only in a clone database.
- Tablespaces that are online are retained online in a clone database.

Chapter 16

Administering third-mirror break-off snapshots

This chapter includes the following topics:

- Database FlashSnap for cloning
- Preparing hosts and storage for Database FlashSnap
- Creating a clone of a database by using Database FlashSnap
- Resynchronizing mirror volumes with primary volumes
- Recovering the clone database manually
- Cloning a database on the secondary host

Database FlashSnap for cloning

Veritas Database FlashSnap helps to create a point-in-time copy of a database for backup and off-host processing. Database FlashSnap lets you make backup copies of your volumes online and with minimal interruption to users.

Database FlashSnap lets you capture an online image of an actively changing database at a given instant that is known as a snapshot. A snapshot copy of the database is referred to as a database snapshot. You can use a database snapshot on the same host as the production database or on a secondary host sharing the same storage. A database snapshot can be used for off-host processing applications, such as backup, data warehousing, and decision-support queries. When the snapshot is no longer needed, the database administrator can import the original snapshot back to the primary host and resynchronize the snapshot to the original database volumes. Database FlashSnap commands are executed from the command line interface.

Database FlashSnap advantages

Database FlashSnap provides the following advantages:

- The database snapshot can be used on the same host as the production database or on a secondary host sharing the same storage.
- In many companies, there is a clear separation between the roles of system administrators and database administrators. Creating database snapshots typically requires superuser (root) privileges, the privileges that database administrators do not usually have. Because superuser privileges are not required, Database FlashSnap overcomes these obstacles by enabling database administrators to easily create consistent snapshots of the database.

Preparing hosts and storage for Database FlashSnap

Review the following details to prepare the hosts and storage for Database FlashSnap.

Setting up hosts

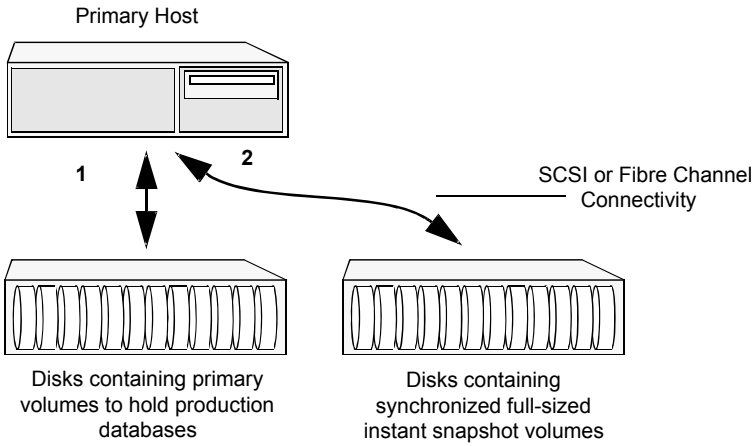
Database FlashSnap requires sufficient disk space in the disk group to add a mirror of equal size of the existing database.

Setting up a storage configuration for Database FlashSnap operations is a system administrator's responsibility and requires superuser (root) privileges. Database FlashSnap utilities do not address setting up an appropriate storage configuration.

Same-node configuration

Figure 16-1 shows the suggested arrangement for implementing Database FlashSnap solutions on the primary host to avoid disk contention.

Figure 16-1 Example of a Database FlashSnap solution on a primary host

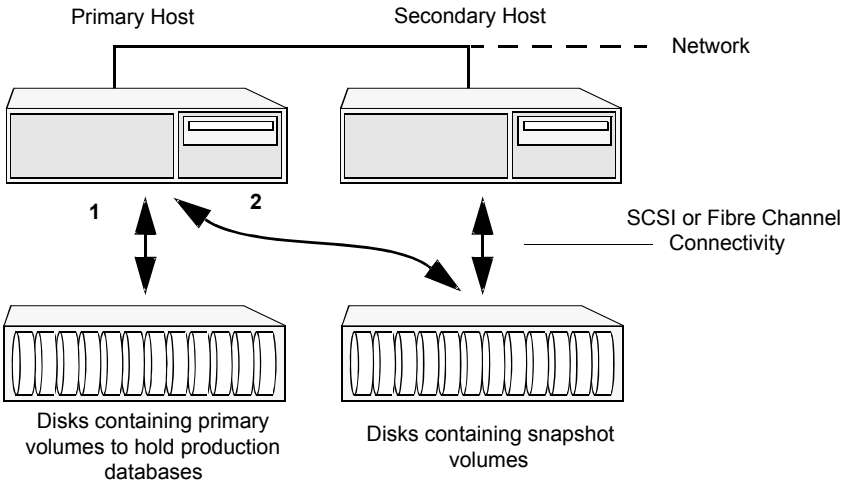


Database FlashSnap off-host configuration

A Database FlashSnap off-host configuration allows CPU- and I/O-intensive operations to be performed for online backup and decision support without degrading the performance of the primary host running the production database. Both the primary and secondary hosts share the storage in which the snapshot database is created. Both the primary and secondary hosts have access to the disks containing the snapshot volumes.

Figure 16-2 shows a Database FlashSnap off-host configuration.

Figure 16-2 Example of an off-host Database FlashSnap solution



Note: If you plan to use the FlashSnap feature in a VVR environment, then perform the FlashSnap setup prerequisite instructions after configuring VVR. In case you already have VVR configured, then make sure that the storage for mirror or snapshot volumes is deportable.

For information on host and storage requirements for an off-host configuration:

See "Requirements for an off-host database configuration" on page 55.

Creating a snapshot mirror of a volume or volume set used by the database

With Database FlashSnap, you can mirror the volumes used by the database to a separate set of disks, and those mirrors can be used to create a snapshot of the database. These snapshot volumes can be split and placed in a separate disk group. This snapshot disk group can be imported on a separate host, which shares the same storage with the primary host. The snapshot volumes can be resynchronized periodically with the primary volumes to get recent changes of the datafiles. If the primary datafiles become corrupted, you can quickly restore them from the snapshot volumes. Snapshot volumes can be used for a variety of purposes, including backup and recovery, and creating a clone database.

You must create snapshot mirrors for all of the volumes used by the database datafiles before you can create a snapshot of the database. This section describes the procedure used to create snapshot mirrors of volumes.

Use the `vxsnap` command to create a snapshot mirror or synchronize a snapshot mirror.

Prerequisites

- You must be logged in as superuser (root).
- The disk group must be version 110 or later.
  For more information on disk group versions, see the vxdg(1M) online manual page.
- Be sure that a data change object (DCO) and a DCO log volume are associated with the volume for which you are creating the snapshot.
- Persistent FastResync must be enabled on the existing database volumes and disks must be assigned for the snapshot volumes.
  FastResync optimizes mirror resynchronization by tracking updates to stored data that have been missed by a mirror. When a snapshot mirror is reattached to its primary volumes, only the updates that were missed need to be re-applied to resynchronize it. FastResync increases the efficiency of the volume snapshot mechanism to better support operations such as backup and decision support.
  For detailed information about FastResync, see the *Storage Foundation Administrator's Guide*.
- Snapshot mirrors and their associated DCO logs should be on different disks than the original mirror plexes, and should be configured correctly for creating snapshots by the system administrator.
- When creating a snapshot mirror, create the snapshot on a separate controller and separate disks from the primary volume.
- Allocate separate volumes for archive logs.
- Do not place any datafiles, including control files, in the $ORACLE_HOME/dbs directory.

Usage Notes

- Create a separate disk group for Oracle database-related files.
- Do not share volumes between Oracle database files and other software.
- ORACLE_HOME cannot be included in the snapshot mirror.
- Resynchronization speed varies based on the amount of data changed in both the primary and snapshot volumes during the break-off time.
- Do not share any disks between the original mirror and the snapshot mirror.
- Snapshot mirrors for datafiles and archive logs should be created so that they do not share any disks with the data of the original volumes. If they are not created in this way, the VxVM disk group cannot be split and, as a result, Database FlashSnap will not work.

Note: Database FlashSnap commands support third-mirror break-off snapshots only. The snapshot mirror must be in the SNAPDONE state.

The following sample procedure is for existing volumes without existing snapshot plexes or associated snapshot volumes. In this procedure, *volume_name* is the name of either a volume or a volume set.

Note: You must be logged in as superuser (root) to issue the commands in the following procedure.

To create a snapshot mirror of a volume or volume set

1. To prepare the volume for being snapshot, use the `vxsnap prepare` command:

   ```
   # vxsnap -g diskgroup prepare volume \
   alloc="storage_attribute ..."
   ```

   The `vxsnap prepare` command automatically creates a DCO and DCO volumes and associates them with the volume, and enables Persistent FastResync on the volume. Persistent FastResync is also set automatically on any snapshots that are generated from a volume on which this feature is enabled.

   For enabling persistent FastResync on a volume either from the command line or from within a script, use the `vxsnap prepare` command as described above.

2. To verify that FastResync is enabled on the volume, use the `vxprint` command:

   ```
   # vxprint -g diskgroup -F%fastresync volume_name
   ```

   This returns on if FastResync is on. Otherwise, it returns off.

3 To verify that a DCO and DCO log volume are attached to the volume, use the `vxprint` command:

```
# vxprint -g diskgroup -F%hasdcolog volume_name
```

This returns on if a DCO and DCO log volume are attached to the volume. Otherwise, it returns off.

4 Create a mirror of a volume:

```
# vxsnap -g diskgroup addmir volume_name alloc=diskname
```

Example of creating 3 mirrors for a particular volume:

```
# vxsnap -g diskgroup addmir datavol \
nmirror=3 alloc=disk1,disk2,disk3
```

5 List the available mirrors:

```
# vxprint -g diskgroup -F%name -e"pl_v_name in \"volume_name\""
```

6 Enable database FlashSnap to locate the correct mirror plexes when creating snapshots:

- Set the dbed_flashsnap tag for the data plex you want to use for breaking off the mirror. You can choose any tag name you like, but it needs to match the `SNAPSHOT_PLEX_TAG` attribute specified in the configuration or snapplan.

  ```
  # vxedit -g diskgroup set putil2=dbed_flashsnap plex_name
  ```

- Verify that the dbed_flashsnap tag has been set to the desired data plex:

  ```
  # vxprint -g diskgroup -F%name -e"pl_v_name in \
  \"volume_name\" && p2 in \"dbed_flashsnap\""
  ```

If you require a backup of the data in the snapshot, use an appropriate utility or operating system command to copy the contents of the snapshot to tape or to some other backup medium.

Creating a clone of a database by using Database FlashSnap

You can use Database FlashSnap to create a clone of a database by performing the steps outlined in Figure 16-3.

Figure 16-3 Creating a Clone - Workflow

See "vxsfadm command reference" on page 323.

See "FlashSnap configuration parameters" on page 327.

See "FlashSnap supported operations" on page 329.

For details, refer to `vxsfadm-flashsnap`(1M) and `vxsfadm-oracle-flashsnap`(1M) man pages.

To create a clone of an Oracle database by using FlashSnap

1 Create a configuration file.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap \
-o setdefaults -a oracle --oracle_sid sfaedb \
--oracle_home /ora_base/db_home -c TEST1
Written config file TEST1
```

This command creates a default configuration file at the location mentioned by the argument given to the **-c** option. If a path is not specified, then the configuration file is created in the present working directory with all the parameters and default values. You can change the parameters, if required.

Note: If a configuration file with the specified name already exists, then specify a unique configuration name, different path, or remove the existing configuration file. Also, the delete operation does not delete this configuration file.

---

2 Validate the setup.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap \
-o validate -a oracle --oracle_sid sfaedb \
--oracle_home /ora_base/db_home --flashsnap_name \
TEST1 --app_mode online
Validating database configuration for third-mirror-break-off snapshot:
ORACLE_SID = sfaedb
ORACLE_HOME = /ora_base/db_home
APP_MODE = online
SNAPSHOT_ARCHIVE_LOG = yes
REVERSE_RESYNC_ARCHIVE_LOG = no
REVERSE_RESYNC_RECOVERY = auto
ARCHIVELOG_DEST = /arch/sfaedb
Database validation successful.
Validating database volume layout for third-mirror-break-off snapshot:
Data volumes ready for snapshot:
Volume/volume-set datavol of diskgroup datadg mounted on /data.
Archivelog volume ready for snapshot:
Volume/volume-set archvol of diskgroup datadg mounted on /arch.
Storage units to be used for snapshot from diskgroup datadg:
vmdk0_25 vmdk0_27
SNAPSHOT_VOL_PREFIX = SNAP_
SNAPSHOT_DG_PREFIX = SNAP_
Database volume layout validated successfully.
```

This command validates the configuration file and the database environment. In case of any problem, appropriate error messages are displayed that you can use to correct the problem and then retry.

3 Create a snapshot of the database.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap \
-o snap  -a oracle --oracle_sid sfaedb --oracle_home \
/ora_base/db_home --flashsnap_name TEST1 \
--app_mode online
Validating database configuration for third-mirror-break-off snapshot:
ORACLE_SID = sfaedb
ORACLE_HOME = /ora_base/db_home
APP_MODE = online
ARCHIVELOG_DEST = /arch/sfaedb
SNAPSHOT_ARCHIVE_LOG = yes
REVERSE_RESYNC_ARCHIVE_LOG = no
REVERSE_RESYNC_RECOVERY = auto
Database validation successful.
snapshot started at Mon Aug 26 10:33:35 2013.
Putting database in backup mode ... Done
Validating database volume layout for third-mirror-break-off snapshot:
Data volumes ready for snapshot:
Volume/volume-set datavol of diskgroup datadg mounted on /data.
Archivelog volume ready for snapshot:
Volume/volume-set archvol of diskgroup datadg mounted on /arch.
Storage units to be used for snapshot from diskgroup datadg:
emc0_7 emc0_5
SNAPSHOT_VOL_PREFIX = SNAP_
SNAPSHOT_DG_PREFIX = SNAP_
Database volume layout validated successfully.
Creating snapshot volumes for data volumes ... Done
Taking database out of backup mode ... Done
Flushing online redo logs ... Done
Creating snapshot volume for archivelog volume ... Done
Copying snapshot information to snapshot volume ... Done
Creating snapshot diskgroups ... Done
Deporting snapshot diskgroups ... Done
SNAP_datadg
snapshot ended at Tue Aug 27 00:33:54 2013.
```

This command breaks the user-specified mirror (parameter `SNAPSHOT_PLEX_TAG`) from the primary volumes and creates a new disk group with the name starting with the string defined in the `snap_dg_prefix` parameter. The default value of this parameter is SNAP_.

When you create a snapshot, the broken-off mirrors are split off into a different disk group that is referred to as the snapshot disk group. The mirrors are the

only components that are present in the snapshot disk group. If additional objects, such as cache objects, must be moved from the original disk group to the snapshot disk group, you can use the EXTRA_OBJECTS parameter to specify these additional objects.

See "FlashSnap configuration parameters" on page 327.

Note: At the validation stage, all the parameters, including the mandatory parameters `--oracle_sid` and `--flashsnap_name`, are read and stored in the repository.

If you need to change any parameter, change the configuration file and specify it with the `-c` option.

4 Mount the snapshot.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap \
-o mount -a oracle --oracle_sid sfaedb \
--oracle_home /ora_base/db_home --flashsnap_name TEST1
Retrieving snapshot information ...                          Done
Importing snapshot diskgroups ...                            Done
Mounting snapshot volumes ...                                Done
```

Note: This command mounts the snapshot on the host to which storage is attached. The secondary host is the system defined in the SECONDARY_HOST parameter of the configuration file.

By default, volumes are mounted under the `/var/tmp` file system.

If you need to specify an alternate location for mounting snapshot volumes, either provide CLONE_PATH on the command line or from the configuration file.

For performing off-host operations, specify the SFDB repository host using the `-r` option of the `vxsfadm` command.

```
$ vxsfadm -s flashsnap -a oracle -o mount \
--oracle_sid sfaedb --flashsnap_name TEST1 --clone_path /cloneoracle
```

Note: Ensure that the Oracle user has the required permissions to create the `/cloneoracle` directory, if it does not exist.

---

5 Clone the database based on the snapshot.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap -o clone \
-a oracle --oracle_sid sfaedb --oracle_home /ora_base/db_home \
--flashsnap_name TEST1 --clone_name clone1
Retrieving snapshot information ... Done
Importing snapshot diskgroups ... Done
Mounting snapshot volumes ... Done
Preparing parameter file for clone database ... Done
Mounting clone database clone1 ... Done
Starting database recovery ...
Clone database clone1 is open.
```

If you have not specified *clone_name*, it is automatically generated.

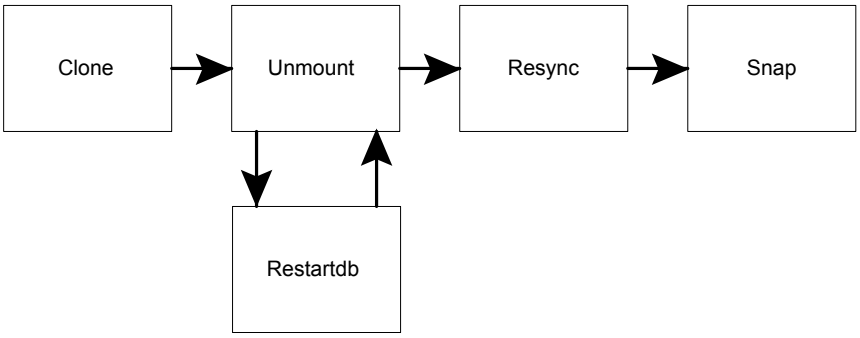
Note: If you have already specified the *clone_name* and the *clone_path* parameters in the configuration file that was used during the validate operation, the *clone_name* parameter is not required on the command line.

If you need to change a memory tuning parameter of the database clone such as memory_target , you can specify a modified pfile during the clone operation.

Resynchronizing mirror volumes with primary volumes

After creating a clone of your database, you can refresh mirror volumes with primary volumes by using the steps outlined in Figure 16-4.

Figure 16-4 Resynchronizing Mirror Volumes



To resynchronize mirror volumes with primary volumes

1 Unmount the clone database.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap -o umount \
 -a oracle --oracle_sid sfaedb --oracle_home /ora_base/db_home \
--flashsnap_name TEST1 --clone_name clone1
Shutting down clone database clone1 ...
Database shut down.
Retrieving snapshot information ...                          Done
Unmounting snapshot volumes ...                              Done
Deporting snapshot diskgroups ...                            Done
```

This command stops the clone database gracefully and unmounts the file systems. The unmounted database can be restarted by using the clone operation.

You can use the `-o clone` option to restart a clone database after it is unmounted.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap -o clone \
-a oracle --oracle_sid sfaedb --oracle_home /ora_base/db_home \
--flashsnap_name TEST1 --clone_name clone1
Retrieving snapshot information ...                          Done
Importing snapshot diskgroups ...                            Done
Mounting snapshot volumes ...                                Done
Starting clone database clone1 ...
Clone database started.
```

This command mounts the snapshot file systems and restarts the cloned database.

2 Refresh mirror volumes with primary volumes.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap -o resync \
-a oracle --oracle_sid sfaedb --oracle_home /ora_base/db_home \
--flashsnap_name TEST1
resync started at Sun Sep 18 14:53:17 2011.
Importing snapshot diskgroups ...                          Done
Joining snapshot diskgroups to original diskgroups ...     Done
Reattaching snapshot volumes to original volumes ...       Done
resync ended at Sun Sep 18 14:53:30 2011.
```

This command resynchronizes all mirror volumes that were broken during snapshot state with primary volumes. After the resync operation is complete and mirror volumes are in the SNAPDONE state, you can take fresh snapshots.

You can use the `vxprint` command to view the state of the volumes.

To reverse resynchronize the snapshot volume

- Use `vxsfadm -o rrbegin` to start the reverse resynchronization operation
- Use `vxsfadm -o rrcommit` to commit the reverse resynchronization changes
- Use `vxsfadm -o rrabort` to abort or cancel the reverse resync operation and to go back to the original data volumes

---

Note: If a Storage Checkpoint, FlashSnap, FileSnap, or SOS object is deleted after a snapshot operation, using `vxsfadm` and followed by a reverse resync operation, then SFDB does not list the deleted object. This behavior occurs because the object entries are deleted from the SFAE repository.

See Figure 32-4 on page 398.

Example outputs for reverse resync parameters

```
[oracle@testbox ~]$ vxsfadm -a oracle -s flashsnap --name sfae -o snapshot
Validating database configuration for third-mirror-break-off snapshot:
    ORACLE_SID                  = RR
    ORACLE_HOME                 = /home/oracle/app/oracle/product/11.2.0/
dbhome_11203
    APP_MODE                    = online
    ARCHIVELOG_DEST             = /arch
    SNAPSHOT_ARCHIVE_LOG        = yes
    REVERSE_RESYNC_ARCHIVE_LOG  = no
```

```
    REVERSE_RESYNC_RECOVERY     = auto
Database validation successful.
snapshot started at Mon Dec 17 13:27:46 2012.
Putting database in backup mode ...                            Done
Validating database volume layout for third-mirror-break-off snapshot:
    Data volumes ready for snapshot:
        Volume/volume-set datavol of diskgroup datadg mounted on /data.
    Archivelog volume ready for snapshot:
        Volume/volume-set archvol of diskgroup datadg mounted on /arch.
    Storage units to be used for snapshot from diskgroup datadg:
        sdl sdk sdm sdj
    SNAPSHOT_VOL_PREFIX         = SNAP_
    SNAPSHOT_DG_PREFIX          = SNAP_
Database volume layout validated successfully.
Creating snapshot volumes for data volumes ...                 Done
Taking database out of backup mode ...                         Done
Flushing online redo logs ...                                  Done
Creating snapshot volume for archivelog volume ...             Done
Copying snapshot information to snapshot volume ...            Done
Creating snapshot diskgroups ...                               Done
Deporting snapshot diskgroups ...                              Done
SNAP_datadg
snaphot ended at Mon Dec 17 13:28:00 2012.

[oracle@testbox ~]$ vxsfadm -a oracle -s flashsnap --name sfae -o rrbegin
Validating reverse resync volumes ... Done
Importing snapshot diskgroups ... Done
Unmounting primary data volumes ... Done
Mounting snapshot volumes ... Done
Creating storage checkpoint ... Done
Starting database recovery ...
Database started.
SFDB vxsfadm WARNING V-81-0951 Note that this is only a temporary session
for previewing Reverse Resync changes. Reverse Resync operation will be
complete after performing Reverse Resync Commit operation. Alternatively,
you can use Reverse Resync Abort operation to discard the changes and
go back to previous state.

[oracle@testbox ~]$ vxsfadm -a oracle -s flashsnap --name sfae -o rrabort
Shutting down database RR ...
Database shut down.
Unmounting snapshot volumes ...                                Done
Restoring storage checkpoint ...                               Done
```

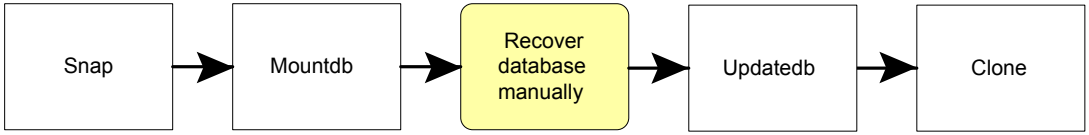
```
Mounting primary data volumes ...                                Done
Deporting snapshot diskgroups ...                                Done

[oracle@testbox ~]$ vxsfadm -a oracle -s flashsnap --name sfae -o rrcommit
Shutting down database RR ...
Database shut down.
Unmounting snapshot volumes ...                                  Done
Joining snapshot diskgroups to original diskgroups ...           Done
Restoring snapshot volumes onto data volumes ...                 Done
Mounting primary data volumes ...                                Done
Removing storage checkpoint ...                                  Done
Starting database RR ...
Database started.
```

Recovering the clone database manually

For Oracle databases, you can perform some special operations such as recovering the database to a System Change Number (SCN) or a particular time. You can use the mountdb operation to perform the database recovery. This operation can be performed after completing the mount or the snap operations. The steps required to recover a database manually are outlined in Figure 16-5.

Figure 16-5 Recovering Database Manually



To recover the database manually

1 Mount the database without recovering or applying the logs.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap -o mountdb \
-a oracle --oracle_sid sfaedb --oracle_home /ora_base/db_home \
--flashsnap_name TEST1 --clone_name clone1
Mounting snapshot volumes ... Done
Preparing parameter file for clone database ... Done
Mounting clone database ... Done
```

If you execute this command after the snap operation, it mounts the file systems and the clone database. Note that archive logs are not applied and the database is not open for read write operations. You can recover the database by using basic Oracle recovery commands.

See "Manual recovery of a clone database" on page 375.

See "Manual recovery after a restore operation" on page 376.

2 Update the repository to change the database state to `clone`.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap -o update_status \
-a oracle --oracle_sid sfaedb --oracle_home /ora_base/db_home \
--flashsnap_name TEST1
```

This step is for bookkeeping that needs to be performed to update the state of the FlashSnap service in repository to clone state after manual recovery.

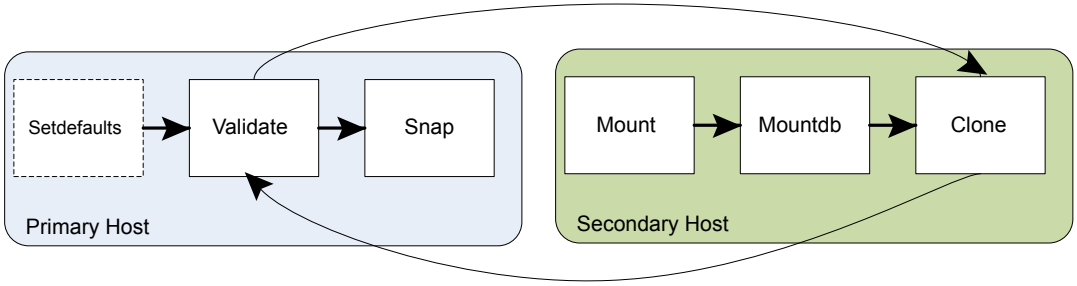
Cloning a database on the secondary host

You can use the `vxsfadm` command to perform end operations without performing the intermediate operations. For example, you need to perform the steps outlined in Figure 16-6.

Figure 16-6 Creating a Clone - Basic Workflow

However, you can use `vxsfadm` to go directly from Validate to Clone for cloning a database on the secondary host. Figure 16-7 outlines the steps for doing this operation.

Figure 16-7 Creating a Clone - Without Intermediate Operations



To clone a database on the secondary host

- Enter the following command.

```
$ vxsfadm -s flashsnap -a oracle -o clone \
--oracle_sid sfaedb --oracle_home /ora_base/db_home \
--flashsnap_name TEST1 --secondary_host host2
Retrieving snapshot information ...                          Done
Importing snapshot diskgroups ...                            Done
Mounting snapshot volumes ...                                Done
Preparing parameter file for clone database ...              Done
Mounting clone database ...                                  Done
Starting database recovery ...
Clone database clone1 is open.
```

In a 2-host setup with the primary host host1 and the secondary host host2, this command creates a clone database on host2.

The SID of the clone database is clone1 and it is mounted under the `/var/tmp` directory.

Note: ORACLE_HOME and User ID of the Oracle user are same on both the nodes. Default values are assigned to all of the other parameters.

As a DBA, if you need to mount the clone database on the secondary host to some other location with a different SID, use the following command.

```
$ vxsfadm - s flashsnap -a oracle -o clone --oracle_sid sfaedb \
--oracle_home /ora_base/db_home -flashsnap_name TEST1 \
--secondary_host host2  --clone_path /home/newclonepath \
--clone_name newsid
```

Chapter 17

Administering space-optimized snapshots

This chapter includes the following topics:

- Planning to create an instant space-optimized snapshot
- Preparing a legacy volume for the creation of an instant snapshot
- Creating a shared cache object
- Creating a clone of an Oracle database by using space-optimized snapshots
- Creating multiple clones using FlashSnap snapshots
- Recovering the clone database manually

Planning to create an instant space-optimized snapshot

The following considerations apply to creating an instant space-optimized snapshot:

- A space-optimized instant snapshot of a volume requires enough storage to record the original contents of the parent volume as they are changed during the life of the snapshot.
- The original volume must have a DCO (version 20 or above) and a DCO volume. Persistent FastResync must be enabled.

In this release, these requirements are the defaults. If you changed these settings or if the original volume was created with an earlier version of Storage Foundation, you can still create an instant snapshot.
See "Preparing a legacy volume for the creation of an instant snapshot" on page 207.

- If you need several space-optimized instant snapshots for the volumes in a disk group, you may find it convenient to create a single shared cache object in the disk group rather than a separate cache object for each snapshot.

Preparing a legacy volume for the creation of an instant snapshot

An instant snapshot requires that the original volume has a DCO volume. Persistent FastResync must be enabled, which is the default for Storage Foundation 6.0 and later. If the original volume was created with an earlier release of Storage Foundation, prepare the volume for an instant snapshot with the following procedure. This procedure may also be required if you previously disabled Persistent FastResync, or if you used the `vxsnap unprepare` command on the volume.

To prepare a legacy volume for the creation of an instant snapshot

1. Verify that the volume has a DCO volume that allows instant snapshots, and that FastResync is enabled on the volume:

```
# vxprint -g volumedg -F%instant volume
# vxprint -g volumedg -F%fastresync volume
```

   If both commands return a value of `on`, proceed to creating the snapshot.

   If one of the commands returns the value of `off`, continue with the next step.

2. Prepare the volume for instant snapshots:

```
# vxsnap [-g diskgroup] prepare volume [regionsize=size] \
  [ndcomirs=number] [alloc=storage_attributes]
```

   For example, to prepare the volume `myvol` in the disk group `mydg`, use the following command:

```
# vxsnap -g mydg prepare myvol regionsize=128k ndcomirs=2 \
  alloc=mydg10,mydg11
```

   This example creates a DCO object and redundant DCO volume with two plexes located on disks `mydg10` and `mydg11`, and associates them with `myvol`. The region size is also increased to 128KB from the default size of 64KB. The region size must be a power of 2, and be greater than or equal to 16KB. A smaller value requires more disk space for the change maps, but the finer granularity provides faster resynchronization.

Creating a shared cache object

If you need to create several instant space-optimized snapshots for the volumes in a disk group, you may find it more convenient to create a single shared cache object in the disk group rather than a separate cache object for each snapshot.

To create a shared cache object

1. Decide on the following characteristics that you want to allocate to the cache volume that underlies the cache object:

   - The size of the cache volume should be sufficient to record changes to the parent volumes during the interval between snapshot refreshes. A suggested value is 10% of the total size of the parent volumes for a refresh interval of 24 hours.

- If redundancy is a desired characteristic of the cache volume, it should be mirrored. This increases the space that is required for the cache volume in proportion to the number of mirrors that it has.
- If the cache volume is mirrored, space is required on at least as many disks as it has mirrors. These disks should not be shared with the disks used for the parent volumes. The disks should also be chosen to avoid impacting I/O performance for critical volumes, or hindering disk group split and join operations.

2 Having decided on its characteristics, use the `vxassist` command to create the volume that is to be used for the cache volume. The following example creates a mirrored cache volume, `cachevol`, with size 1GB in the disk group, `mydg`, on the disks `disk16` and `disk17`:

```
# vxassist -g mydg make cachevol 1g layout=mirror \
  init=active disk16 disk17
```

The attribute `init=active` is specified to make the cache volume immediately available for use.

3 Use the `vxmake cache` command to create a cache object on top of the cache volume that you created in the previous step:

```
# vxmake [-g diskgroup] cache cache_object \
  cachevolname=volume [regionsize=size] [autogrow=on] \
  [highwatermark=hwmk] [autogrowby=agbvalue] \
  [maxautogrow=maxagbvalue]]
```

If you specify the region size, it must be a power of 2, and be greater than or equal to 16KB (`16k`). If not specified, the region size of the cache is set to 64KB.

Note: All space-optimized snapshots that share the cache must have a region size that is equal to or an integer multiple of the region size set on the cache. Snapshot creation also fails if the original volume's region size is smaller than the cache's region size.

If the cache is not allowed to grow in size as required, specify `autogrow=off`. By default, the ability to automatically grow the cache is turned on.

In the following example, the cache object, `cobjmydg`, is created over the cache volume, `cachevol`, the region size of the cache is set to 32KB, and the `autogrow` feature is enabled:

```
# vxmake -g mydg cache cobjmydg cachevolname=cachevol \
  regionsize=32k autogrow=on
```

4 Having created the cache object, use the following command to enable it:

```
# vxcache [-g diskgroup] start cache_object
```

For example to start the cache object, `cobjmydg`:

```
# vxcache -g mydg start cobjmydg
```

Creating a clone of an Oracle database by using space-optimized snapshots

You can use space-optimized snapshots to create a clone of an Oracle database by performing the following procedure.

See "vxsfadm command reference" on page 323.

See "Space-optimized snapshots configuration parameters" on page 331.

See "Space-optimized snapshots supported operations" on page 333.

For details, refer to `vxsfadm-sos`(1M) and `vxsfadm-oracle-sos`(1M) man pages.

To create a clone of an Oracle database by using space-optimized snapshots

1 Create a configuration file.

```
$ /opt/VRTSdbed/bin/vxsfadm -s sos -a oracle \
 -o setdefaults --oracle_sid sfaedb --oracle_home /ora_base/db_home \
 --sos_name TEST1 -c dailyconfig
Written config file dailyconfig
```

This command creates a default configuration file at the location mentioned by the argument given to the **-c** option. If a path is not specified, then the configuration file is created in the present working directory with all the parameters and default values. You can change the parameters, if required.

Note: If a configuration file with the specified name already exists, then specify a unique configuration name, different path, or remove the existing configuration file. Also, the destroy operation does not delete a configuration file.

---

2 Validate the setup.

```
$ /opt/VRTSdbed/bin/vxsfadm -s sos -a oracle -o validate --oracle_sid \
sfaedb --oracle_home /ora_base/db_home --sos_name TEST1
Validating database configuration for space-optimized snapshot:
ORACLE_SID = sfaedb
ORACLE_HOME = /ora_base/db_home
APP_MODE = online
ARCHIVELOG_DEST = /arch/sfaedb
SNAPSHOT_ARCHIVE_LOG = yes
Database validation successful.
Validating database volume layout for space-optimized snapshot:
Data volumes ready for snapshot:
Volume/volume-set datavol of diskgroup datadg mounted on /data.
Archivelog volume ready for snapshot:
Volume/volume-set archvol of diskgroup datadg mounted on /arch.
CACHE_OBJECT = datadg:cacheobj
SNAPSHOT_VOL_PREFIX = SNAP_
Database volume layout validated successfully.
```

This command validates the configuration file and the database environment. In case of any problem, appropriate error messages are displayed that you can use to correct the problem and then retry.

3 Create a space-optimized snapshot of the database.

```
$ /opt/VRTSdbed/bin/vxsfadm -s sos -a oracle \
-o snap --oracle_sid sfaedb --oracle_home /ora_base/db_home \
--sos_name TEST1
Validating database configuration for space-optimized snapshot:
ORACLE_SID = sfaedb
ORACLE_HOME = /ora_base/db_home
APP_MODE = online
ARCHIVELOG_DEST = /arch/sfaedb
SNAPSHOT_ARCHIVE_LOG = yes
Database validation successful.
Putting database in backup mode ... Done
Validating database volume layout for space-optimized snapshot:
Data volumes ready for snapshot:
Volume/volume-set datavol of diskgroup datadg mounted on /data.
Archivelog volume ready for snapshot:
Volume/volume-set archvol of diskgroup datadg mounted on /arch.
CACHE_OBJECT = datadg:cacheobj
SNAPSHOT_VOL_PREFIX = SNAP_
Database volume layout validated successfully.
Creating snapshot volumes for data volumes ...          Done
Taking database out of backup mode ...                  Done
Flushing online redo logs ...                           Done
Creating snapshot volume for archivelog volume ...      Done
```

This command creates space-optmized snapshot volumes from the primary volumes by using the specified cache objects.

Note: At the validation stage, all the parameters, including the mandatory parameters `--oracle_sid` and `--sos_name`, are read and stored in the repository.

---

4 Mount the snapshot.

```
$ /opt/VRTSdbed/bin/vxsfadm -s sos -a oracle \
-o mount --oracle_sid sfaedb --oracle_home /ora_base/db_home \
--sos_name TEST1
Mounting snapshot volumes ...                          Done
```

If you need to specify the clone path, use the CLONE_PATH variable. If this variable is not specifed, the snapshot is mounted under the `/var/tmp` directory, by default.

```
$ vxsfadm -s sos -a oracle -o mount --oracle_sid  sfaedb \
--sos_name TEST1 --clone_path /cloneoracle
```

Note: Ensure that the Oracle user has the required permissions to create the `/cloneoracle` directory, if it does not exist.

---

5 Clone the database based on the snapshot.

```
$ /opt/VRTSdbed/bin/vxsfadm -s sos -a oracle -o clone \
--oracle_sid sfaedb --oracle_home /ora_base/db_home \
--sos_name TEST1
Preparing parameter file for clone database ...        Done
Mounting clone database ...                            Done
Starting database recovery ...
Clone database sfaOjqwF is open.
```

If you have not specified *clone_name*, it is automatically generated.

If you need to change a memory tuning parameter of the database clone such as memory_target , you can specify a modified pfile during the clone operation.

You can also use space-optimized snapshots in a replicated environment.

See "Considerations for using volume snapshots with VVR" on page 313.

See "Creating space-optimized snapshots in a VVR setup" on page 314.

Space-optimized snapshots in HA environment

If you are using space-optimized snapshots to clone an Oracle database in a VCS cluster, ensure the following before a switchover:

- If you are using volume agents, unmount and destroy the clone, and then disable the cache object. After the switchover, you can take a snapshot of the database and recreate the clone database.

- If you do not want to destroy the clone and then recreate it, remove volume agents for database volumes, if there are any. Use diskgroup and mount agents instead.

Creating multiple clones using FlashSnap snapshots

Online mode, third-mirror-break-off type snapshot i.e. online FlashSnap snapshot of a database instance contains all the information needed to create a clone of the database instance. It can act as a template for creating the clone database instances. You can thus allocate a FlashSnap snapshot that can be used as a master copy for creating one or more clone instances.

The clone instances created from a FlashSnap image, termed as the 'golden image', are incremental copies of the master or the golden image. These depend on the FlashSnap image for their operations.

You can create multiple database instance clones from a single backup image using the staggering technique. Here you create a space-optimized snapshot of a FlashSnap snapshot, and bring up a clone database instance.

Two new configurations parameters are used to support this particular use case.

- EXTRA_OBJECTS [short form --extra]
- SOURCE_SNAPSHOT [short form --source]

You need a cache object to create a space-optimized snapshot. This object can either be included in the FlashSnap snapshot, or be created upon snapshot. EXTRA_OBJECTS is used to specify a cache object to be included in the FlashSnap snapshot. This parameter is not needed if you create a cache object after the creation of a FlashSnap snapshot.

To create database clones using space-optimized snapshots from a FlashSnap golden image

1 Create an online FlashSnap snapshot (golden image) to be used as the source snapshot.

```
$ vxsfadm -s flashsnap -a oracle --name friday -o snapshot \
--oracle_sid sfaedb --oracle_home /ora_base/db_home \
--extra_objects datadg:cacheobj --extra_objects datadg2:cacheobj2
Validating database configuration for third-mirror-break-off snapshot:
ORACLE_SID                  = sfaedb
ORACLE_HOME                 = /ora_base/db_home
APP_MODE                    = online
ARCHIVELOG_DEST             = /arch/sfaedb
SNAPSHOT_ARCHIVE_LOG        = yes
Database validation successful.
snapshot started at Mon Jan 30 15:05:15 2012.
Putting database in backup mode ...                          Done
Validating database volume layout for third-mirror-break-off snapshot:
Data volumes ready for snapshot:
Volume/volume-set datavol2 of diskgroup datadg2 mounted on /data2.
Storage units to be used for snapshot from diskgroup datadg2:
disk_6  Volume/volume-set datavol of diskgroup datadg mounted on /data.
Storage units to be used for snapshot from diskgroup datadg:
disk_3
Archivelog volume ready for snapshot:
Volume/volume-set archvol of diskgroup datadg mounted on /arch.
Storage units to be used for snapshot from diskgroup datadg:
disk_4
SNAPSHOT_VOL_PREFIX         = SNAP_
SNAPSHOT_DG_PREFIX          = SNAP_
Database volume layout validated successfully.
Creating snapshot volumes for data volumes ...               Done
Taking database out of backup mode ...                       Done
Flushing online redo logs ...                                Done
Creating snapshot volume for archivelog volume ...           Done
Copying snapshot information to snapshot volume ...          Done
Creating snapshot diskgroups ...                             Done
Deporting snapshot diskgroups ...                            Done
SNAP_datadg2 SNAP_datadg
snaphot ended at Mon Jan 30 15:05:28 2012.
```

You can use the `--extra_objects` option to include cache objects from the original disk group in the snapshot disk groups. In the example procedure, cache objects cacheobj and cacheobj2 from datadg and datadg2 disk groups are included in the snapshot disk groups SNAP_datadg and SNAP_datadg2.

If you do not use this option, you must manually create cache objects in the snapshot disk groups.

See “FlashSnap configuration parameters” on page 327.

2 Create clones using space-optimized snapshots.

```
$ vxsfadm -s sos -a oracle --name reporting --source-snapshot \
friday --prefix RPT -o clone -r primary-host
Validating database configuration for space-optimized snapshot:
ORACLE_SID                  = sfaedb
ORACLE_HOME                 = /ora_base/db_home
APP_MODE                    = online
ARCHIVELOG_DEST             = /arch/sfaedb
SNAPSHOT_ARCHIVE_LOG        = yes
Database validation successful.
Validating source snapshot friday ...                        Done
Validating database volume layout for space-optimized snapshot:
CACHE_OBJECT                = SNAP_datadg2:cacheobj2,SNAP_datadg:cacheobj
SNAPSHOT_VOL_PREFIX         = RPT
Database volume layout validated successfully.
Creating snapshot volumes for data volumes ...               Done
Generated mount path: /var/tmp/IKrlBXRcZ_
Mounting snapshot volumes ...                                Done
Cloning using autogenerated SID ( sfaxwYXY )
Preparing parameter file for clone database ...              Done
Mounting clone database ...                                  Done
Starting database recovery ...
Clone database sfaxwYXY is open.
```

This command creates a clone database using the FlashSnap snapshot friday as the golden image. You can create multiple space-optimized snapshots using the same golden image.

The golden image snapshot, like any other FlashSnap snapshot, can be imported on a host sharing storage with the host running production database. You can create database clones using space-optimized snapshots from a FlashSnap golden image in such off-host scenarios as well. Note that you must use `vxsfadm -r` option in such cases.

You can create the clones on a host different from the host on which the production database is running. In this scenario, you must specify the primary host using the `-r` option of `vxsfadm`. When the clone is no longer required, you can delete it by using the `-o delete` option of `vxsfadm`.

Note: You cannot resynchronize the FlashSnap snapshot while there exist space-optimized snapshots created by using that FlashSnap snapshot as a golden image.

See "vxsfadm command reference" on page 323.

See "Space-optimized snapshots configuration parameters" on page 331.

See "Space-optimized snapshots supported operations" on page 333.

Recovering the clone database manually

For Oracle databases, you can perform some special operations such as recovering the database to a System Change Number (SCN) or a particular time. You can use the mountdb operation to perform the database recovery. This operation can be performed after completing the mount or the snap operations.

To recover the database manually

1. Mount the database without recovering or applying the logs.

```
$ /opt/VRTSdbed/bin/vxsfadm -s sos -a oracle \
-o mountdb --oracle_sid sfaedb --oracle_home /ora_base/db_home \
--sos_name TEST1
Mounting snapshot volumes ...                                Done
```

   If you execute this command after the snap operation, it mounts the file systems and the clone database. Please note that archive logs are not applied and the database is not open for read write operations. You can recover the database by using basic Oracle recovery commands.

   See "Manual recovery of a clone database" on page 375.

   See "Manual recovery after a restore operation" on page 376.

2. Update the repository to change the database state to `clone`.

```
$ /opt/VRTSdbed/bin/vxsfadm -s sos -a oracle \
-o update_status --oracle_sid sfaedb \
--oracle_home /ora_base/db_home --sos_name TEST1
```

   This step is for bookkeeping that needs to be performed to update the state of the space-optimized snapshots service in repository to clone state after manual recovery.

Chapter 18

Administering Storage Checkpoints

This chapter includes the following topics:

- About Storage Checkpoints
- Database Storage Checkpoints for recovery
- Creating a Database Storage Checkpoint
- Deleting a Database Storage Checkpoint
- Mounting a Database Storage Checkpoint
- Unmounting a Database Storage Checkpoint
- Creating a database clone using a Database Storage Checkpoint
- Restoring database from a Database Storage Checkpoint
- Gathering data for offline-mode Database Storage Checkpoints

About Storage Checkpoints

Veritas File System (VxFS) provides a Storage Checkpoint feature that quickly creates a persistent image of a file system at an exact point in time. Storage Checkpoints significantly reduce I/O overhead by identifying and maintaining only the file system blocks that have changed since the last Storage Checkpoint or backup via a copy-on-write technique.

See “Copy-on-write” on page 173.

Storage Checkpoints provide:

- Persistence through reboots and crashes.

- The ability for data to be immediately writeable by preserving the file system metadata, the directory hierarchy, and user data.

Storage Checkpoints are actually data objects that are managed and controlled by the file system. You can create, remove, and rename Storage Checkpoints because they are data objects with associated names.

See "How a Storage Checkpoint works" on page 171.

Unlike a disk-based mirroring technology that requires a separate storage space, Storage Checkpoints minimize the use of disk space by using a Storage Checkpoint within the same free space available to the file system.

After you create a Storage Checkpoint of a mounted file system, you can also continue to create, remove, and update files on the file system without affecting the logical image of the Storage Checkpoint. A Storage Checkpoint preserves not only the name space (directory hierarchy) of the file system, but also the user data as it existed at the moment the file system image was captured.

You can use a Storage Checkpoint in many ways. For example, you can use them to:

- Create a stable image of the file system that can be backed up to tape.
- Provide a mounted, on-disk backup of the file system so that end users can restore their own files in the event of accidental deletion. This is especially useful in a home directory, engineering, or email environment.
- Create a copy of an application's binaries before installing a patch to allow for rollback in case of problems.
- Create an on-disk backup of the file system in that can be used in addition to a traditional tape-based backup to provide faster backup and restore capabilities.
- Test new software on a point-in-time image of the primary fileset without jeopardizing the live data in the current primary fileset by mounting the Storage Checkpoints as writable.

Database Storage Checkpoints for recovery

A Database Storage Checkpoint creates an exact image of a database instantly and provides a consistent image of the database from the point in time the Database Storage Checkpoint was created. The Database Storage Checkpoint image is managed and available through the command line interface (CLI).

Because each Database Storage Checkpoint is a consistent point-in-time image of a file system, Storage Rollback is the restore facility for these on-disk backups. Storage Rollback rolls back the changed blocks that are contained in a Database Storage Checkpoint into the primary file system for faster database restoration.

The combination of data redundancy (disk mirroring) and Database Storage Checkpoints is recommended for highly critical data to protect them from both physical media failure and logical errors.

Advantages and limitations of Database Storage Checkpoints

Database Storage Checkpoints and rollback provides the following advantages:

- Initially, a Database Storage Checkpoint contains no data—it contains only the inode list and the block map of the primary fileset. The block map points to the actual data on the primary file system.
- Because only the inode list and block map are needed and no data is copied, creating a Database Storage Checkpoint takes only a few seconds and very little space.
- A Database Storage Checkpoint keeps track of block change information and thereby enables incremental database backup at the block level.
- A Database Storage Checkpoint helps recover data from incorrectly modified files.
- A Database Storage Checkpoint can be mounted, allowing regular file system operations to be performed. Mounted Database Storage Checkpoints can be used for a wide range of application solutions that include backup, investigations into data integrity, staging upgrades or database modifications, and data replication solutions.

The limitations of Database Storage Checkpoints are as follows:

- Database Storage Checkpoints can only be used to restore from logical errors (for example, a human error).
- Because all the data blocks are on the same physical device, Database Storage Checkpoints cannot be used to restore files due to a media failure. A media failure requires a database restore from a tape backup or a copy of the database files that are kept on a separate medium.

Creating a Database Storage Checkpoint

You can use the Storage Checkpoint feature of Storage Foundation to create a Database Storage Checkpoint of a database by performing the following procedure.

See “vxsfadm command reference” on page 323.

See “Database Storage Checkpoints configuration parameters” on page 336.

See “Database Storage Checkpoints supported operations” on page 339.

For details, refer to `vxsfadm-checkpoint`(1M) and `vxsfadm-oracle-checkpoint`(1M) man pages.

To create a Database Storage Checkpoint

- Use the following command.

```
$ /opt/VRTSdbed/bin/vxsfadm -s checkpoint -o create -a oracle \
--oracle_sid=sfaedb --oracle_home=/ora_base/db_home \
--checkpoint_name=XYZ
Putting database in backup mode... Done
Creating Storage Checkpoint XYZ ... Done
Storage Checkpoint XYZ created
Removing the database from backup mode... Done
```

  This command creates a read-only Database Storage Checkpoint of the Oracle database of the specified name. If the *checkpoint_name* parameter is not specified, a name is automatically generated.

Deleting a Database Storage Checkpoint

You can delete a Database Storage Checkpoint as follows.

To delete a Database Storage Checkpoint

- Use the following command.

```
$ /opt/VRTSdbed/bin/vxsfadm -s checkpoint -o delete -a oracle \
--oracle_sid=sfaedb --oracle_home=/ora_base/db_home \
--checkpoint_name=XYZ
Deleting the Checkpoint XYZ...
Deleting Checkpoint from /data... Done
```

  This command deletes the Database Storage Checkpoint XYZ.

Mounting a Database Storage Checkpoint

You can mount a Database Storage Checkpoint as follows.

To mount a Database Storage Checkpoint

- Do one of the following.
  - Use the following command to mount the Database Storage Checkpoint in the read-only mode.

```
$ /opt/VRTSdbed/bin/vxsfadm -s checkpoint -o mount -a oracle \
--oracle_sid=sfaedb --oracle_home=/ora_base/db_home \
--checkpoint_name=XYZ --mount_path=/tmp/abcd
Mounting Checkpoint to /tmp/abcd... Done
```

This command mounts the Database Storage Checkpoint to the specified mount point.
If the *checkpoint_name* parameter is not specified, a name is automatically generated.

- Use the following command to mount the Database Storage Checkpoint in the read-write mode.

```
$ /opt/VRTSdbed/bin/vxsfadm -s checkpoint -o mountrw -a oracle \
--oracle_sid=sfaedb --oracle_home=/ora_base/db_home \
--checkpoint_name=XYZ --clone_path=/tmp/abcd
Creating Storage Checkpoint XYZ_rw_1316516805 ... Done
Storage Checkpoint XYZ_rw_1316516805 created
Mounting Checkpoint to /tmp/abcd... Done
```

This command creates a Storage Checkpoint of the Database Storage Checkpoint XYZ and then mounts the newly created Database Storage Checkpoint to the specified mount point in the read-write mode.
If the *checkpoint_name* parameter is not specified, a name is automatically generated.

Note: This command maintains the point-in-time copy of the original Database Storage Checkpoint.

Unmounting a Database Storage Checkpoint

You can unmount a Database Storage Checkpoint as follows.

To unmount a Database Storage Checkpoint

◆ Use the following command.

```
$ /opt/VRTSdbed/bin/vxsfadm -s checkpoint -o umount -a oracle \
--oracle_sid=sfaedb --oracle_home=/ora_base/db_home \
--checkpoint_name=XYZ
Unmounting the checkpoint... Done
```

This command unmounts the Database Storage Checkpoint XYZ.

Creating a database clone using a Database Storage Checkpoint

You can create a database clone by using a Database Storage Checkpoint as follows.

To create a clone of a Database Storage Checkpoint

- Use the following command.

```
$ /opt/VRTSdbed/bin/vxsfadm -s checkpoint -o clone -a oracle \
--oracle_sid=sfaedb --oracle_home=/ora_base/db_home \
--checkpoint_name=XYZ --clone_path=/tmp/abcd --clone_name clone1
Creating Storage Checkpoint XYZ_rw_1316523411 ... Done
Storage Checkpoint XYZ_rw_1316523411 created
Mounting Checkpoint to /tmp/abcd... Done
Cloning the Pfile ... Done
Mounting the database... Done
Recovering the clone database... Done
```

This command creates a Storage Checkpoint of the Database Storage Checkpoint XYZ and then mounts the newly created Database Storage Checkpoint to the the specified mount point in the read-write mode.

If you do not specify *checkpoint_name* and *clone_name*, they are automatically generated.

Note: This command maintains the point-in-time copy of the original Database Storage Checkpoint.

Restoring database from a Database Storage Checkpoint

You can restore your database from a Database Storage Checkpoint as follows.

Note: The following operation requires your primary database to be shut down.

To restore the database from a Database Storage Checkpoint

- Use the following command.

```
$ /opt/VRTSdbed/bin/vxsfadm -s checkpoint -o restore -a oracle \
--oracle_sid=sfaedb --oracle_home=/ora_base/db_home \
--checkpoint_name=XYZ
The SCN at the time of taking the checkpoint was : 1170070
Rolling back the application files... Done

You may want to use the following file(s) for application recovery
/var/tmp/hCgJmBm4AT/control01.ctl
/var/tmp/hCgJmBm4AT/binary_controlfile
```

This command restores all the data files of the application to the point-in-time when the Storage Checkpoint was created.

Note: Rollback of Database Storage Checkpoints that are mounted in the read-write mode is not supported.

After the restore operation, you can bring up the database by using standard recovery techniques. The control file provided is the one present at the time the Storage Checkpoint was created. This control file can be used if there have been some structural changes to the database after the Storage Checkpoint was created.

Note: The following operation requires the tablespace and all data files in the tablespace to be offline.

To restore the data files in a tablespace from a Database Storage Checkpoint

- Use the following command.

```
$ /opt/VRTSdbed/bin/vxsfadm -s checkpoint -o restore -a oracle \
--oracle_sid=sfaedb --oracle_home=/ora_base/db_home \
--checkpoint_name=XYZ --tablespace_rollback newtbs
Rolling back the application files... Done
```

This command restores all the data files in the specified tablespace to the point-in-time when the Database Storage Checkpoint was created.

After the restore operation, you must recover the tablespace by using standard recovery techniques.

Note: The TEMP tablespaces cannot be rolled back.

Note: The following operation requires the data files that need to be restored to be offline.

To restore specified data files from a Database Storage Checkpoint

- Use the following command.

```
$ /opt/VRTSdbed/bin/vxsfadm -s checkpoint -o restore -a oracle \
--oracle_sid=sfaedb --oracle_home=/ora_base/db_home \
--checkpoint_name=XYZ \
--datafile_rollback /data/sfaedb/newf01.dbf,/data/sfaedb/newf02.dbf
Rolling back the application files... Done
```

  This command restores all the specified data files to the point-in-time when the Database Storage Checkpoint was created.

  You can specify the data files to restore by using one of the following options:

  - datafile_rollback: A list of data files separated by comma.
  - datafile_list_rollback: A file that contains a list of data files separated by newlines.

  After the restore operation, you must recover the data files by using the standard recovery procedures.

Gathering data for offline-mode Database Storage Checkpoints

You can gather data for offline-mode Database Storage Checkpoints as follows.

Note: You can perform the following operation only when the database is up.

To gather information necessary to create a Storage Checkpoint when the database is offline

- Use the following command.

```
$ /opt/VRTS/bin/vxsfadm -s checkpoint -a oracle -o getappdata
Gathering offline data... Done
```

Note: If you attempt to create an offline Database Storage Checkpoint without performing the data gathering operation, the Checkpoint creation fails.

Chapter 19

Administering FileSnap snapshots

This chapter includes the following topics:

- Preparing to use FileSnap
- Creating a clone of an Oracle database by using FileSnap
- Destroying a database clone created by using FileSnap
- Destroying a FileSnap snapshot
- Restoring database files from a FileSnap snapshot
- Viewing the list of snapshots and clones created by using FileSnap

Preparing to use FileSnap

To use FileSnap, ensure the following:

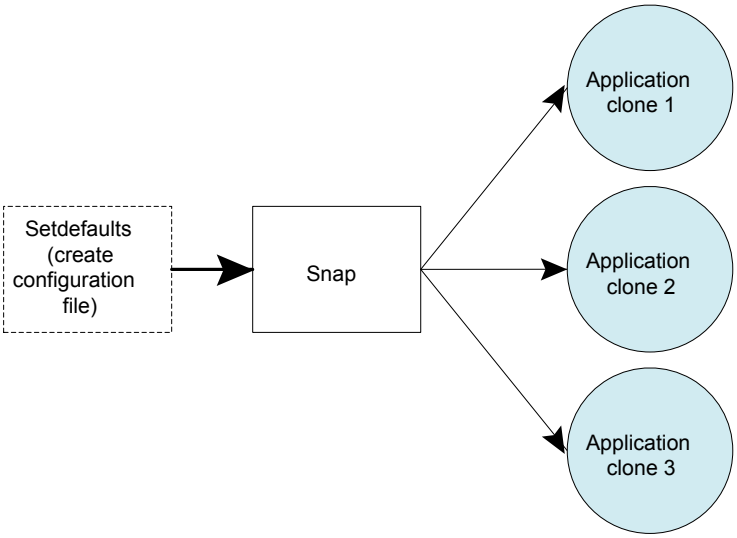
- The underlying VxFS file systems are using disk layout Version 8 or later.
- The Veritas InfoScale Enterprise and Storage licenses are available.

If some of the file systems are using layout Version 7 or earlier, use the `vxupgrade` utility to upgrade to Version 8 or later while the file systems are online.

Creating a clone of an Oracle database by using FileSnap

You can use FileSnap to create a clone of an Oracle database by performing the steps outlined in Figure 19-1.

Figure 19-1 Creating a Clone - FileSnap



See "vxsfadm command reference" on page 323.

See "FileSnap configuration parameters" on page 334.

See "FileSnap supported operations" on page 335.

For details, refer to `vxsfadm-filesnap`(1M) and `vxsfadm-oracle-filesnap`(1M) man pages.

To create a clone of an Oracle database by using FileSnap

1. Create a configuration file.

```
$ vxsfadm -s filesnap -a oracle --oracle_sid sfaedb -oracle_home \
 '/ora_base/db_home' --filesnap_name 'daily_snap' \
 -c dailyconfig -o setdefaults

Written config file dailyconfig
```

This command creates a default configuration file at the location mentioned by the argument given in the **-c** option, if a path is not specified, then the configuration file is created in the present working directory with all the parameters and default values. You can change the parameters, if required.

Note: If a configuration file of the same name exists, then specify a different configuration name, a different path, or remove the existing configuration file.

Note: The destroy operation does not delete the configuration file.

---

2. Create a snapshot of the database.

```
$ vxsfadm -s filesnap -a oracle --oracle_sid sfaedb -oracle_home \
'/ora_base/db_home' --filesnap_name 'daily_snap' -o snap
Validating database configuration for filesnap...
Database validation successful.
ORACLE_SID = sfaedb
ORACLE_HOME = /ora_base/db_home
APP_MODE = online
Validating filesystem configuration for filesnap...
Database filesystem configuration validated successfully.
Starting database online backup mode... Done
Snapshotting files... Done
Stopping database online backup mode... Done
```

This command creates a FileSnap snapshot of all the data files. All the temp files, archive logs, and redo logs are excluded.

3 Create a clone of the primary database based on the snapshots of the data files.

```
$ vxsfadm -s filesnap -a oracle --oracle_sid sfaedb -oracle_home \
'/ora_base/db_home' --filesnap_name 'daily_snap' \
 --clone_name clone1 -o clone
Cloning files... Done
Preparing parameter file for clone database... Done
Mounting clone database... Done
Starting clone database recovery... Done
Clone database clone1 is open.
```

FileSnap provides the flexibility to create multiple database clones by using one FileSnap snapshot. To create another clone from the FileSnap snapshot, use the following command:

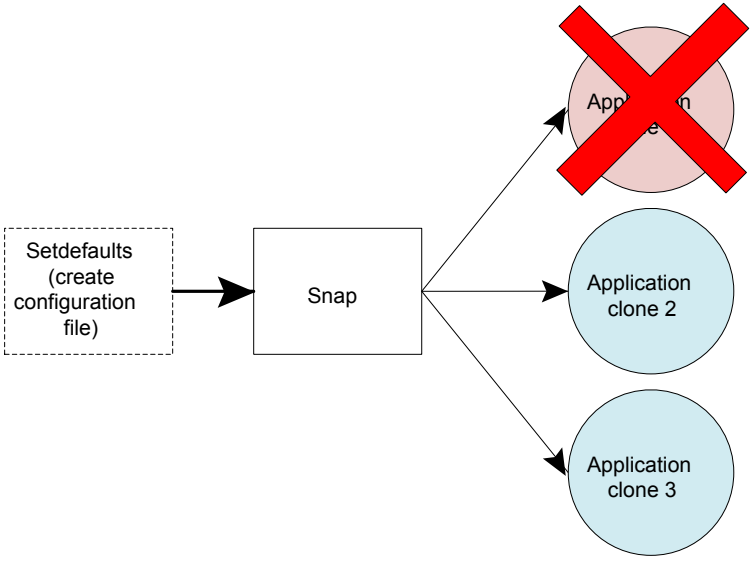
```
$ vxsfadm -s filesnap -a oracle --oracle_sid sfaedb -oracle_home \
'/ora_base/db_home' --filesnap_name 'daily_snap' \
--clone_name clone2 -o clone
Cloning files... Done
Preparing parameter file for clone database... Done
Mounting clone database... Done
Starting clone database recovery... Done
Clone database clone2 is open.
```

Note: You must specify the *clone_name* parameter. It is not automatically generated in case it is not specified on the command line or in the configuration file.

Destroying a database clone created by using FileSnap

You can destroy a clone of a database that is created by using a FileSnap snapshot. Figure 19-2 provides an overview of this operation.

Figure 19-2 Destroying a Clone - FileSnap



To destroy a database clone

- Use the following command.

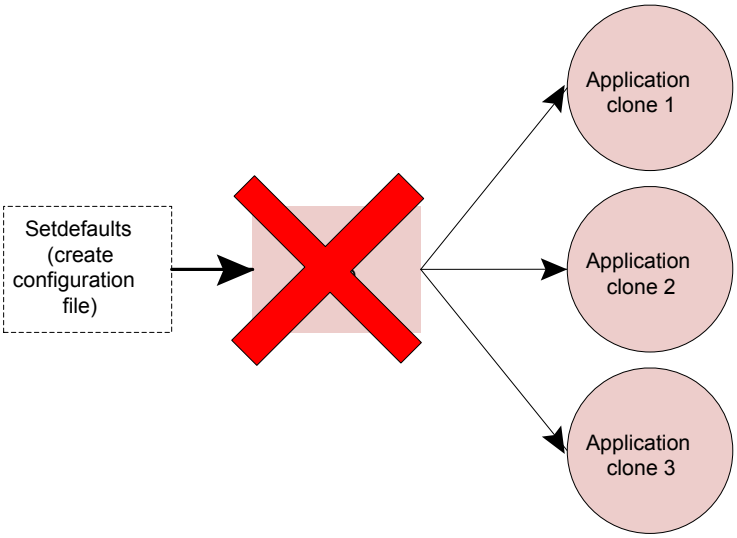
```
$ vxsfadm -s filesnap -a oracle --oracle_sid sfaedb -oracle_home \
'/ora_base/db_home' --filesnap_name 'daily_snap' \
 --clone_name clone1 -o destroyclone
Shutting down clone database... Done
Destroying clone... Done
```

This command shuts down the Oracle database clone and destroys the files that are associated with the clone. This operation does not destroy the original snapshot.

Destroying a FileSnap snapshot

You can destroy the FileSnap snapshots of your data files by using the procedure outlined in Figure 19-3.

Figure 19-3 Destoying a Snapshot - FileSnap



Note: Before destroying a snapshot, you must destroy all database clones that were created by using the snapshot.

To destroy a FileSnap snapshot

- Use the following command.

```
$ vxsfadm -s filesnap -a oracle --oracle_sid sfaedb \
-oracle_home '/ora_base/db_home' --filesnap_name 'daily_snap' \
 -o destroysnap
Destroying snapshot... Done
```

Restoring database files from a FileSnap snapshot

You can restore the database files from a FileSnap snapshot that was taken at an earlier point of time, with the current data files. This operation is useful in case of data corruption, logical errors, and other scenarios in which you want to restore data files from a snapshot.

Note: This operation requires the primary database to be shut down.

Note: This operation does not recover the database.

To restore a FileSnap snapshot

- Use the following command.

```
$ vxsfadm -s filesnap -a oracle -o restore --oracle_sid \
 sfaedb -oracle_home '/ora_base/db_home' --filesnap_name 'daily_snap'
 The SCN at the time of taking the filesnap was : 1174830
 Restoring files... Done

 You may want to use the following file(s) for application recovery
 /var/tmp/VrBJ8N5XaA/binary_controlfile
```

Viewing the list of snapshots and clones created by using FileSnap

You can use the following procedure to view the list of snapshots that were created by using FileSnap and the clones that were created by using those snapshots.

To view the list of snapshots and clones created on a database instance

- Use the following command.

```
$ vxsfadm -s filesnap -a oracle -o list --oracle_sid sfaedb
The following snapshot(s)/clone(s) are available:
SNAPSHOT NAME     CLONE NAME     STATUS
daily_snap                       Done
                  clone1         Done
                  clone2         Done
```

Chapter 20

Backing up and restoring with Netbackup in an SFHA environment

This chapter includes the following topics:

- About Veritas NetBackup
- About using Veritas NetBackup for backup and restore for Oracle
- About using Veritas NetBackup to backup and restore Oracle Disk Manager files
- Using NetBackup to backup and restore Quick I/O files for Oracle
- Using NetBackup in an SFHA Solutions product environment

About Veritas NetBackup

NetBackup provides backup, archive, and restore capabilities for database files and directories contained on client systems in a client-server network. NetBackup server software resides on platforms that manage physical backup storage devices. The NetBackup server provides robotic control, media management, error handling, scheduling, and a repository of all client backup images.

Administrators can set up schedules for automatic, unattended full and incremental backups. These backups are managed entirely by the NetBackup server. The administrator can also manually back up clients. Client users can perform backups, archives, and restores from their client system, and once started, these operations also run under the control of the NetBackup server.

NetBackup, while not a shipped component of Storage Foundation Enterprise products, can be purchased separately.

How NetBackup block-level incremental backup works for Oracle database files

Block-Level Incremental (BLI) Backup extends the capabilities of NetBackup to back up only changed data blocks of Oracle database files. BLI Backup accomplishes this backup methodology using the Storage Checkpoint facility in the Veritas File System (VxFS) available through Storage Foundation for Databases tools. BLI Backup reduces both the time required to complete a database backup and the amount of data transferred during backups. It also allows more frequent backups, resulting in more up-to-date backup images. When restoring from backups, the restore time is increased only by the extra time needed for NetBackup to apply the incremental backups after a full restore completes. However, frequent incremental backups can speed up the database recovery by reducing the number of redo logs to apply.

BLI Backup is particularly useful in a database environment where a database can be hundreds of gigabytes or terabytes. Using traditional backup methods for an offline database backup, any change in the database file—no matter how small—requires backing up the entire database file. Using BLI Backup, only modified data blocks need to be backed up.

Note: To allow BLI Backups, the database must be on VxFS file systems using the Version 4, 5, 6, or 7 layout, and you must have a valid Veritas InfoScale Storage Foundation Enterprise license. Use the `fstyp -v device` command to determine the layout version of your file system. See the `vxupgrade`(1M) manual page for more information.

For information on how to install, configure, and use NetBackup for Oracle Advanced BLI Agent to perform Oracle database backups, see the *Veritas NetBackup for Oracle Advanced BLI Agent System Administrator's Guide*.

About using Veritas NetBackup for backup and restore for Oracle

Veritas NetBackup provides for high performance, online (hot) backups of databases that must be available on a 24x7 basis, as well as offline (cold) database backups. Veritas NetBackup enables you to back up and restore database files and directories. You can set up schedules for automatic, unattended, online, and offline database

backup, as well as full or incremental backup. These backups are managed entirely by the NetBackup server. You can also manually back up database files from any of the NetBackup clients. Client users can perform database backups and restores from their client systems on demand.

About using Veritas NetBackup to backup and restore Oracle Disk Manager files

Oracle allocates Oracle Disk Manager files with contiguous extent layouts for good database performance. When you back up your database using Veritas NetBackup, extent attributes are backed up automatically. When you restore database files they are allocated using these extent attributes. If you are using Oracle RMAN's conventional backup method with any backup software, datafiles are also restored with the proper extent layouts.

If you are not using NetBackup or you are using RMAN's “proxy copy” backup method with a backup software other than NetBackup, the extent attributes may not be backed up. To ensure the restored datafiles have proper extent layouts, preallocate the lost datafiles using the `odmmkfile` command. This command preallocates contiguous space for files prior to restoring them.

See the `odmmkfile`(1M) manual page.

For example, to preallocate an Oracle datafile with size 100 M, assuming the Oracle database block size is 8K, use the `odmmkfile` command and enter:

```
$ /opt/VRTS/bin/odmmkfile -h 8k -s 100m filename
```

Using NetBackup to backup and restore Quick I/O files for Oracle

If you are using NetBackup for Oracle, then you should use Oracle RMAN to backup and restore Quick I/O files.

NetBackup does not follow symbolic links when backing up files. Typical backup management applications are designed this way to avoid backing up the same data twice. This would happen if both the link and the file it points to were included in the list of files to be backed up.

A Quick I/O file consists of two components: a hidden file with the space allocated for it, and a link that points to the Quick I/O interface of the hidden file. Because NetBackup does not follow symbolic links, you must specify both the Quick I/O link and its hidden file in the list of files to be backed up.

To view all files and their attributes in the `db01` directory:

```
$ ls -la /db01

total 2192

drwxr-xr-x    2 root   root   96   Oct 20 17:39 .

drwxr-xr-x    9 root   root   8192   Oct 20 17:39 ..

-rw-r--r--    1 oracle   dba   1048576   Oct 20 17:39

.dbfile

lrwxrwxrwx    1 oracle   dba   22   Oct 20 17:39

dbfile ->\
.dbfile::cdev:vxfs:
```

In the example above, you must include both the symbolic link `dbfile` and the hidden file `.dbfile` in the file list of the backup class.

If you want to back up all Quick I/O files in a directory, you can simplify the process by just specifying the directory to be backed up. In this case, both components of each Quick I/O file will be properly backed up. In general, you should specify directories to be backed up unless you only want to back up some, but not all files, in those directories.

Because NetBackup is integrated with Storage Foundation, NetBackup backs up extent attributes of a Quick I/O file and restores them accordingly. Quick I/O files can then be backed up and restored as regular files using NetBackup, while preserving the Quick I/O file's extent reservation. Without this feature, restoring the file could cause the loss of contiguous reservation, which can degrade performance.

When restoring a Quick I/O file, if both the symbolic link and the hidden file already exist, NetBackup will restore both components from the backup image. If either one or both of the two components are missing, NetBackup creates or overwrites as needed.

Note: The Oracle backup and restore utility cannot be used to back up and restore Quick I/O files.

Some back up software may not be able to back up and restore VxFS extent attributes. See the `qio_recreate`(1M) online manual page for information on how to restore missing Quick I/O files.

Using NetBackup in an SFHA Solutions product environment

You can enhance the ease of use and efficiency of your SFHA Solutions product and NetBackup by integrating them as follows:

- Clustering a NetBackup Master Server
- Backing up and recovering a VxVM volume using NetBackup
- Backing up and recovering an Oracle database using NetBackup

Clustering a NetBackup Master Server

To enable your NetBackup Master Server to be highly available in a cluster environment, use the following procedure.

To make a NetBackup Master Server, media, and processes highly available

1. Verify that your versions of NetBackup and Cluster Server are compatible. Detailed combination information is included in the NetBackup cluster compatibility list:
   - For NetBackup 7.x cluster compatibility:
     See https://www.veritas.com/support/en_US/article.TECH126902
   - For NetBackup 6.x cluster compatibility:
     See https://www.veritas.com/support/en_US/article.TECH43619
   - For NetBackup 5.x cluster compatibility:
     See https://www.veritas.com/support/en_US/article.TECH29272
   - For more on NetBackup compatibility, see
     https://www.veritas.com/support/en_US/dpp.15145.html
2. The steps to cluster a Master Server are different for different versions of NetBackup. See the applicable NetBackup guide for directions.

   https://sort.veritas.com

To verify the robustness of the VCS resources and NetBackup processes

1. Verify that you can online the Netbackup master.
2. Verify that you can offline the Netbackup master.
3. Verify that you can monitor all the NetBackup resources.

Backing up and recovering a VxVM volume using NetBackup

To enable NetBackup to backup objects on a VxVM volume, use the following procedure. This procedure enables an Instant Recovery (IR) using a VxVM volume.

To back up objects in a VxVM volume using NetBackup

1. Create a VxVM disk group with six disks. The number of disks may vary depending on the volume size, disk size, volume layout, and snapshot method.

   If the system this test is running on is a clustered system, create a shared disk group using the `-s` option.

   ```
   # vxdg -s init database_dg disk1 disk2 disk3 \
   disk4 disk5 disk6
   ```

2. Create a "mirror-striped" VxVM volume with a size of 10 Gbytes or the maximum size of the disk, whichever is larger.

   ```
   # vxassist -g database_dg make vol_name 10G \
   layout=mirror-stripe init=active
   # vxvol -g database_dg set fastresync=on vol_name
   # vxassist -g database_dg snapstart nmirror=1 vol_name
   ```

   Note: There are three types of snapshot: mirror, full-size instant, and space-optimized instant shanpshots. The example uses an Instant Recovery (IR) snapshot. For snapshot creation details:

   See pages 104-107 of the *NetBackup Snapshot Client Administrator's Guide* for 7.6.

   See https://www.veritas.com/support/en_US/article.DOC6459

   ---

3. Make the file system on the volume.
4. Mount a VxFS file system on the volume.

   If the VxVM volume is a clustered volume, mount the VxFS file system with the "-o cluster" option.

5. Fill up the VxFS file system up to the desired level. For example, you can fill to 95% full, or to whatever level is appropriate for your file system.
6. Store the `cksum`(1) for these files.
7. Un-mount the VxFS file system.
8. Enable the following Advanced Client option:

   - Perform Snapshot Backup.

- Set **Advanced Snapshot Options** to **vxvm**.
- Enable **Retain snapshots for instant recovery**.

9 Back up the VxVM volume with the NetBackup policy.

See *NetBackup Snapshot Client Administrator's Guide* for 7.6.

See https://www.veritas.com/support/en_US/article.DOC6459

Recovering a VxVM volume using NetBackup

To enable NetBackup to recover objects on a VxVM volume, use the following procedure. This procedure performs an Instant Recovery (IR) using a VxVM volume.

To recover objects in a VxVM volume using NetBackup

1 Initialize the VxVM volume to zeros.

2 Recover the VxVM volume to the newly initialized VxVM volume.

3 Mount the VxFS file system on the empty VxVM volume.

4 Verify the cksum(1) values against the files recovered.

Backing up an Oracle database using NetBackup

To backup an Oracle tablespace using NetBackup, use the following procedure.

To backup an Oracle tablespace using NetBackup's Oracle policy

1 Install and configure NetBackup HA or a single NetBackup server.

2 Configure Oracle RAC based on different layouts

Create an Oacle database, using any number of volumes and any volume layout. Make sure your Oracle datafiles are reside on a VxFS file system.

3 Install the NetBackup client on all Oracle nodes.

4 Create a backup policy. For example, the policy might be for performing Oracle tablespace backup using Flashsnap.

- Policy type: Oracle
- Snapshot method: Flashsnap

Note: Snapshot method is optional for Oracle backup.

- Alternate client: a host sharing data with oracle host

Note: Alternate client is optional for Oracle backup.

If you use the alternate client option, you must have a host that is sharing storage with the Oracle host.

- Schedules: Default-Application-Backup and Automatic Full Backup
- Clients : Oracle host
- Selections: using RMAN shell scripts to backup tablespace
- To configure an Oracle backup policy:
  See the *NetBackup for Oracle Administrator's Guide for UNIX, Windows, and Linux* for 7.1.
  See https://www.veritas.com/support/en_US/article.DOC3672

5. To prepare your volumes for snapshots:

   See the *NetBackup Snapshot Client Administrator's Guide* for 7.1.

   See https://www.veritas.com/support/en_US/article.DOC3661

6. Use the policy to back up the database during a database insert operation.

Recovering an Oracle database using NetBackup

To recover an Oracle tablespace using NetBackup, use the following procedure.

To backup and recover an Oracle tablespace using NetBackup's Oracle policy

1. Fail over the NetBackup cluster and restore it.
2. Recover the tablespace to the original node using RMAN scripts.

   To write the rman script for the Oracle restore:

   See the *NetBackup for Oracle Administrator's Guide for UNIX, Windows, and Linux* for 7.1.

   See https://www.veritas.com/support/en_US/article.DOC3672
3. Take the tablespace online.
4. Verify the restored database.
5. Verify the number of rows in the table.

Section 6

Optimizing storage costs for Oracle

- Chapter 21. Understanding storage tiering with SmartTier
- Chapter 22. Configuring and administering SmartTier
- Chapter 23. SmartTier use cases for Oracle
- Chapter 24. Compressing files and databases to optimize storage costs
- Chapter 25. Using the Compression Advisor tool

Chapter 21

Understanding storage tiering with SmartTier

This chapter includes the following topics:

- About SmartTier
- How SmartTier for Oracle works
- SmartTier in a High Availability (HA) environment

About SmartTier

SmartTier matches data storage with data usage requirements. After data matching, the data can then be relocated based upon data usage and other requirements determined by the storage or database administrator (DBA).

As more and more data is retained over a period of time, eventually, some of that data is needed less frequently. The data that is needed less frequently still requires a large amount of disk space. SmartTier enables the database administrator to manage data so that less frequently used data can be moved to slower, less expensive disks. This also permits the frequently accessed data to be stored on faster disks for quicker retrieval.

Tiered storage is the assignment of different types of data to different storage types to improve performance and reduce costs. With SmartTier, storage classes are used to designate which disks make up a particular tier. There are two common ways of defining storage classes:

- Performance, or storage, cost class: The most-used class consists of fast, expensive disks. When data is no longer needed on a regular basis, the data can be moved to a different class that is made up of slower, less expensive disks.

- Resilience class: Each class consists of non-mirrored volumes, mirrored volumes, and n-way mirrored volumes.

  For example, a database is usually made up of data, an index, and logs. The data could be set up with a three-way mirror because data is critical. The index could be set up with a two-way mirror because the index is important, but can be recreated. The redo and archive logs are not required on a daily basis but are vital to database recovery and should also be mirrored.

SmartTier is a VxFS feature that enables you to allocate file storage space from different storage tiers according to rules you create. SmartTier provides a more flexible alternative compared to current approaches for tiered storage. Static storage tiering involves a manual one- time assignment of application files to a storage class, which is inflexible over a long term. Hierarchical Storage Management solutions typically require files to be migrated back into a file system name space before an application access request can be fulfilled, leading to latency and run-time overhead. In contrast, SmartTier allows organizations to:

- Optimize storage assets by dynamically moving a file to its optimal storage tier as the value of the file changes over time
- Automate the movement of data between storage tiers without changing the way users or applications access the files
- Migrate data automatically based on policies set up by administrators, eliminating operational requirements for tiered storage and downtime commonly associated with data movement

---

Note: SmartTier is the expanded and renamed feature previously known as Dynamic Storage Tiering (DST).

SmartTier policies control initial file location and the circumstances under which existing files are relocated. These policies cause the files to which they apply to be created and extended on specific subsets of a file systems's volume set, known as placement classes. The files are relocated to volumes in other placement classes when they meet specified naming, timing, access rate, and storage capacity-related conditions.

In addition to preset policies, you can manually move files to faster or slower storage with SmartTier, when necessary. You can also run reports that list active policies, display file activity, display volume usage, or show file statistics.

SmartTier leverages two key technologies included with Veritas InfoScale Storage Foundation Enterprise products: support for multi-volume file systems and automatic policy-based placement of files within the storage managed by a file system. A multi-volume file system occupies two or more virtual storage volumes and thereby enables a single file system to span across multiple, possibly heterogeneous,

physical storage devices. For example the first volume could reside on EMC Symmetrix DMX spindles, and the second volume could reside on EMC CLARiiON spindles. By presenting a single name space, multi-volumes are transparent to users and applications. This multi-volume file system remains aware of each volume's identity, making it possible to control the locations at which individual files are stored. When combined with the automatic policy-based placement of files, the multi-volume file system provides an ideal storage tiering facility, which moves data automatically without any downtime requirements for applications and users alike.

In a database environment, the access age rule can be applied to some files. However, some data files, for instance are updated every time they are accessed and hence access age rules cannot be used. SmartTier provides mechanisms to relocate portions of files as well as entire files to a secondary tier.

To use SmartTier, your storage must be managed using the following features:

- VxFS multi-volume file system
- VxVM volume set
- Volume tags
- SmartTier management at the file level
- SmartTier management at the sub-file level

About VxFS multi-volume file systems

Multi-volume file systems are file systems that occupy two or more virtual volumes. The collection of volumes is known as a volume set, and is made up of disks or disk array LUNs belonging to a single Veritas Volume Manager (VxVM) disk group. A multi-volume file system presents a single name space, making the existence of multiple volumes transparent to users and applications. Each volume retains a separate identity for administrative purposes, making it possible to control the locations to which individual files are directed.

This feature is available only on file systems meeting the following requirements:

- The minimum disk group version is 140.
- The minimum file system layout version is 7 for file level SmartTier.
- The minimum file system layout version is 8 for sub-file level SmartTier.

To convert your existing VxFS system to a VxFS multi-volume file system, you must convert a single volume to a volume set.

See "Configuring SmartTier for Oracle" on page 257.

The VxFS volume administration utility (fsvoladm utility) can be used to administer VxFS volumes. The fsvoladm utility performs administrative tasks, such as adding,

removing, resizing, encapsulating volumes, and setting, clearing, or querying flags on volumes in a specified Veritas File System.

See the `fsvoladm` (1M) manual page for additional information about using this utility.

About VxVM volume sets

Volume sets allow several volumes to be represented by a single logical object. Volume sets cannot be empty. All I/O from and to the underlying volumes is directed via the I/O interfaces of the volume set. The volume set feature supports the multi-volume enhancement to Veritas File System (VxFS). This feature allows file systems to make best use of the different performance and availability characteristics of the underlying volumes. For example, file system metadata could be stored on volumes with higher redundancy, and user data on volumes with better performance.

About volume tags

You make a VxVM volume part of a placement class by associating a volume tag with it. For file placement purposes, VxFS treats all of the volumes in a placement class as equivalent, and balances space allocation across them. A volume may have more than one tag associated with it. If a volume has multiple tags, the volume belongs to multiple placement classes and is subject to allocation and relocation policies that relate to any of the placement classes.

Warning: Multiple tagging should be used carefully.

A placement class is a SmartTier attribute of a given volume in a volume set of a multi-volume file system. This attribute is a character string, and is known as a volume tag.

SmartTier file management

SmartTier enables administrators of multi-volume VxFS file systems to manage the placement of files on individual volumes in a volume set by defining placement policies that control both initial file location and the circumstances under which existing files are relocated. These placement policies cause the files to which they apply to be created and extended on specific subsets of a file system's volume set, known as placement classes. The files are relocated to volumes in other placement classes when they meet the specified naming, timing, access rate, and storage capacity-related conditions.

File-based movement:

- The administrator can create a file allocation policy based on filename extension before new files are created, which will create the datafiles on the appropriate tier during database creation.
- The administrator can also create a file relocation policy for database files or any types of files, which would relocate files based on how frequently a file is used.

SmartTier sub-file object management

SmartTier enables administrators of multi-volume VxFS file systems to manage the placement of file objects as well as entire files on individual volumes.

Using sub-file based movement you can:

- Move a set of ranges of a specified set of files of a specified set of mounts to a desired set of tiers on command.
- Move segments of files using automation to:
  - Monitor a set of files for collecting I/O statistics
  - Periodically collect and persist the statistics, cluster-wide if applicable
  - Periodically enforce the ranges of the registered sets of files based on their relative frequency of access to a desired set of tiers
  - Track the historical movements of those ranges

How SmartTier for Oracle works

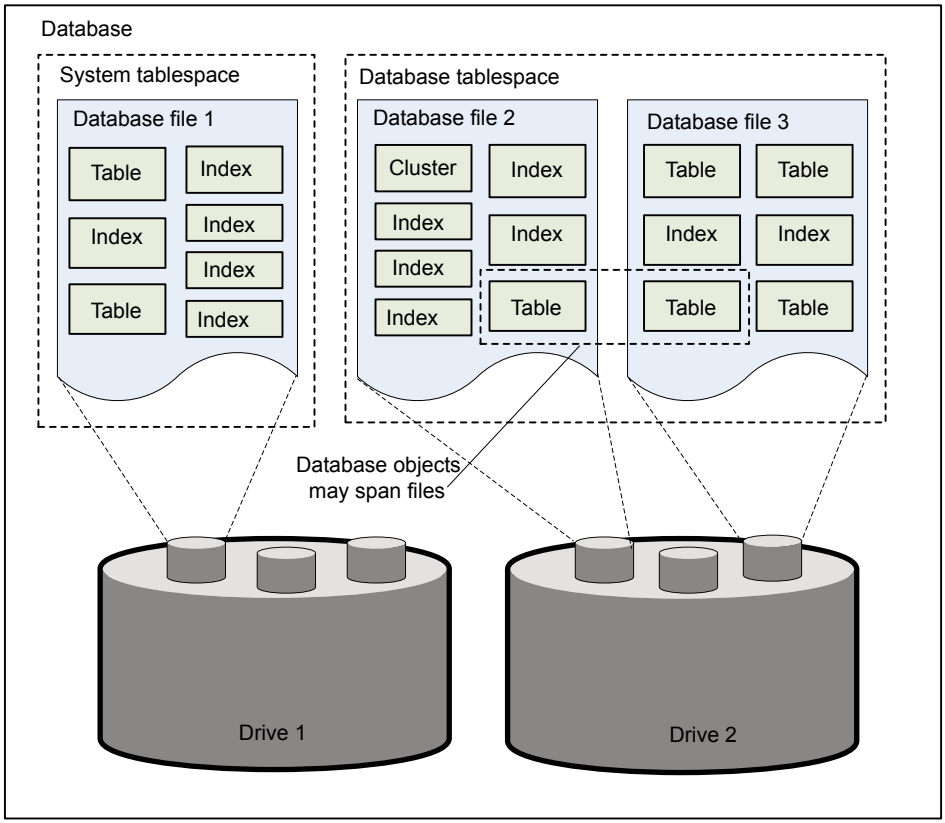
In an Oracle database environment, the access age rule can be applied to archivelog files and Flashback files. Oracle updates the header of each datafile at every database checkpoint and hence access age rules cannot be used for datafiles. For a partitioned table, we can use the name base rule to relocate files belonging to a given partition, for instance last year, to the secondary storage tier. However if a database does not have partitioned tables, current methods for relocation do not fit to the Oracle database environment. To understand how to optimize file relocation for Oracle databases, we need to study how Oracle stores objects in the database.

How partitions change the way Oracle stores database objects

Oracle Database stores data logically in tablespaces and physically in datafiles associated with the corresponding tablespace. A database is divided into one or more logical storage units called tablespaces. A tablespace in an Oracle database consists of one or more physical datafiles. A datafile can be associated with only one tablespace and only one database. Tablespaces are divided into logical units

of storage called segments, which are further divided into extents. Extents are a collection of contiguous blocks in a datafile. Tables are the basic unit of data storage in an Oracle database. Data is stored in rows and columns. Tables are defined with a table name (such as employees) and set of columns. Usually a table resides within a single tablespace except for partitioned tables. A partitioned table is a table which has one or more partitions, and each partition may reside on a different tablespace.

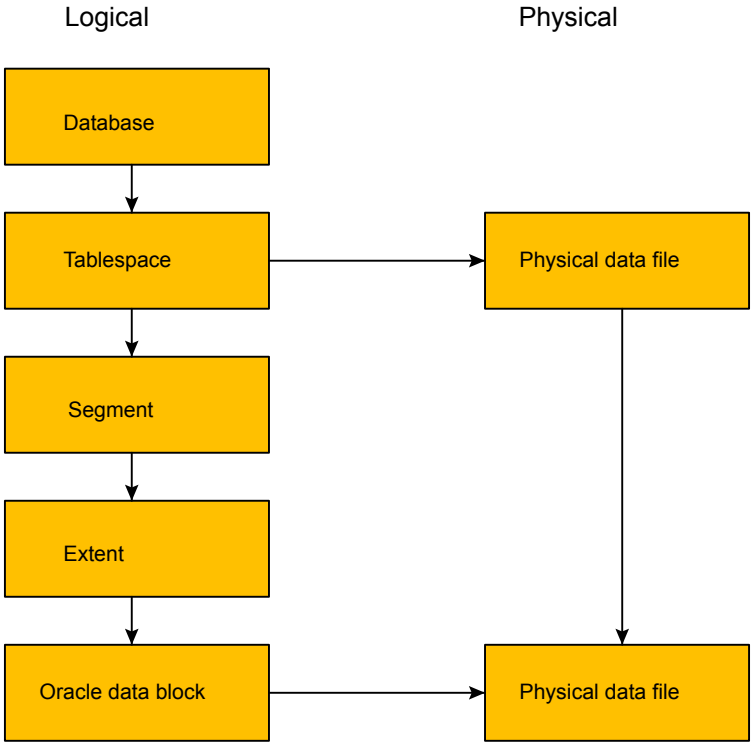
Figure 21-1 How Oracle stores database objects



At the finest level of granularity, an Oracle database stores data in data blocks (also called logical blocks, Oracle blocks, or pages). One data block corresponds to a specific number of bytes of physical database space on disk. The next level of logical database space is an extent. An extent is a specific number of contiguous data blocks allocated for storing a specific type of information from a given datafile in a tablespace. The level of logical database storage greater than an extent is called a segment. A segment is a set of extents, each of which has been allocated

for a specific data structure and all of which are stored in the same tablespace. For example, each table's data is stored in its own data segment, while each index's data is stored in its own index segment. If the table or index is partitioned, each partition is stored in its own segment. An Oracle database allocates space for segments in units of one extent. When the existing extents of a segment are full, an Oracle database allocates another extent for that segment. Because extents are allocated as needed, the extents of a segment may or may not be contiguous on disk. A segment and all its extents are stored in one tablespace. Within a tablespace, a segment can include extents from more than one file; that is, the segment can span datafiles. However, each extent can contain data from only one datafile. When a new extent is allocated for a segment, Oracle tries to spread (or stripe) extents among all datafiles in the tablespace at that time. When you run out of free data blocks in a tablespace, you can add additional datafiles. Here is a pictorial representation of logical objects and physical storage.

Figure 21-2 Logical objects on storage



Oracle has several types of segments. Depending on the type, the usage pattern of the data within the segment varies. For Storage Tiering relocation, it is important

to know what kind of segments is suitable for relocation. Here is a list of segment types.

- Data segments: contain data blocks where table rows are stored. Data segments/extents are prime candidates for relocation to a secondary storage tier. The old extents contain rows inserted earlier than new extents. Segment and extent properties are available by querying dictionary table dba_segments and dba_extents.
- Index segments: contain blocks for table indexes. It is important to database performance that I/O to these blocks are read/written at highest rate and hence should always be in the primary storage tier. Index segments move from one location to other in a tablespace/datafile and are not good candidates for relocating to secondary tier.
- Rollback segment: contain UNDO data which is critical for database recovery and should always be in on primary storage.
- Temp segment: the temporary workspace for intermediate stages of sort operations required by Oracle. Performance depends on I/O time of the temp segments and hence these segments must always be in primary storage.

Database extents to file location

Logical objects like tables are stored as database extents. These database extents are contiguous file system blocks within a given file. Dividing a file into and equal number of objects does not cleanly match database extents. This is the reason why the statistics feature of Veritas File System may not work for an Oracle database. A file object may span more than one database extents. Oracle has catalog entries which will help to convert a database extents to (file, offset, length) set. If we get (file, offset, length) set, we could use our relocation API to relocate that extent to a secondary tier.

Oracle Catalog tables dba_extents and dba_segments can be used to compute (file, offset, length) set for a given extent.

For a given dba_segment, we could join dba_segments and dba_extents to list the (extent-id, file_id, starting_bloc/offset, length) set. This will tell, the exact location in the file of a database extent.

Statistics in Oracle

Oracle collects various statistics and stores them inside a database repository called the Automatic Workload Repository (AWR). The statistics collected include CPU, memory, file IO, and disk statistics. AWR is useful to find out usage trends, bottlenecks and improve performance. By default, the statistics snapshot is taken every 60 minutes and stored in AWR. AWR keeps the snapshot for seven days by

default. Both statistics interval and retention period can be changed via init.ora. Oracle recommends a retention period of 30 days for statistics. Oracle provides tools to collect extra statistics on demand. There are various scripts and tools available to query the AWR and generate trend reports. AWR can be used to get file usage trends that will identify least-used files that can then be relocated to secondary storage.

Advantages of using file statistics from AWR:

- AWR file stats are available by default. No extra configuration is needed.
- AWR file stats are the statistics generated by the database only. File stats from Veritas File System include both database transaction and other IO like backup/restore.
- In a clustered Oracle RAC environment, AWR file stats include I/O from all the nodes. There is no need collect from each node and sum it up.
- AWR file statistics provides history and hence it is easy to get the usage trends.

AWR also provides Oracle object level statistics like segment statistics. These can be used to find out the least used table segments.

One large disadvantage of Oracle file statistics is that they do not provide statistics at less than the file level.

Optimizing storage for Oracle database objects

To adopt a successful SmartTier policy and save storage costs, a method is needed for relocating Oracle objects which are accessed infrequently to the secondary tier. Relocating the entire file is not useful in all cases.

Only data segments are suitable for relocation, and they may be spread among several datafiles. When new rows are inserted in a table, Oracle allocates space for the table on demand. Oracle allocates space through one or more extents.

As a result of this allocation scheme, a given datafile has:

- Multiple extents belonging to more than one table.
- The extents belonging to a table may not be contiguous in the file.
- Each datafile will have a header and headers are updated at every database checkpoint.
- Relocating a file to a secondary storage tier means relocating rows/data belonging to one or more tables.
- If an entire datafile is relocated to a secondary storage tier, the Oracle checkpoint performance may suffer and there will be some performance loss.

When a tablespace is locally managed:

- The free space in a datafile is managed using a bitmap which is kept right next to datafile header.
- Each bit in the bitmap represents 1MB of storage in the file.
- Free space management using bitmaps is far superior to dictionary managed datafiles.
- Several extents are allocated from the same file such that 1MB of the datafile is used by extents. In other words, each table will have 1MB of storage allocated from a given datafile and all these extents are contiguous in the datafile.

This makes using SmartTier for Oracle to tier storage at the file portion level practical.

Table 21-1 Recommended SmartTier option for Oracle use cases

Oracle file type	Recommended SmartTier option
Archive logs	File-based storage tiering
Flashback logs	File-based storage tiering
Partitions	File-based storage tiering
Table relocation	Sub-file based storage tiering
Hot datafile	Sub-file based storage tiering
Portions to SSD	Sub-file based storage tiering

About Automatic Workload Repository statistics

Starting with Oracle 10g, Oracle collects various statistics and stores it inside a database repository called Automatic Workload Repository (AWR).

Statistics include:

- CPU
- Memory
- File I/O
- disk statistics

AWR is useful for finding usage trends, bottlenecks and improve performance.

By default, AWR snapshots are:

- Taken every 60 minutes
- Stored in AWR
- Kept for seven days

Statistics interval and retention period can be changed via init.ora. Oracle recommends a retention period of 30 days for statistics and provides tools to collect extra statistics on demand. There are various scripts and tools available to query the AWR and generate trend reports. You can use AWR to get file usage trends and relocate least used files to secondary storage tier.

Advantages of using file statistics from AWR:

- AWR file stats are available by default. No extra configuration is needed.
- In a clustered environment, AWR file stats include IO from all the nodes. There is no need collect from each node and sum it up.
- AWR file statistics provides history and hence it is easy to get the usage trends.

AWR also provides Oracle Object level statistics like segment statistics. These can be used find out the least used table segments.

See "Using AWR statistics to identify sub-file objects for potential relocation" on page 289.

SmartTier in a High Availability (HA) environment

Cluster Server does not provide a bundled agent for volume sets. If issues arise with volumes or volume sets, the issues can only be detected at the DiskGroup and Mount resource levels.

The DiskGroup agent brings online, takes offline, and monitors a Veritas Volume Manager (VxVM) disk group. This agent uses VxVM commands. When the value of the StartVolumes and StopVolumes attributes are both 1, the DiskGroup agent onlines and offlines the volumes during the import and deport operations of the disk group. When using volume sets, set StartVolumes and StopVolumes attributes of the DiskGroup resource that contains the volume are set to 1. If a file system is created on the volume set, use a Mount resource to mount the volume set.

The Mount agent brings online, takes offline, and monitors a file system or NFS client mount point.

If you are using any of the SmartTier for Oracle commands in a high availability (HA) environment, the time on each system in the cluster must be synchronized. Otherwise, the scheduled task may not be executed at the expected time after a service group failover.

For additional information, see the *Cluster Server Bundled Agents Reference Guide*.

Chapter 22

Configuring and administering SmartTier

This chapter includes the following topics:

- Configuring SmartTier for Oracle
- Optimizing database storage using SmartTier for Oracle
- Running reports using SmartTier for Oracle
- Extent balancing in a database environment using SmartTier for Oracle
- Running sub-file database object reports using SmartTier for Oracle
- Optimizing sub-file database object placement using SmartTier for Oracle

Configuring SmartTier for Oracle

To use SmartTier for Oracle, the following requirements must be met:

- An Oracle database must be up and running.
- Only the Oracle database administrator can run SmartTier for Oracle commands.

To configure SmartTier for Oracle, the following tasks must be performed:

- Review the SmartTier for Oracle command requirements.
- Define the database parameters.
- Set up storage classes.
- Convert an existing Veritas File System database file system to a Veritas File System multi-volume file system for use with SmartTier for Oracle.

- Classify, or tag, volumes so that the tags indicate the quality of the underlying disk.
- Display the free space on each class.
- Add or remove volumes as necessary.

Note: SmartTier for Oracle is the expanded and renamed feature known as SmartTier for Oracle in previous releases.

SmartTier for Oracle command requirements

Before defining your database parameters, review the following command requirements:

- Run the `dbed_update` command before running any of the SmartTier for Oracle commands. You should also run the `dbed_update` command if any of the database files change.
  The repository must be up to date, since the SmartTier for Oracle commands retrieve database information from the repository.
- You do not need to set the environment variable LD_LIBRARY_PATH to use the SFDB commands. However, if you set this environment variable for another reason, Veritas recommends including the library path /opt/VRTSdbed/common/lib before other library paths so the SFDB commands do not mistakenly link with libraries with same name in the other library paths.
  You do not need to set the environment variable LIBPATH to use the SFDB commands. However, if you set this environment variable for another reason, Veritas recommends including the library path /opt/VRTSdbed/common/lib before other library paths so the SFDB commands do not mistakenly link with libraries with same name in the other library paths.
- If you are using any of the SmartTier for Oracle commands in a high availability (HA) environment, the time on each system in the cluster must be synchronized.
- Create the volumes that you want to add to the multi-volume file system in the same disk group as the file system volume. As root, use the following command to change the owner of each volume:

  ```
  # /opt/VRTS/bin/vxedit -g disk_group \
  set user=oracle volume
  ```

- Change the owner of the mount point on which you want to implement SmartTier for Oracle to oracle.

Defining database parameters

Running the `dbdst_admin` command defines parameters for the entire database. You must run this command at least once to define the database parameters for SmartTier for Oracle. Three pre-defined storage classes will be created (PRIMARY, SECONDARY, and BALANCE). Parameter values are stored in the SFDB repository.

Set at least one of the parameters in `maxclass`, `minclass`, `statinterval`, `sweeptime`, `sweepinterval`, `purgetime`, or `purgeinterval`, to enable default values. Add at least one class to enable the default classes.

Table 22-1 lists the options for the `dbdst_admin` command:

Table 22-1 dbdst_admin command options

Command option	Description
`-S $ORACLE_SID`	Specifies the `ORACLE_SID`, which is the name of the Oracle instance.
`list`	Lists all of the SmartTier for Oracle parameters of the database, including class name and description. This option should be used exclusively from the other options.
`maxclass`	Maximum number of storage classes allowed in the database. The default value is 4.
`minclass`	Minimum number of storage classes allowed in the database. The default value is 2.
`sweepinterval`	Interval for file sweeping for file relocation. Default value is 1, which means one per day. If this value is set to 0, all scheduled sweep tasks will become unscheduled.
`sweeptime`	Time per day for the file sweep to take place. Times are entered in 24-hour periods and should list hour: minute. For example, 8:30 AM is represented as 08:30 and 10:00 PM is represented as 22:00. Default value is 22:00.
`statinterval`	Interval in minutes for gathering file statistics. Default value is 30, which represents every 30 minutes. If this value is set to 0, all scheduled tasks will become unscheduled.

Table 22-1 dbdst_admin command options *(continued)*

Command option	Description
`purgeinterval`	Number of days after which the file statistics in the repository will be summarized and purged. Default value is 30. It is recommended that you set your purge interval sooner because you will not be able to view any statistics until the first 30-day interval is over, if you use the default.
`purgetime`	Time per day for the file purge to take place. Times are entered in 24-hour periods and should list hour: minute. For example, 8:30 AM is represented as 08:30 and 8:00 PM is represented as 20:00. Default value is 20:00.
`addclass`	Parameter that allows you to add a class to a database. The information should be entered as class:"description", where the class represents the class name and description is a string of up to 64 characters enclosed by double quotes used to describe the class.
`rmclass`	Parameter that allows you to remove a class from a database. Enter the class name as it appears in the database.
`-o definechunk` *`classname`*: `{128k|256k|512k|1m}`	Defines a chunksize in bytes for the given storage class. Valid chunksizes are 128k, 256k, 512k or 1m bytes. When a chunksize is specified for a storage class, the files in this storage class will be extent-balanced. Each chunk of the file will be in a separate volume of the storage class. A given file will have approximately equal number of chunks on each component volumes of the storage class. When a new volume is added or an existing volume is removed from the storage class (using dbdst_addvol or dbdst_rmvol), the files are automatically balanced again.

Note: If you do not want to change specific default values, you can omit those parameters when you run the dbdst_admin command. You only need to enter the parameters that need to be changed.

To define database parameters

- Use the `dbdst_admin` command as follows:

```
dbdst_admin -S $ORACLE_SID -o setup-parameters\
[,storage_class operations]
        setup-parameters
                maxclass=number,minclass=number,\
                        statinterval=minutes
                sweeptime=HH:MM,sweepinterval=days
                purgetime=HH:MM,purgeinterval=days
        storage_class operations
                addclass=classname:"description"
                rmclass=classname
                definechunk=classname:128k | 256k | 512k | 1m
dbdst_admin -S $oracle_sid -o list
```

For example, to add a class called tier1 for database PROD, and to set up a purge interval of one, meaning that the file statistics will be gathered for one day and then summarized and purged, use the dbdst_admin command as follows:

```
$ /opt/VRTS/bin/dbdst_admin -S PROD -o addclass=tier1:"Fast Storage",\
purgeinterval=1
```

Configuring storage classes

When you define your database parameters, three pre-defined storage classes are created. You must add or remove storage classes to meet your needs.

In addition to the default storage classes, you can add storage classes to better manage your data. Before adding a storage class, review the following information:

To add storage classes

- Use the `dbdst_admin` command as follows:

```
$ /opt/VRTS/bin/dbdst_admin -S $ORACLE_SID -o addclass=classname:\
"description"
```

For example, to create a storage class named "FAST" for an EMC array, use the `dbdst_admin` command as follows:

```
$ /opt/VRTS/bin/dbdst_admin -S $ORACLE_SID -o addclass=FAST:\
"fast EMC array"
```

If you no longer require a specific storage class, you can remove it. Before removing a storage class, review the following information:

Note: You cannot remove the pre-defined storage classes (PRIMARY, SECONDARY, and BALANCE).

To remove a storage class

- Use the `dbdst_admin` command as follows:

  ```
  $ /opt/VRTS/bin/dbdst_admin -S $ORACLE_SID -o rmclass=classname
  ```

  For example, to remove a storage class called "SLOW," use the dbdst_admin command as follows:

  ```
  $ /opt/VRTS/bin/dbdst_admin -S $ORACLE_SID -o rmclass=SLOW
  ```

You can display a list of SmartTier for Oracle properties and storage classes using the `dbdst_admin` command.

Before displaying your storage classes, review the following information:

To display storage classes

- Use the `dbdst_admin` command as follows:

  ```
  $ /opt/VRTS/bin/dbdst_admin -S $ORACLE_SID -o list
  ```

  For example:

  ```
  $ /opt/VRTS/bin/dbdst_admin -S VRTS11r2 -o list
  TSDB Parameters for VRTS11r2
  -----------------------------------
  Maxclass        =  4
  Minclass        =  2
  Stat-Interval   =  30
  Sweep Information = 22HH:0MM, Frequency = 1
  Purge Information = 20HH:0MM, Frequency = 1


  TSDB Storage Classes for VRTS11r2
  -----------------------------------
  Name = BALANCE : Description = BALANCE Storage Class
  Name = PRIMARY : Description = PRIMARY Storage Class
  Name = SECONDARY : Description = SECONDARY Storage Class
  Name = TIER1 : Description = Fast Storage
  ```

Converting a Veritas File System (VxFS) to a VxFS multi-volume file system

To convert your existing VxFS file system to a VxFS multi-volume file system, you must convert a single volume to a volume set.

When you convert to a volume set using the `dbdst_convert` command, the original volume will be renamed to a new volume name. The mount device name will become the new volume set name. Creating the new volume set name with the mount device name nullifies the need to rename the mount device in various locations.

To convert a single volume to a volume set

1. Before converting to a volume set, make sure the following conditions have been met:

Prerequisites	■ The Oracle database must not be active. ■ Create at least one additional volume. ■ You must unmount the source filesystem hence database must be shutdown.
Usage Notes	■ You must convert the single-volume file system on which you plan to implement SmartTier for Oracle. ■ The file system has to be unmounted when you run the `dbdst_convert` command. ■ If the file system has n volumes, volumes 1 through n–1 will be placed in the storage class "PRIMARY" and volume n will be placed in the storage class "SECONDARY." ■ The volumes specified when running the conversion must be in the same disk group as the mount device. ■ The file system must be unmounted on all nodes.

2. Use the `dbdst_convert` command as follows:

```
$ /opt/VRTS/bin/dbdst_convert -S $ORACLE_SID -M mount_device -v \
volume_name,volume_name
```

3. Bring the database objects online.

For example, to convert the volume-based file system oradata to a SmartTier for Oracle-ready volume set file system on mount device /dev/vx/dsk/oradg/oradata, use the `dbdst_convert` command as follows:

```
$ /opt/VRTS/bin/dbdst_convert -S PROD -M \
/dev/vx/dsk/oradg/oradata -v new_vol1,new_vol2
```

After conversion, you will have a volume set named oradata containing three volumes (oradata_b4vset, new_vol1, and new_vol2). The file system will have two storage classes defined as PRIMARY and SECONDARY. The volumes will be assigned as follows:

- PRIMARY storage class will contain volumes oradata_b4vset and new_vol1.
- SECONDARY storage class will contain volume new_vol2.

Classifying volumes into a storage class

Before creating a SmartTier policy or manually moving data, assign classes to your volumes.

To classify a volume

1. Before assigning classes to volumes, review the following information:

Usage notes	■ You must convert your Veritas File System file system to a multi-volume file system first. ■ Storage classes must be registered using the `dbdst_admin` command before assigning classes to volumes. ■ The database can be online or offline.

2. Use the `dbdst_classify` command as follows:

```
$ /opt/VRTS/bin/dbdst_classify -S $ORACLE_SID -M mount_device \
-v volume_name:class[,volume_name:class]
```

For example, to assign the class "FAST" to volume new_vol1, use the `dbdst_classify` command as follows:

```
$ /opt/VRTS/bin/dbdst_classify -S $ORACLE_SID \
-M /dev/vx/dsk/oradg/oradata -v new_vol1:FAST
```

Displaying free space on your storage class

To see the free space, class information, and volume information on your storage classes, use the `dbdst_show_fs` command.

Table 22-2 shows the `dbdst_show_fs` command options.

Table 22-2 dbdst_show_fs command options

Command options	Description
`-S $ORACLE_SID`	Specifies the `ORACLE_SID`, which is the name of the Oracle instance.
`-o volume`	Displays the free space on volumes in each class.
`-m`	Specifies the mount point.

To display the free space on a storage class

1 Before displaying the free space on a storage class, review the following information:

Prerequisites

- Make sure the file system is mounted.
- See the `dbdst_show_fs` (1M) manual page.

2 Use the `dbdst_show_fs` command as follows:

```
dbdst_show_fs -S $ORACLE_SID -m mount_point
```

For example:

```
$ dbdst_show_fs -S VRTS11r2 -m /data11r2

MAXCLASS=4
MINCLASS=2
TOTAL CLASS=3

VOLUME NAME                    SIZE      USED    AVAILABLE
-----------                    ----      ----    ---------

CLASS=FAST
datavol1                       1024000   1024000 0

CLASS=MEDIUM
datavol2                       4608000   2347872 2260128

CLASS=PRIMARY
datavol-b4vset                 10240000  3585514 6654486
```

```
dbdst_show_fs -S $ORACLE_SID -m mount_point -o volume
```

For example:

```
$ dbdst_show_fs -S VRTS11r2 -m /data11r2 -o volume

VOLUME NAME                    SIZE      USED    AVAILABLE  TAG
-----------                    ----      ----    ---------  --
datavol-b4vset                 10240000  3585514 6654486    PRIMARY
datavol1                       1024000   1024000 0          FAST
datavol2                       4608000   2347872 2260128    MEDIUM
```

Adding new volumes to a storage class

Use the `dbdst_addvol` command to add volumes to a volume set.

To add a volume to a volume set

1 Before adding a volume, review the following information:

Usage notes	■ The database must be inactive when adding volumes to a storage class. ■ The database file system has to be mounted.

2 Use the `dbdst_addvol` command as follows:

```
$ /opt/VRTS/bin/dbdst_addvol -S $ORACLE_SID -M mount_device \
-v volume_name:class[,volume_name:class]
```

For example:

```
$ /opt/VRTS/bin/dbdst_addvol -S VRTS11r2 \
-M /dev/vx/dsk/dstdatadgot/datavol -v datavol4:FAST
```

Removing volumes from a storage class

You may need to remove a volume from a volume set. To remove a volume, use the `dbdst_rmvol` command.

1 Before removing a volume, review the following information:

Usage notes	■ The database must be inactive when removing volumes from a storage class. ■ Only a volume that does not contain any file system data can be removed.

2 Use the `dbdst_rmvol` command as follows:

```
$ /opt/VRTS/bin/dbdst_rmvol -S $ORACLE_SID -M mount_device \
-v volume_name[,volume_name]
```

For example:

```
$/opt/VRTS/bin/dbdst_rmvol -S VRTS11r2 \
-M /dev/vx/dsk/dstdatadgot/datavol  -v datavol6
```

Optimizing database storage using SmartTier for Oracle

SmartTier for Oracle provides multiple tools to optimize your storage. For instance, you can choose to manually relocate files or tablespaces, or you can use a preset SmartTier policy.

Note: You must issue commands as an Oracle database administrator in the following procedures.

SmartTier for Oracle options for improving storage performance

- Relocating files manually
- Relocating tablespaces manually
- Relocating table partitions manually
- Setting policies to automate storage tiering
- Running reports
- Extent balancing
- Running sub-file database database object reports
- Optimizing sub-file database object placement

Relocating files manually using SmartTier for Oracle

Table 22-3 shows the `dbdst_file_move` command options.

Table 22-3 dbdst_file_move command options

Command options	Description
`-o archive[n] | flashback`	Specifies which archive logs or Flashback logs to move. Do not use this option with the `-f` option.
`-o external | datafile`	Specifies whether to move external files or datafiles. Use this option with the -f option.
`-f listfile`	Specifies a listfile that contains a list of files or directories to be moved.

Table 22-3 dbdst_file_move command options *(continued)*

Command options	Description
`-c storage_class [:days]`	Specifies the storage class to which the files should be moved. If the days option is used, the files will be moved to the class specified if they have not been accessed in the number of days specified. Do not specify days if you are using the -o datafile option.
`-R`	Removes the policy for the specified object.

To relocate a file manually

1 Before relocating a file, review the following information:

Usage notes	Multiple partitions cannot reside on the same tablespace.

2 Use the `dbdst_file_move` command as follows:

```
$ /opt/VRTS/bin/dbdst_file_move -S $ORACLE_SID -o datafile \
-f listfile -c storage_class:days \
[-c storage_class:days]
```

For example:

```
$ /opt/VRTS/bin/dbdst_file_move -S VRTS11r2  -o datafile \
-f /tmp/external.spec -c MEDIUM

Assign placement policy file successful on filesystem /data11r2
Begin enforcing filesystem /data11r2 ...
Enforce placement policy successful on filesystem /data11r2
```

Relocating tablespaces manually using SmartTier for Oracle

Use the `dbdst_tbs_move` command to move tablespaces to the desired storage class. The command queries the SFDB repository for the tablespace file names, then performs a one-time move based on your immediate requirements.

To relocate a tablespace

◆ Use the `dbdst_tbs_move` command as follows:

```
$ /opt/VRTS/bin/dbdst_tbs_move -S $ORACLE_SID -t tablespace \
-c storage_class
```

where

- *tablespace* indicates which tablespace to move.
- *storage_class* indicates to which class the tablespace should be moved.

For example:

```
$ /opt/VRTS/bin/dbdst_tbs_move -S VRTS11r2  -t MYTBS  -c MEDIUM

Assign placement policy file successful on filesystem /data11r2
Begin enforcing filesystem /data11r2 ...
Enforce placement policy successful on filesystem /data11r2
```

Relocating table partitions manually using SmartTier for Oracle

Use the `dbdst_partition_move` to move table partitions. The command queries the database to validate the names of the table and partition. From this information, a list of datafiles is derived and a one-time move of the files to the desired class is executed.

Before relocating table partitions, review the following information:

Prerequisites	The database must be up when you run the `dbdst_partition_move` command.

To relocate a table partition

◆ Use the `dbdst_partition_move` command as follows:

```
$ /opt/VRTS/bin/dbdst_partition_move -S $ORACLE_SID -T table_name \
-p partition_name -c storage_class
```

where

- `-T` indicates the table name.
- `-p` indicates the partition name.
- `-c` indicates the class to which the table partition is to be moved.

For example, to move the SALES_Q1 partition of the SALES table to storage class SLOW, use the `dbdst_partition_move` as follows:

```
$ /opt/VRTS/bin/dbdst_partition_move -S $ORACLE_SID -T SALES \
-p SALES_Q1 -c SLOW
```

Another example:

```
$ /opt/VRTS/bin/dbdst_partition_move -S VRTS11r2  -T EMP1  -p P3  \
     -c MEDIUM

Assign placement policy file successful on filesystem /data11r2
Begin enforcing filesystem /data11r2 ...
Enforce placement policy successful on filesystem /data11r2
```

Setting policies to automate storage tiering using SmartTier for Oracle

Use the `dbdst_preset_policy` command to set a policy based on file name patterns before the files are created.

Table 22-4 shows the preset policies command options.

Table 22-4 dbdst_present_policy command options

Command option	Description
`-d` directory	Indicates the directory on which the placement policy will be applied.
`-e`	Enforces the file system of the specified directory. Use this option if there was an error in the previous enforcement that has been corrected and needs to be enforced again.
`-R`	Removes all pattern-based placement policies related to this directory.
`-l`	Lists the existing file placement that is set to the specified directory.
`-P` pattern_spec	Specifies file patterns and class assignment. This option will automatically place files in the desired class as soon as they are created. Existing files and newly created files will be moved immediately to the class specified.
`-f` pattern file	Specifies a file that contains a particular class and pattern. New files with this pattern will be placed in the class immediately. Existing files will be moved as well.
`-E`	Specifies that existing files should be moved to the designated class in a one-time move to be scheduled at a later time, such as the sweeptime specified in the `dbdst_admin` command.

To create a preset policy

- Use the `dbdst_preset_policy` command as follows:

```
$ dbdst_preset_policy -S $oracle_sid -d directory \
[ -e | -R | -l | -P pattern_spec | -f pattern_file ] [-E]
```

where pattern_spec has the format of "class=pattern,pattern,...[:class=pattern,pattern,...]"

For example:

```
$ /opt/VRTS/bin/dbdst_preset_policy -S VRTS11r2  \
     -d /data11r2/VRTS11r2 -P "MEDIUM=*.dbf:FAST=*.log"

Assign placement policy file successful on filesystem /data11r2
Begin enforcing filesystem /data11r2 ...
Enforce placement policy successful on filesystem /data11r2
```

Running reports using SmartTier for Oracle

You can create a report that lists all modified allocation policies or you can view an audit report, which lists recent relocation changes for a specific date range resulting from your policies.

To create a list of modified allocation policies, use the `dbdst_report` command with the `policy` option.

To view modified allocation policies

- Use the `dbdst_report` command to view a list of modified allocation policies.

  For example:

```
$ /opt/VRTS/bin/dbdst_report -S $ORACLE_SID -o policy
```

```
$ /opt/VRTS/bin/dbdst_report -S  VRTS11r1 -o policy
TSDB Database Policy Report
------------------------------------------------------------

Plan Name = preset_dir:/data11r1/VRTS11r1
Plan Type = PRESET POLICY DIRECTORY
Rule Name =  DBED_precreate_ino4_PRIMARY
Directory=/dev/vx/dsk/dstdatadg09/datavol : VRTS11r1
Rule Description =  PRIMARY=MEDIUM=*.log,undo*.dbf

Plan Name = tbs_move:MYTBS
Plan Type = TBS MOVE
Rule Name =  DBED_relocate_once_ino4_MEDIUM
Directory=/dev/vx/dsk/dstdatadg09/datavol : VRTS11r1
Rule Description =  MEDIUM
```

To view an audit report, which lists recent file relocation changes within a specific date range, use the `dbdst_report` command with the `audit` option.

To view audit reports

- Use the `dbdst_report` command as follows:

```
$ /opt/VRTS/bin/dbdst_report -S $ORACLE_SID -o audit \
startdate=yyyy-mm-dd,enddate=yyyy-mm-dd
```

For example, to view an audit report of changes from January 1, 2007 through March 1, 2007, use the `dbdst_report` command as follows:

```
$ /opt/VRTS/bin/dbdst_report -S $ORACLE_SID -o audit \
startdate=2007-01-01,enddate=2007-03-01
```

Another example:

```
$ /opt/VRS/bin/dbdst_report -S VRTS11r2 -o audit \
startdate=2007-01-01,enddate=2010-08-13

AUDIT DATETIME            AUDIT DESCRIPTION
-------------------------------------------------------------
2010-08-13 23:38:48       ADMIN;Added;class name=TIER1
2010-08-13 23:38:48       ADMIN;Modified Successfully;max/min class=4/2,
 \statinterval=30, sweeptime=22:0, purgetime=20:0, swp/prg freq=1/1
2010-08-13 23:46:48       ADMIN;Added;class name=FAST
2010-08-13 23:46:58       ADMIN;Added;class name=FAST
```

Extent balancing in a database environment using SmartTier for Oracle

To obtain better performance in a database environment, you would normally use a volume striped over several disks. As the amount of data stored in the file system increases over time, additional space in the form of new disks must be added.

To increase space, you could perform a volume relayout using the command. However, changing a large volume, for example, from a four-way striped volume to six-way striped volume involves moving old block information into temporary space and writing those blocks from the temporary space to a new volume, which would require an extended amount of time. To solve this problem, SFDB tools provide the Extent Balanced File System or EBFS .

An Extent Balanced File System is created on a multi-volume file system where individual volumes are not striped over individual disks. For data-availability, these individual volumes can be mirrored. The file system on the EBFS has a special placement policy called a balance policy. When the balance policy is applied, all

the files are divided in`vxrelayout`to small "chunks" and the chunks are laid out on volumes so that adjacent chunks are on different volumes. The default chunk size is 1MB and can be modified. Since every file contains chunks on all available volumes, it is important that individual volumes that make up the EBFS and volume set be of same size and same access properties.

Setting up the file system in this way provides the same benefit as striping your volumes.

Note: You cannot convert an existing file system to an EBFS file system.

Extent balancing file system to imrove database performance

You can define allocation policies with a balance allocation order and "chunk" size to files or a file system, known as extent balancing. The chunk size is the maximum size of any extent that files or a file system with this assigned policy can have. The chunk size can only be specified for allocation policies with a balance allocation order.

An extent balancing policy specifies the balance allocation order and a non-zero chunk size. The balance allocation order distributes allocations randomly across the volumes specified in the policy and limits each allocation to a maximum size equal to the specified chunk size.

Extent balancing extends the behavior of policy enforcement by rebalancing extent allocations such that each volume in the policy is as equally used as possible. Policy enforcement handles the following cases:

- New volumes are added to the policy, and the extents associated with a file need rebalancing across all volumes, including the new ones.
- Volumes are removed from the volume set or from the policy, and the extents for a file residing on a removed volume need to be moved to other volumes in the policy.
- An extent balancing policy is assigned to a file and its extents have to be reorganized to meet the chunk size requirements defined in the policy.

The extent balancing policy is intended for balancing data extents belonging to files across volumes defined in the policy. However, there is no restriction imposed in assigning extent balancing policy for metadata.

Note: If the fixed extent size is less than the chunk size, then the extent size will be limited to the largest multiple of the fixed extent size that is less than the chunk size. If the fixed extent size is greater than the chunk size, then the extent size will be the fixed extent size.

Creating an extent balanced file system using SmartTier for Oracle

Any multi-volume file system (MVFS) can become an extent balanced file system, if the storage tier has a chunk size associated with the class. The `dbdst_admin` command permits the user to define a chunk size for the class.

For example, the following `dbdst_admin` commands define chunk sizes for the gold and silver storage classes:

```
$ /opt/VRTS/bin/dbdst_admin -S $ORACLE_SID \
-o definechunk=gold:256K

$ /opt/VRTS/bin/dbdst_admin -S $ORACLE_SID \
-o definechunk=silver:128K
```

For example:

```
$ /opt/VRTS/bin/dbdst_admin -S $ORACLE_SID \
-o definechunk=TIER1:256K
```

The above commands make storage class gold as extent balanced.

Once the chunksize is defined for a storage tier, we can classify any MVFS into this storage tier.

For example, assume that /oradata is the filesystem created on volume-set /dev/vx/dsk/oradg/ora_vset, and contains database datafiles. Let us further assume that datafile names end with extension *.dbf. To define storage class in this MVFS, the following `dbdst_classify` command is used:

```
$ /opt/VRTS/bin/dbdst_classify  -S $ORACLE_SID \
-M /dev/vx/dsk/oradg/ora_vset -v  vol1:GOLD,vol2:GOLD,vol3:GOLD
```

For example:

```
$ /opt/VRTS/bin/dbdst_classify -S VRTS11r2 \
-M /dev/vx/dsk/dstdatadgot/datavol -v datavol1:FAST,datavol2:MEDIUM
```

It is important to note that, an MVFS can have multiple storage tiers and that each tier may have a different chunk size. For example, for the same MVFS in the above example, we can define another storage tier using the `dbdst_classify` command:

```
$/opt/VRTS/bin/dbdst_classify -S $ORACLE_SID \
-M /dev/vx/dsk/oradg/ora_vset -v vol4:silver,vol5:silver
```

For example:

```
$ /opt/VRTS/bin/dbdst_classify -S VRTS11r2 \
-M /dev/vx/dsk/dstarchdgot/archvol -v archvol1:FAST,archvol2:MEDIUM
```

At this point we have two storage tiers in MVFS /oradata each having different chunksizes. To create the real extent balance, we need to assign a SmartTier policy and to enforce it.

To define and enforce the policy, you could use the following `dbdst_preset_policy` command:

```
$/opt/VRTS/bin/dbdst_preset_policy -S $ORACLE_SID \
-d /oradata -P GOLD=*.dbf:SILVER=*.inx
```

The above example creates a SmartTier policy, assigns the policy to /oradata and enforces the policy. All datafiles of the form *.dbf will be extent balanced in GOLD tier with chunksize 256K and all index files of the form *.inx will be extent balanced in SILVER tier with chunk size 128K.

Another example:

```
$ /opt/VRTS/bin/dbdst_preset_policy -S VRTS11r2  \
-d /data11r2/VRTS11r2 -P "MEDIUM=temp*.dbf:FAST=*.log"

Assign placement policy file successful on filesystem /data11r2
Begin enforcing filesystem /data11r2 ...
Enforce placement policy successful on filesystem /data11r2
```

To view the space usage in the /oradata MVFS use the `dbdst_show_fs` command. For example:

```
$ /opt/VRTS/bin/dbdst_show_fs -S $ORACLE_SID -m /oradata
```

For example:

```
$ /opt/VRTS/bin/dbdst_show_fs -S VRTS11r2 -m /data11r2 -o volume

VOLUME NAME                    SIZE      USED    AVAILABLE       TAG
-----------                    ----      ----    ---------       ---
datavol-b4vset                 10240000  3585514 6654486         PRIMARY
datavol1                       1024000   1024000 0               FAST
datavol2                       4608000   2347872 2260128         MEDIUM
```

When the GOLD or SILVER tier requires more space, we could add extra space by adding new volumes to the respective storage tier using the `dbdst_addvol` command. For example:

```
$ /opt/VRTS/bin/dbdst_addvol -S $ORACLE_SID \
-M /dev/vx/dsk/oradg/ora_vset  -v vol7:GOLD
```

As soon as you add a new volume, the SmartTier policy is enforced and the extents are balanced over the new volume too. This can be viewed by using the `dbdst_show_fs` command again.

To view detailed extent information about a given file, you can use the `fsmap` command. For example:

```
$ /opt/VRTSvxfs/sbin/fsmap -a /oradata/emp.dbf
```

```
$ /opt/VRTS/bin/fsmap -a  /oradata/emp.dbf
```

```
$ fsmap -a /standby/oradata1_1/slave/system01.dbf |more
 Volume  Extent Type  File Offset  Extent Size   File
  vol1      Data           0      262144      /standby/oradata1_1/slave/system01.dbf
  vol2      Data      262144        8192      /standby/oradata1_1/slave/system01.dbf
  vol3      Data      270336      131072      /standby/oradata1_1/slave/system01.dbf
  vol1      Data      401408      262144      /standby/oradata1_1/slave/system01.dbf
  vol2      Data      663552      180224      /standby/oradata1_1/slave/system01.dbf
  vol3      Data      843776       32768      /standby/oradata1_1/slave/system01.dbf
  vol1      Data      876544       49152      /standby/oradata1_1/slave/system01.dbf
  vol2      Data      925696       65536      /standby/oradata1_1/slave/system01.dbf
  vol3      Data      991232      196608      /standby/oradata1_1/slave/system01.dbf
```

Running sub-file database object reports using SmartTier for Oracle

Use the `dbdst_obj_view` command to get detailed specific database, index, or partition information. The report indicates in which storage class each database table/index extent resides. The reports provide information for decisions to move the database objects within different storage clases of the multi-volume filesystem.

An Oracle tablespace is usually constructed of several datafiles. A database object such as a table or an index has extents allocated from multiple datafiles. A given datafile usually contains extents from multiple database objects.

The `dbdst_obj_view` command queries Oracle dba_extents catalog table and for each extent gets file offset and length in bytes. This command passes the offset

and length information to the File System low level SmartTier commands to get the exact location of the extent blocks.

Usage for `dbdst_obj_view`

```
dbdst_obj_view -S $ORACLE_SID -H $ORACLE_HOME -t table [-s start_extent] [-e end_extent]
dbdst_obj_view -S $ORACLE_SID -H $ORACLE_HOME -i index [-s start_extent] [-e end_extent]
dbdst_obj_view -S $ORACLE_SID -H $ORACLE_HOME -p partition [-s start_extent] [-e end_extent]
dbdst_obj_view -S $ORACLE_SID -H $ORACLE_HOME -f datafile
```

Prerequisites
- The database must be online.
- The command must be run as Oracle user.

Notes
- To make sure the SFDB repository is up-to-date, run `dbed_update` at least once before using the command.
- Generating a report for large tables will take long time.
- To restrict the scope of the table/index, use start/end extents.

For Oracle RAC environments
- This command can be run on any node of an Oracle RAC cluster on which the database is running.

Table 22-5 shows the sub-file report command options.

Table 22-5 dbdst_obj_view command options

Command option	Description
`-S ORACLE_SID`	Specifies the name of the Oracle database from which information will be retrieved.
`-H ORACLE_HOME`	Specifies the ORACLE_HOME for the database.
`-t table name`	Name of the table for which view report needs to be generated.
`-i index name`	Name of the index for which view report needs to be generated.
`-p partition name`	Name of the partition for which view report needs to be generated.
`-s start-extent`	Specifies the starting extent of the table/index. The default starting extent is zero.
`-e end-extent`	Specifies the last extent of the table/index . If this option is not specified, then the report will be generated to include last extent of the table.

To verify objects have been moved as expected to a new tier

- Use following command to verify a relocation of the entire emp table to SYMMETRIX, extents 100 to 400 of table emp to SSD, and partition q1_orders to SSD.

```
$ /opt/VRTS/bin/dbdst_obj_view -S PROD -H /opt/oracle/home11g \
-t emp

File Name            Ext-Id   File-Offset    Length    Tier-Name
---------              ------   -----------     ------    ---------
/demo_data/emp1.dbf       1      8462336     1048576    SYMMETRIX
/demo_data/emp1.dbf       2      2170880     1048576    CLARION
/demo_data/emp1.dbf       3      3219456     1048576    CLARION
/demo_data/emp1.dbf       4      9510912     1048576    CLARION
/demo_data/emp1.dbf       5     12656640     1048576    CLARION
/demo_data/emp1.db        6     13705216     1048576    SYMMETRIX
/demo_data/emp1.db        7     14753792     1048576    SYMMETRIX
/demo_data/emp1.dbf       8     15802368     1048576    SYMMETRIX
/demo_data/emp1.db              16850944     1048576    SYMMETRIX
```

Optimizing sub-file database object placement using SmartTier for Oracle

Use the `dbdst_obj_move` command to move the specified database table, index or a partition files to the desired storage class. The `dbdst_obj_move` command queries Oracle dba_extents catalog table and for each extent gets file offest and lengh in bytes. This command passes these offset/length information to VxFS low level SmartTier commands to move the database extents to desired storage class.

Usage for `dbdst_obj_move`

```
dbdst_obj_move -S $ORACLE_SID -H $ORACLE_HOME [-v] -t table \
     [-s start_extent] [-e end_extent] -c storage_class
dbdst_obj_move -S $ORACLE_SID -H $ORACLE_HOME [-v] -i index \
     [-s start_extent] [-e end_extent] -c storage_class
dbdst_obj_move -S $ORACLE_SID -H $ORACLE_HOME [-v] -p partition \
     [-s start_extent] [-e end_extent] -c storage_class
```

Prerequisites

- The database must be online.
- The command must be run as Oracle user.

Notes

- To make sure the SFDB repository is up-to-date, run `dbed_update` at least once before using the command.

For Oracle RAC environments

- This command can be run on any node of an Oracle RAC cluster on which the database is running.

Table 22-6 shows the sub-file object movement command options.

Table 22-6 dbdst_obj_move command options

Command option	Description
`-S $ORACLE_SID`	Specifies the name of the Oracle database from which information will be retrieved.
`-H $ORACLE_HOME`	Specifies the ORACLE_HOME for the database.
`-v`	This is an optional argument. If this option is specified, a verification will be done after moving each extent to the target storage class. If the target storage class does not have enough free space, some the the table extents may not be moved.
`-t table name`	Name of the table for which view report needs to be generated.
`-i index name`	Name of the index which must be relocated.
`-p partition name`	Name of the partition which must be relocated. More than one partition can co-exiest in the same tablespace.
`-c storage_class`	Specifies the storage class to which the table or index extents will be moved.
`-s start_extent`	Specifies the starting extent of the table or index. The default starting extent is zero.
`-e end_extent`	Specifies the last extent of the table or index . If this option is not specified, then the complete table or index will be moved to target class.

To relocate entire emp table to SYMMETRIX

1. Run the `dbed_update` command to make sure the repository is current.
2. Run the the following command:

```
$ /opt/VRTS/bin/dbdst_obj_move -S PROD -H /opt/oracle/home_dir \
-t emp -c SYMMETRIX
```

To relocate extents 100 to 400 of table emp to SSD

1. Run the `dbed_update` command to make sure the repository is current.
2. Run the following command:

```
$ /opt/VRTS/bin/dbdst_obj_move -S PROD -H /opt/oracle/home_dir \
-t emp -s 100 -e 400 -c SSD
```

To relocate and verify the relocation of partition q1_orders to SSD

1. Run the `dbed_update` command to make sure the repository is current.
2. Run the following command:

```
$ /opt/VRTS/bin/dbdst_obj_move -S PROD -H /opt/oracle/home_dir \
-v -p q1_orders  -c SSD
```

Chapter 23

SmartTier use cases for Oracle

This chapter includes the following topics:

- SmartTier use cases for Oracle
- SmartTier for Oracle file-based use cases
- SmartTier for Oracle sub-file use cases

SmartTier use cases for Oracle

Veritas InfoScale products include SmartTier, a storage tiering feature which enables you to tier your data to achieve optimal use of your storage.

Example procedures illustrate the following use cases:

For file-level SmartTier for Oracle:

- Migrating partitioned data and tablespaces
- Scheduling the relocation of archive and Flashback logs

For sub-file-level SmartTier for Oracle

- Using AWR statistics to identify sub-file objects for potential relocation
- Relocating a hot database segment to higher performance tier
- Relocating old table data to a lower cost tier

SmartTier for Oracle file-based use cases

The following file-based use cases are supported for SmartTier for Oracle:

- Migrating partitioned data and tablespaces
- Scheduling the relocation of archive and Flashback logs

Migrating partitioned data and tablespaces

Perhaps the simplest application of multi-tier storage to databases is relocation of individual table partitions between different placement classes as usage requirements change. If exact relocation times are unpredictable, or if relocation is infrequent, administrators may wish to relocate table partitions when necessary rather than defining strict periodic relocation schedules.

Ad hoc relocation of table partitions can be useful, for example, with databases that track sales and inventory for seasonal businesses such as sports equipment or outdoor furniture retailing. As the selling season for one type of inventory (for example, summer equipment or furniture) approaches, database table partitions that represent in-season goods can be relocated to high-performance storage, since they will be accessed frequently during the coming months. Similarly, partitions that represent out-of-season goods can be relocated to lower-cost storage, since activity against them is likely to be infrequent.

For example, sales are mostly catalog-driven for a large retailer specializing in sports equipment. Product details are saved in a large database and the product table is partitioned based on type of activity. Some of the products are seasonal and do not sell well at other times. For example, very few snow skis are sold during the summer. To achieve season-based migration, see the following example. Assume the table product_tab has two partitions, summer and winter. Each of these partitions is mapped to a separate data file.

First, you must set up your system to use SmartTier for Oracle.

To add the fast_storage and slow_storage storage classes

◆ Use the `dbdst_admin` command as follows:

```
$ /opt/VRTS/bin/dbdst_admin -S PROD -o addclass=\
fast_storage:"Fast Storage for Production DB"

$ /opt/VRTS/bin/dbdst_admin -S PROD -o addclass=\
slow_storage:"Slow Storage for Production DB"
```

To convert the database's file system and add volumes for use with SmartTier for Oracle

◆ Use the `dbdst_convert` command as follows:

```
$ /opt/VRTS/bin/dbdst_convert -S PROD \
-M /dev/vx/dsk/oradg/oradata -v new_vol1,new_vol2,new_vol3
```

To classify volumes into storage classes

- Use the `dbdst_classify` command as follows:

```
$ /opt/VRTS/bin/dbdst_classify -S PROD \
-M /dev/vx/dsk/oradg/oradata -v new_vol1:fast_storage
```

```
$ /opt/VRTS/bin/dbdst_classify -S PROD \
-M /dev/vx/dsk/oradg/oradata -v new_vol2:slow_storage,\
new_vol3:slow_storage
```

Once the volumes are configured, an administrator can define file placement policy rules that specify seasonal relocation of selected tablespaces and partitions and assign them to the database's file system.

To move summer data to slower storage and winter data to faster storage at the beginning of winter

- Use the `dbdst_partition_move` command as follows:

```
$ /opt/VRTS/bin/dbdst_partition_move -S PROD -T product_tab \
-p winter -c fast_storage
```

```
$ /opt/VRTS/bin/dbdst_partition_move -S PROD -T product_tab \
-p summer -c slow_storage
```

These commands relocate the files that comprise the winter partition of the product_tab table to placement class fast_storage, and the files that comprise the summer partition to placement class slow_storage. SmartTier for Oracle determines which files comprise the winter and summer partitions of product_tab, and uses underlying SmartTier services to immediately relocate those files to the fast_storage and slow_storage placement classes respectively.

To move winter data to slower storage and summer data to faster storage at the beginning of summer

- Use the `dbdst_partition_move` command as follows:

```
$ /opt/VRTS/bin/dbdst_partition_move -S PROD -T product_tab \
-p summer -c fast_storage
```

```
$ /opt/VRTS/bin/dbdst_partition_move -S PROD -T product_tab \
-p winter -c slow_storage
```

SmartTier for Oracle formulates SmartTier policy rules that unconditionally relocate the files containing the target partitions to the destination placement classes. It merges these rules into the database file system's active policy, assigns the resulting

composite policy to the file system, and enforces it immediately to relocate the subject files. Because the added policy rules precede any other rules in the active policy, the subject files remain in place until the dbdst_partition_move command is next executed, at which time the rules are removed and replaced with others.

Scheduling the relocation of archive and Flashback logs

As are the primary mechanism for recovering from data corruption, database logs are normally kept on premium storage, both for I/O performance and data reliability reasons. Even after they have been archived, logs are normally kept online for fast recovery, but the likelihood of referring to an archived log decreases significantly as its age increases. This suggests that archived database logs might be relocated to lower-cost volumes after a certain period of inactivity.

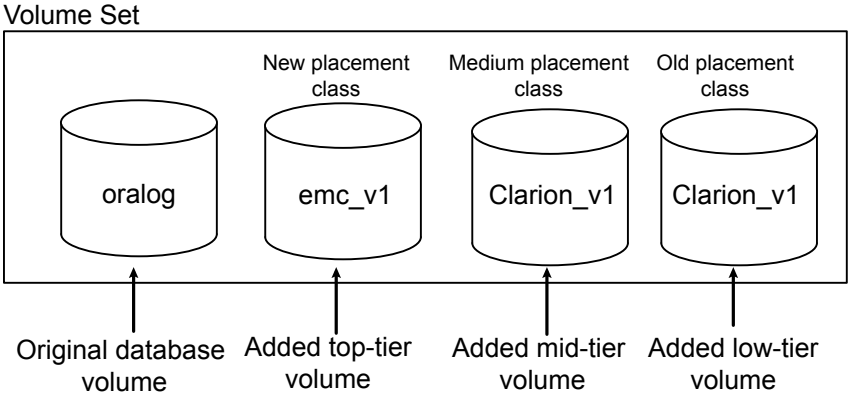
Similarly, Storage Foundation Flashback technology creates logs that can be used for quick recovery from database corruption by restoring a database to its state at a previous time. Flashback logs are normally kept for a shorter period than archived database logs, if used at all, they are typically used within a few hours of creation. Two or three days are a typical Flashback log lifetime.

The rapidly decaying probability of use for archive and Flashback logs suggests that regular enforcement of a placement policy that relocates them to lower-cost storage after a period of inactivity can reduce an enterprise's average cost of online storage.

For example procedures, a customer could be using a large OLTP Oracle database with thousands of active sessions, which must be up and running 24 hours a day and seven days a week with uptime of over 99%. The database uses Flashback technology to correct any accidental errors quickly and generates a large number of archive logs per day. If the database goes down for any reason, there is a business requirement to bring the database back online and get it functional with in 15 minutes. To prevent Oracle log switch delays during transactions, the archive logs need to be created in a fast EMC array. Archive logs older than a week can be moved to a mid-range Clarion array. Archive logs older than 15 days can be moved to slow JBOD disks. Archive logs are purged after 30 days. Current Flashback logs are created manually by the database administrator on fast EMC storage and can be moved to Clarion storage after two days. The database administrator then deletes the Flashback logs after a week. To set up a system like this, see the following example. Assume that archive logs and Flashback logs are created on the same file system, /oralog. On the file system, /oralog/archive1 contains archive logs and /oralog/flashback contains Flashback logs.

Figure 23-1 illustrates a three-tier volume configuration that is suitable for automatic relocation and deletion of archive logs and Flashback logs.

Figure 23-1 Database storage configuration suitable for automatic relocation of archive and Flashback logs



The file system used by the production database in this example originally resides on the single volume oralog, which must be prepared by adding volumes and placement classes assigned to the volumes.

To add the NEW, MEDIUM, and OLD storage classes

- Use the `dbdst_admin` command as follows:

```
$ /opt/VRTS/bin/dbdst_admin -S PROD -o addclass=\
NEW:"EMC Storage for Production DB"

$ /opt/VRTS/bin/dbdst_admin -S PROD -o addclass=\
MEDIUM:"Clarion Storage for Production DB"

$ /opt/VRTS/bin/dbdst_admin -S PROD -o addclass=\
OLD:"JBOD Storage for Production DB"
```

To convert the database's file system and add volumes for use with SmartTier for Oracle

- Use the `dbdst_convert` command as follows:

```
$ /opt/VRTS/bin/dbdst_convert -S PROD \
-M /dev/vx/dsk/oradg/oralog -v emc_v1,clarion_v1,jbod_v1
```

To classify volumes into storage classes

- Use the `dbdst_classify` command as follows:

```
$ /opt/VRTS/bin/dbdst_classify -S PROD \
-M /dev/vx/dsk/oradg/oralog -v emc_v1:NEW

$ /opt/VRTS/bin/dbdst_classify -S PROD \
-M /dev/vx/dsk/oradg/oralog -v clarion_v1:MEDIUM

$ /opt/VRTS/bin/dbdst_classify -S PROD \
-M /dev/vx/dsk/oradg/oralog -v jbod_v1:OLD
```

Once the volumes are configured, an administrator can define file placement policy rules that specify access age-based relocation of selected files and assign them to the database's file system.

To define rules that periodically relocate Flashback and archive logs

- Use the `dbdst_file_move` command as follows:

```
$ /opt/VRTS/bin/dbdst_file_move -S PROD -o flashback -c MEDIUM:2
```

This command relocates files in the Flashback directory that have not been accessed for two days to the MEDIUM volume.

```
$ /opt/VRTS/bin/dbdst_file_move -S PROD -o archive1 -c MEDIUM:7 \
-c OLD:15
```

This command relocates files in the archive1 directory that have not been accessed for seven days to the MEDIUM volume, and files that have not been accessed for 15 days to the OLD volume.

SmartTier for Oracle translates these commands into SmartTier access age-based policy rules, merges them with the file system's placement policy, and assigns the resulting policy to the file system. By default, SmartTier for Oracle enforces the active policy daily. During enforcement, the new rules relocate qualifying files to the destination storage tiers specified in the `dbdst_file_move` commands used to create the policies.

SmartTier for Oracle sub-file use cases

The following sub-file based use cases are supported for SmartTier for Oracle:

- Using AWR statistics to identify sub-file objects for potential relocation
- Relocating a hot database segment to higher performance tier

- Relocating old table data to a lower cost tier

Using AWR statistics to identify sub-file objects for potential relocation

You can use Oracle's Automatic Workload Repository (AWR) statistics to identify sub-file objects for potential relocation by identifying key properties such as activity levels. The Oracle database collects these statistics for a variety of purposes, including making self-management decisions. See the guidelines below for enabling AWR and generating AWR reports.

To enable AWR statistics collection

◆ Specify the level of collection for database and operating system statistics by setting the initialization parameter: STATISTICS_LEVEL:

- TYPICAL: is the default setting. This setting ensures collection of all major statistics required for database self-management functionality and provides best overall performance. The default value should be adequate for most environments.
- ALL: additional statistics are added to the set of statistics collected with the TYPICAL setting. The additional statistics are timed OS statistics and plan execution statistics.
- BASIC: disables the collection of many of the important statistics required by Oracle database features and functionality.

Veritas recommends setting STATISTICS_LEVEL = typical .This parameter can be dynamically changed when database is online.

To change the STATICS_LEVEL

◆ Use the following:

```
SQL> alter system set statistics_level=typical;
```

An AWR report shows data captured between two snapshots (or two points in time). The AWR reports are divided into multiple sections. The HTML report includes links that can be used to navigate quickly between sections. The content of the report contains the workload profile of the system for the selected range of snapshots.

To generate an HTML or text report for a range of snapshot IDs

1. Run the `awrrpt.sql` script at the SQL prompt:

   ```
   SQL> @$ORACLE_HOME/rdbms/admin/awrrpt.sql
   ```

2. Specify whether you want an HTML or a text report.

   ```
   Enter value for report_type: text
   ```

3. Specify the number of days for which you want to list snapshot IDs.

   ```
   Enter value for num_days: 2
   ```

4. After the list displays, you are prompted for the beginning and ending snapshot Id for the workload repository report.

   ```
   Enter value for begin_snap: 150
   Enter value for end_snap: 160
   ```

5. Accept the default report name or enter a report name. The default name is accepted in the following example:

   ```
   Enter value for report_name:
   Using the report name awrrpt_1_150_160
   ```

   The workload repository report is generated.

You can use the AWR reports to identify candidate tables or indexes for moving to alternative storage. The AWR report has two sections which are useful for sub-file SmartTier. For a directed move of an object to a desired tier, the Segment Statistics portion of the AWR report can be used. Here is an example from TPCC workload. Using the Segment Statistics hot table/index can be moved to SSD tier.

Table 23-1 Sample segments by physical read

Owner	Tablespace name	Object Name	Object type	Physical reads	%Total
TPCC	USERS	ITEM_ID	INDEX	13,551	43.59
TPCC	USERS	ITEM	TABLE	7,718	24.83
TPCC	USERS	CUSTOMER	TABLE	4,403	14.16
TPCC	USERS	STOCK	TABLE	3,692	11.88
TPCC	USERS	ORDERS	TABLE	352	1.13

ITEM_ID and ITEM from the sample segments reported above are the two most likely candidates for relocation to a higher performance tier. The commands to relocate them are:

```
$ dbdst_obj_move -S $ORACLE_SID -H $ORACLE_HOME \
-i ITEM_ID -c SSD_TIER
```

```
$ dbdst_obj_move -S $ORACLE_SID -H $ORACLE_HOME \
-t ITEM -c SSD_TIER
```

For more on using the `dbdst_obj_move` command:

See "Optimizing sub-file database object placement using SmartTier for Oracle" on page 280.

Relocating a hot database segment to a higher performance tier

You can relocate hot database segments or objects to higher performance storage using sub-file storage tiering when relocating the entire datafiles is not practical.

For the example values in the procedure below, see the information in Table 23-1

The command in the example below queries dba_segments and dba_extents and prepares a list of extents for a given table/index. From the extent information, a list containing (file, offset) is generated. This list is handed over to `fsppadm` to perform the sub-file enforcement.

To relocate a hot database segment to a high-performance tier

1. Identify the most active hot objects or segments of the database using an Oracle AWR report.
2. Determine the size needed to relocate them to the higher-performance tier.
3. To ensure an up-to-date SFDB repository, run `dbed_update`.
4. Run the command to move the table emp to the tier *SSD*.

   ```
   $ dbdst_obj_move  -S PROD -H /home_dir -t EMP -c SSD
   ```

The command in the example below displays the set datafiles where the specified table/index resides. If the underlying file system is a multi-volume file system, it will display the corresponding volume and tier name.

To verify that the database segment is relocated to the correct tier

- Run the command to view the relocated table:

  ```
  $ dbdst_obj_view  -S PROD -H /home_dir -t EMP
  ```

For more on using the `dbdst_obj_move` command:

See "Optimizing sub-file database object placement using SmartTier for Oracle" on page 280.

Relocating old table data to a lower cost or secondary tier

In some databases such as TELCO applications, call detail records (CDR) accumulate very quickly. For regulatory purposes, these records must be saved for several years, causing the size of the database to grow at an enormous rate. If the CDR table is partitioned, the old partitions can be relocated to the lower cost tier. However, if the CDR table is not partitioned, alternatives are needed. One potential alternative is described below.

The extents for CDR tables are allocated on demand. This means extent 100 is created much earlier than extent 900. It also means that the activity for CDR records in extent 100 is less. From the extent-id, using dba_extents and dba_segments dictionary tables, file, offset, length information can be generated. From this information, you can relocate the identified portion of the file to the secondary tier.

To relocate old table data to a lower cost or secondary tier

1. Identify the less active objects or segments of the database using an Oracle AWR report.
2. Determine the storage tier to which the less active objects or segments can be relocated.
3. To ensure an up-to-date SFDB repository, run `dbed_update`.
4. Run the command to move the table *CDR_100* to *tier-2*.

```
$ dbdst_obj_move  -S PROD -H /home_dir -t   CDR_100 -c tier-2
```

The command in the example below displays the set datafiles where the specified table/index resides. If the underlying file system is a multi-volume file system, it will display the corresponding volume and tier name.

To verify that the database segment is relocated to the correct tier

- Run the command to view the relocated table:

```
$ dbdst_obj_view  -S PROD -H /home_dir -o CDR_100
```

For more on using the `dbdst_obj_move` command:

See "Optimizing sub-file database object placement using SmartTier for Oracle" on page 280.

Chapter 24

Compressing files and databases to optimize storage costs

This chapter includes the following topics:

- Compressed files and databases

Compressed files and databases

Compressing files helps to reduce the storage cost in a database environment. For Oracle databases, compression provides an excellent value add to reduce storage cost for archived logs, partitioned tables, and infrequently accessed tablespaces and datafiles. The compression ratio of database files depends on the type of object stored in the datafiles. Oracle traditionally stores TABLES and INDEXES in datafiles, in which case the compression ratio depends on type of columns associated with the TABLE and the type of keys in the INDEXES. Oracle also has the ability to store unstructured data, such as XML, spreadsheets, MS Word documents, and pictures, within a TABLE via the Secured Files feature. These types of unstructured data are very good candidates for compression. You can achieve up to 90% compression for archived logs, and about 50% to 65% compression for Oracle datafiles and indexes.

Oracle database files can be compressed and uncompressed as needed while the database is active, although this can have a significant performance impact on the database. Other than reduced I/O response time, compression runs seamlessly while the Oracle database is online and actively doing transactions to the files. Compression works seamlessly with advanced I/O methods, such as direct I/O, asychronous I/O, concurrent I/O, ODM, and Cached ODM. Any updates and new inserts to the datafile result in uncompressing the portion of the file associated with

the write. The queries get uncompressed data in memory and the file remains compressed.

Note: You cannot compress Quick I/O files.

You can run the `vxcompress` command as a DBA user.

The following use cases apply to databases:

- Supported database versions and environment
- Compressing archive logs
- Compressing read-only tablespaces
- Compressing infrequently accessed table partitions
- Compressing infrequently accessed datafiles
- Best practices for compressing files in an Oracle database

Supported database versions and environment

Compression is supported in Storage Foundation (SF), Storage Foundation and High Availability (SFHA), Storage Foundation for Oracle RAC (SF Oracle RAC), and Storage Foundation Cluster File System High Availability (SFCFSHA). In a clustered environment, such as SF Oracle RAC and SFCFSHA, Veritas recommends that you compress files on a node that has minimal load. In a Fast Failover SFCFSHA environment, Veritas recommends that you compress files on a passive node where the database is offline.

Compressing archive logs

Archive logs are critical files required for database recovery. In a busy online transaction processing (OLTP) database, several gigabytes of archive logs are generated each day. Company guidelines often mandate preserving archive logs for several days. The Oracle archive logs are read-only files and are never updated after they are generated. During recovery, Oracle reads archive logs sequentially. As such, archive logs are very good candidates for compression, and archive logs are highly compressible.

The following example procedure compresses all archive logs that are older than a day.

To compress all archive logs that are older than a day

1. As an Oracle DBA, run the following query and get the archive log location:

   ```
   SQL> select destination from v$archive_dest where status = 'VALID'
   and valid_now = 'YES';
   ```

   Assume `/oraarch/MYDB` is the archive log destination.

2. Compress all of the archive logs that are older than a day:

   ```
   $ find /oraarch/MYDB -mtime +1 -exec /opt/VRTS/bin/vxcompress {} \;
   ```

   You can run this step daily via a scheduler, such as `cron`.

Compressing read-only tablespaces

In a large database environment, it is a common practice to keep static tablespaces that do not have any changes in read-only mode. The primary purpose of read-only tablespaces is to eliminate the need to perform backup and recovery of large, static portions of a database. Read-only tablespaces also provide a way to protecting historical data so that users cannot modify it. Making a tablespace read-only prevents updates on all tables and objects residing in the tablespace, regardless of a user's update privilege level. These kinds of read-only tablespaces are excellent candidates for compression. In some cases such as month end reports, there may be large queries executed against these read-only tablespaces. To make the report run faster, you can uncompress the tablespace on demand before running the monthly reports.

In the following example, a sporting goods company has its inventory divided into two tablespaces: `winter_items` and `summer_items`. In the end of the Spring season, you can compress the `winter_item` tablespace and uncompress the `summer_item` tablespace. You can do the reverse actions at end of the Summer season. The following example procedure performs these tasks.

To compress and uncompress tablespaces depending on the season

1. Using SQL, get a list of files in each tablespace and store the result in the files `summer_files` and `winter_files`:

   ```
   SQL> select file_name from dba_data_files where
   tablespace_name = 'WINTER_ITEM';
   ```

   Store the result in the `winter_files` file.

   ```
   SQL> select  file_name from dba_data_files where
   tablespace_name = 'SUMMER_ITEM';
   ```

   Store the result in the `summer_files` file.

2. Compress the `winter_files` file:

   ```
   $ /opt/VRTS/bin/vxcompress `/bin/cat winter_files`
   ```

3. Uncompress the `summer_files` file:

   ```
   $ /opt/VRTS/bin/vxcompress -u `/bin/cat summer_files`
   ```

Compressing infrequently accessed table partitions

Partitioned tables is a frequently used feature for large Oracle databases. Table partitioning improves database queries and updates because partitioning helps parallelizing transactions that use Parallel Queries. Partitioning also makes maintenance of database easy and improves the availability of TABLES. If a partition is down, only the corresponding portion of the TABLE goes offline and the rest of the TABLE remains online. In a telecommunications environment, a common practice is to partition a 'call_details' table by month or quarter. The contents in the partition are less active as the partition gets older. The new records are added to a new partition, and previous quarter records do not get updated. Since telecommunications databases are generally very large, compressing last year's data provides great savings.

In the following example, assume that the table 'CALL_DETAIL' is partitioned by quarters, and the partition names are CALL_2010_Q1, CALL_2010_Q2, and CALL_2011_Q1, and so on. In the first Quarter of 2011, you can compress the CALL_2010_Q1 data.

To compress the CALL_2010_Q1 partition

1. Use SQL to retrieve the filenames belonging to the CALL_2010_Q1 partition:

```
SQL> select tablespace_name from dba_tab_partitions
where table_name = 'CALL_DETAIL' and partition_name = 'CALL_2010_Q1';
```

   Assume that the query returns "TBS_2010_Q1".

2. Store the names in the `my_compress_files` file:

```
SQL> select file_name from dba_data_files where
tablespace_name = 'TBS_2010_Q1';
```

   Store the result in the `my_compress_files` file.

3. Compress the files:

```
$ /opt/VRTS/bin/vxcompress `/bin/cat my_compress_files`
```

Compressing infrequently accessed datafiles

Many customer databases do not use the Oracle partitioning feature. If partitioning is not used, then you can use Oracle catalog queries to identify datafiles that are not very active. Periodically, you can query the catalog tables and identify the least active datafiles and compress those files, as illustrated in the following example procedure.

To identify the least active datafiles and compress those files

1. Query v$filestat and identify the least active datafiles:

```
SQL> select  name, phyrds + phywrts 'TOT_IO' from v$datafile d
and v$filestat f where d.file# = f.file# order by TOT_IO;
```

2. Select files that have the least I/O activity from the report and compress those files:

```
$ /opt/VRTS/bin/vxcompress file1 file2 file3 ...
```

3. Periodically run the query again to ensure that the compressed files do not have increased I/O activity. If I/O activity increases, uncompress the files:

```
$ /opt/VRTS/bin/vxcompress -u file1 file2 file3 ...
```

Best practices for compressing files in an Oracle database

Even though an Oracle database runs without any errors when files are compressed, increased I/O to compressed files decreases database performance. Use the following guidelines for compressing Oracle datafiles:

- Do not compress database control files.
- Do not compress files belonging to TEMPORARY tablespaces.
- Do not compress files belonging to SYSTEM and SYSAUX tablespace.
- Monitor the I/O activity on compressed files periodically and uncompress the files if I/O activity increases.

Chapter 25

Using the Compression Advisor tool

This chapter includes the following topics:

- About the Compression Advisor tool
- Compressing Oracle archive logs using Compression Advisor
- Displaying compression candidate data files using Compression Advisor
- Compressing Oracle data files using Compression Advisor
- Displaying compressed data files using Compression Advisor
- Uncompressing Oracle data files using Compression Advisor
- Compression Advisor command reference

About the Compression Advisor tool

Veritas File System (VxFS) provides the `vxcompress` utility that can be used to compress individual files transparent to the underlying applications. An application reading a compressed file automatically receives the uncompressed data that is uncompressed in memory only; the on disk part of the data remains compressed. If an application writes to a compressed file, parts of the file are uncompressed on disk.

Compression Advisor provides extended compression functionality for Oracle database files in Oracle single instance and Oracle RAC environments. You can use Compression Advisor to:

- Display Oracle data files that can be compressed for storage savings without affecting database performance.

- Compress and uncompress data files.
- Compress archive logs that were generated a specified number of days ago or earlier.
- Display compressed data files.

The Compression Advisor command `sfae_comp_adm` resides in the `/opt/VRTS/bin` directory and it should be run by the DBA user.

Note: If you have executed the FileSnap operations on a database, you cannot use the compression feature on the database.

See "Compression Advisor command reference" on page 305.

See "Compressing Oracle archive logs using Compression Advisor" on page 300.

See "Displaying compression candidate data files using Compression Advisor" on page 301.

See "Compressing Oracle data files using Compression Advisor" on page 302.

See "Displaying compressed data files using Compression Advisor" on page 303.

See "Uncompressing Oracle data files using Compression Advisor" on page 304.

Compressing Oracle archive logs using Compression Advisor

The Oracle archive logs are read-only files and are never updated after they are generated. During recovery, Oracle reads archive logs sequentially. Archive logs are very good candidates for compression, and archive logs are highly compressible.

You can use the Compression Advisor command `sfae_comp_adm` to compress archive logs.

To compress archive logs using Compression Advisor

- Use the following command.

```
$ sfae_comp_adm -o compress_archive_logs -n 2 -d /arch/sfaedb

Compressing file /arch/sfaedb/1_4_773425912.dbf resides on /arch/sfaedb
Compressing file /arch/sfaedb/1_2_767364459.dbf resides on /arch/sfaedb
Compressing file /arch/sfaedb/1_3_773425912.dbf resides on /arch/sfaedb
Compressing file /arch/sfaedb/1_2_773425912.dbf resides on /arch/sfaedb
Compressing file /arch/sfaedb/1_4_767364459.dbf resides on /arch/sfaedb
Compressing file /arch/sfaedb/1_3_767364459.dbf resides on /arch/sfaedb
Compressing file /arch/sfaedb/1_5_773425912.dbf resides on /arch/sfaedb
Compressing file /arch/sfaedb/1_1_773425912.dbf resides on /arch/sfaedb
Compressing file /arch/sfaedb/1_6_773425912.dbf resides on /arch/sfaedb
```

This command compresses all archive logs that were generated two days ago or earlier, and that belong to the destination `/arch/sfaedb`.

See "About the Compression Advisor tool" on page 299.

See "Compression Advisor command reference" on page 305.

Displaying compression candidate data files using Compression Advisor

You can use Compression Advisor to identify Oracle data files that can be compressed without affecting the database performance. Compression Advisor uses Oracle Automatic Workload Repository (AWR) snapshots for this purpose. The data files that show minimal I/O activity in AWR snapshots are recommended for compression.

To display candidate data files using Compression Advisor

◆ Use the following command.

```
$ sfae_comp_adm -o display -t tbs1

Tablespace     Filename                    Amount of      Amount of
                                           Data Read      Data Written
                                           (bytes)        (bytes)
TBS1           /data/sfaedb/tbs1_2.dbf     302            344
TBS1           /data/sfaedb/tbs1_1.dbf     1246           6064
```

This command displays the data files in the tablespace TBS1. The output table is displayed in increasing order of I/O activity such that the data file with the least I/O activity is in the top row.

See "About the Compression Advisor tool" on page 299.

See "Compression Advisor command reference" on page 305.

Compressing Oracle data files using Compression Advisor

You can use Compression Advisor to compress the specified Oracle data files to save storage costs. Use the compression candidate data files information to determine the data files that can be compressed. You can either specify a single data file name for compression or specify a file name with the `-f` option that contains a list of data files to be compressed. If you are using SmartTier, you can specify a tier on which all data files can be compressed.

The following data files are not compressed by Compression Advisor. Any attempts to compress any of these data files result in a warning. If you have specified a set of data files and any of them belongs to one of following categories, those data files are not compressed and a warning is generated.

- Data files that belong to the SYSTEM tablespace
- Data files that belong to the SYSAUX tablespace
- Oracle temp files
- Data files that belong to undo tablespaces
- The files that are not data files for the given Oracle instance
- The files that are not on Veritas File System (VxFS)

To compress data files using Compression Advisor

◆ Use the following command.

```
$ sfae_comp_adm -o compress_datafiles -t ARCHIVE
Finding datafiles on tier ARCHIVE,
please be patient as this may take a while ...
Found on tier ARCHIVE, following datafiles:
/datavset/mytbs5.dbf
/datavset/mytbs4.dbf
/datavset/mytbs1.dbf
/datavset/mytbs2.dbf
/datavset/mytbs3.dbf

Compressing file /datavset/mytbs5.dbf resides on /datavset
Compressing file /datavset/mytbs4.dbf resides on /datavset
Compressing file /datavset/mytbs1.dbf resides on /datavset
Compressing file /datavset/mytbs2.dbf resides on /datavset
Compressing file /datavset/mytbs3.dbf resides on /datavset
```

This command compresses all data files that are on the tier ARCHIVE.

See "About the Compression Advisor tool" on page 299.

See "Compression Advisor command reference" on page 305.

Displaying compressed data files using Compression Advisor

You can use Compression Advisor to display the list of compressed Oracle data files.

To display compressed data files using Compression Advisor

◆ Use the following command.

```
$ sfae_comp_adm -o  display -c
Tablespace     Filename                    Filesize       Compression
                                                          Ratio (%)
TBS1           /data/sfaedb/tbs1_2.dbf     1.01 MB        99.51%
TBS1           /data/sfaedb/tbs1_1.dbf     112 KB         99.11%
USERS          /data/sfaedb/users01.dbf    5.01 MB        78.98%

Total storage saving 5.15 MB (92.54%)
```

This command displays the compressed data files. The final summary displays the amount of storage space saving.

See "About the Compression Advisor tool" on page 299.

See "Compression Advisor command reference" on page 305.

Uncompressing Oracle data files using Compression Advisor

You can use Compression Advisor to uncompress the specified Oracle data files. You can either specify a single data file name for uncompressing or specify a file name with the `-f` option that contains a list of data files to be uncompressed. If you are using SmartTier, you can specify a tier on which all data files can be uncompressed.

To uncompress data files using Compression Advisor

◆ Use the following command.

```
$ sfae_comp_adm -o uncompress_datafiles -t ARCHIVE

Finding datafiles on tier ARCHIVE,
please be patient as this may take a while ...
Found on tier ARCHIVE, following datafiles:
/datavset/mytbs5.dbf
/datavset/mytbs4.dbf
/datavset/mytbs1.dbf
/datavset/mytbs2.dbf
/datavset/mytbs3.dbf

Uncompressing file /datavset/mytbs5.dbf resides on /datavset
Uncompressing file /datavset/mytbs4.dbf resides on /datavset
Uncompressing file /datavset/mytbs1.dbf resides on /datavset
Uncompressing file /datavset/mytbs2.dbf resides on /datavset
Uncompressing file /datavset/mytbs3.dbf resides on /datavset
```

This command uncompresses all data files that are on the tier ARCHIVE.

See "About the Compression Advisor tool" on page 299.

See "Compression Advisor command reference" on page 305.

Compression Advisor command reference

The `sfae_comp_adm` command syntax is illustrated as follows and its options are listed in Table 25-1.

```
sfae_comp_adm [ -S ORACLE_SID ] [ -H ORACLE_HOME ]
-o compress_archive_logs [ -n N ] [ -d DESTINATION ]

sfae_comp_adm [ -S ORACLE_SID ] [ -H ORACLE_HOME ]
-o { compress_datafiles | uncompress_datafiles }
DATAFILE1 [ DATAFILE2 ... ]

sfae_comp_adm [ -S ORACLE_SID ] [ -H ORACLE_HOME ]
-o { compress_datafiles | uncompress_datafiles }
-f LISTFILE

sfae_comp_adm [ -S ORACLE_SID ] [ -H ORACLE_HOME ]
-o { compress_datafiles | uncompress_datafiles }
```

```
-t TIER

sfae_comp_adm [ -S ORACLE_SID ] [ -H ORACLE_HOME ]
-o display [ -n N ] [ -t TABLESPACE ] [ -c ]
```

Table 25-1 Compression Advisor command options

Option	Use
`-S` *`ORACLE_SID`*	Specify the ORACLE_SID, which is the SID of the Oracle database to be used. You can set this value in the environment by using the environment variable *`ORACLE_SID`*.
`-H` *`ORACLE_HOME`*	Specify the ORACLE_HOME, which is the HOME of the Oracle instance to be used. You can set this value in the environment by using the environment variable *`ORACLE_HOME`*.
`-o compress_archive_logs`	Compress the specified archive logs.
`-d` *`DESTINATION`*	Used with `-o compress_archive_logs`. Compress archive logs generated from the specified destination. If your database is configured to archive to multiple destinations, archive logs on all destinations are compressed, by default.
`-o compress_datafiles`	Compress the specified data files.
`-o uncompress_datafiles`	Uncompress the specified data files.
`-f` *`LISTFILE`*	Used with `-o compress_datafiles` or `-o uncompress_datafiles` . Specify a file that contains the names of the data files to be compressed or uncompressed, with each data file name on a single line.
`-o display`	Display the list of candidate data files that can be compressed.

Table 25-1 Compression Advisor command options *(continued)*

Option	Use
`-t` *`TIER`* `-t` *`TABLESPACE`*	■ When using with `-o compress_datafiles` or `-o uncompress_datafiles`: Specify the storage tier of which all data files are to be compressed or uncompressed. ■ When using with `-o display`: Specify the tablespace of which all data files that are candidates for compression are to be displayed.
`-n` *`N`*	■ When using with `-o compress_archive_logs`: Compress archive logs generated N days ago or older. If you specify N as 0, all archive logs are compressed. The default value is 1. ■ When using with `-o display`: Specify the number of files that can be displayed. The default value is 10.
`-c`	Used with `-o display`. Display the list of compressed data files.

See "About the Compression Advisor tool" on page 299.

Section 7

Managing Oracle disaster recovery

- Chapter 26. Using Storage Foundation for Databases (SFDB) tools in a replicated environment
- Chapter 27. Using volume snapshots in a replicated environment

Chapter 26

Using Storage Foundation for Databases (SFDB) tools in a replicated environment

This chapter includes the following topics:

- About deploying SF Databases tools with Volume Replicator (VVR)
- Using database accelerators for the Oracle database in a Volume Replicator (VVR) environment
- About deploying Storage Foundation for Databases (SFDB) tools with Oracle Data Guard

About deploying SF Databases tools with Volume Replicator (VVR)

Storage Foundation for Databases (SFDB) extends the strengths of the Storage Foundation Enterprise products with database-specific enhancements to offer performance, availability, and manageability for Oracle database servers. SFDB tools provide the means for:

- For AIX, and Solaris:
  Enhanced IO methods such as Quick IO, Cached Quick IO, ODM, and Cached ODM, which enhance database performance
  For Linux:

Enhanced IO methods such as ODM, and Cached ODM, which enhance database performance

- Taking point-in-time copies filesystem checkpoints and Volume Snapshots
- Creating a clone database from volume snapshots
- Performing a rollback of the database from a given filesystem checkpoint
- Performing storage tiering using SmartTier for Oracle

Volume Replicator (VVR) is an option of Veritas Volume Manager (VxVM) that works as a fully integrated component of VxVM. VVR benefits from the robustness, ease of use, and high performance of VxVM, and at the same time, adds replication capability to VxVM.

VVR is able to:

- Replicate existing VxVM configurations
- Be transparently configured while the application is active
- Replicate application writes on the volumes at the source location to one or more remote locations across any distance
- Provide a consistent copy of application data at the remote locations

If a disaster occurs at the source location, you can use the copy of the application data at the remote location and restart the application at the remote location. This section describes how to use SFDB tools at a remote VVR site.

For more information about how to setup and configure VVR replication for Oracle database environment:

See the *Veritas InfoScale 7.4.2 Replication Administrator's Guide*.

Using database accelerators for the Oracle database in a Volume Replicator (VVR) environment

Database acclerators are fast I/O methods used for better database performance:

- Oracle Disk Manager (ODM)
- Cached Oracle Disk Manager (CODM)
- Quick I/O (QIO)
- Cached Quick I/O (CQIO)

When using SFDB tools in a global environment replicated by Volume Replicator (VVR):

- Using database accelerators for Oracle at the remote site does not require replicating the SFDB repository and hence there is no need to replicate the repository.
- It is not necessary to use the same I/O method at the primary and the remote sites. However if you need the same application performance at the remote site you may want to use same I/O method at remote site.
- There is no need to run SFDB commands qio_getdbfiles and qio_convertdbfiles because the Quick IO files are replicated.

Best practices for using database accelerators with Oracle:

- Make sure you use same version of Oracle at the primary and the remote site.
- Make sure you enable ODM by linking the proper ODM library at the remote site.
- Quick IO files will be replicated as Quick IO files at the remote site and hence no extra configuration is required. However, if you are using Cached Quick IO, and you would like to use Cached Quick IO at the remote site, then you need to change the mount properties using the `qioadmin` command and also need to copy the /etc/vx/qioadmin file from the primary to the remote.

Note: Quick I/O and Cached Quick I/O are not supported on Linux.

About deploying Storage Foundation for Databases (SFDB) tools with Oracle Data Guard

The SFDB tools functionality is extended to support standby databases in an Oracle Data Guard environment.

Oracle Data Guard terminology:

- Primary Database: Also referred to as ‘Production Database’, this database is accessed by most of the user applications.
- Standby Database: A standby database is a transactionally consistent copy of the primary database.
- Data Guard Configuration: A Data Guard configuration consists of one production database and one or more standby databases.

- Physical Standby Database: A physically identical copy of the primary database, with on disk database structures that are identical to the primary database on a block-for-block basis.

The following point-in-time copy methods can be used on a Physical Standby database:

- Database Storage Checkpoints
- Third mirror breakoff snapshot

For cloning a standby read-write access database with Oracle Data Guard:

- Physical standby databases are read-only databases. Under normal circumstances, they offer limited utility value since you can not update their contents.
- Using the Data Guard support feature, DBI commands can clone a read-only standby database into a writable database.

In environments where the Oracle Active Data Guard feature is used, the redo log application must be stopped explicitly on standby database before taking a database snapshot.

To use Database FlashSnap in an Active Data Guard environment

1. Stop redo log apply.

   For example:

   ```
   SQL>ALTER DATABASE RECOVER MANAGED STANDBY DATABASE CANCEL;
   ```

2. Run dbed_update.
3. Create a snapplan and validate it.
4. Take a snapshot.
5. Start redo log apply.

   For example:

   ```
   SQL>ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT;
   ```

6. Clone the snapshot.

Chapter 27

Using volume snapshots in a replicated environment

This chapter includes the following topics:

- Considerations for using volume snapshots with VVR
- Creating space-optimized snapshots in a VVR setup
- Creating third-mirror break-off snapshots in a VVR setup
- Considerations for using vxsfadm after replication role transfer

Considerations for using volume snapshots with VVR

If you are planning to use space-optimized snapshots or third-mirror break-off snapshots in a Volume Replicator (VVR) setup where the Oracle database on the primary host is replicated on the secondary host, the following considerations for SFDB tools apply.

- An Oracle single instance database is running on the Storage Foundation stack with all volumes in one disk group.
- The data and the archive log volumes are replicated on a secondary host.
- All the application volumes are within a single replicated volume group (RVG).
- For snapshots or clones on the VVR secondary site, instant and online snapshot modes are supported. Instant mode snapshots do not support Oracle RAC

databases. Hence, the primary VVR site must use an Oracle single instance configuration only, if you plan to use instant mode snapshots.

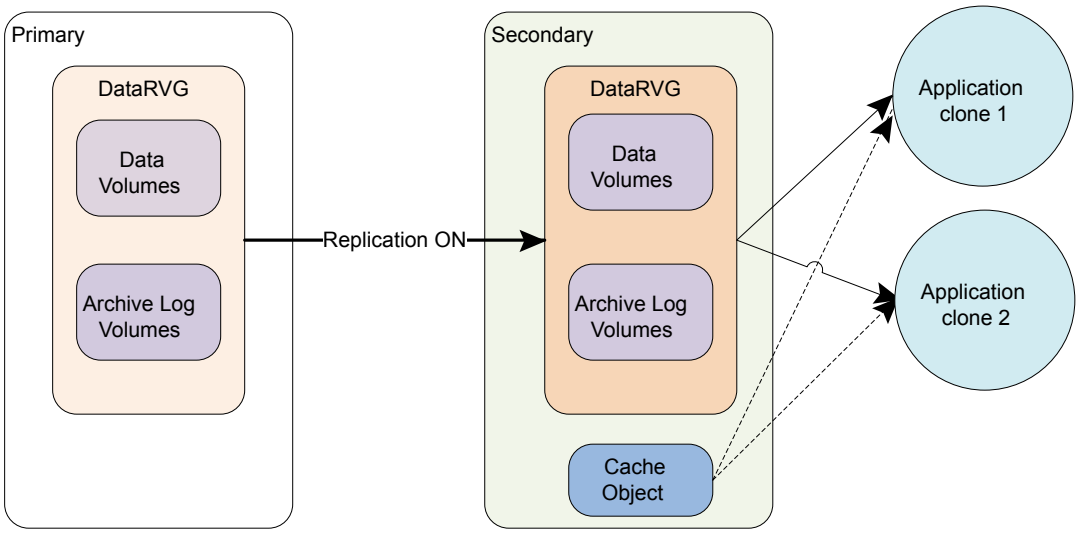
- Replication to multiple secondary hosts is supported.

Creating space-optimized snapshots in a VVR setup

To use space-optimized snapshots in a Volume Replicator (VVR) based replication environment, you need to generate a snapshot configuration on the primary host and then use that configuration on the secondary host.

Figure 27-1 provides an overview of a replicated database setup where space-optimized snapshots can be used.

Figure 27-1 Space-Optimized Snapshots - Replicated Database Setup



To create space-optimized snapshots in a VVR setup

1. On the primary host, generate the snapshot configuration for the secondary host.

```
$ opt/VRTSdbed/bin/vxsfadm -a oracle -o getvvrinfo
Gathering offline data...                                Done

ORACLE_SID           = sfaedb
REPLICATION_ROLE     = secondary
REPLICATION_DG_VOL   = orasecondarydg,orasecondarydg2:datavol
SNAPSHOT_TIME        = Wed Feb 15 09:50:17 2012

Configuration file /tmp/config1 is ready.
Use this configuration file on replication
secondary site for cloning application instance.
```

Note: You do not need to specify the configuration name and the replication role. The same configuration file can be used on any secondary site for online mode or instant mode snapshots.

---

2. Copy the newly generated snapshot configuration from the primary host to the secondary host.

3 On the secondary host, validate the snapshot configuration, and then create the snapshot.

```
$ /opt/VRTS/bin/vxsfadm -s sos -a oracle -o valid -c config1
Validating database configuration for space-optimized snapshot:
ORACLE_SID               = sfaedb
ORACLE_HOME              = /ora_base/db_home
APP_MODE                 = online
Database validation successful.
Validating database volume layout for space-optimized snapshot:
REPLICATION_DG_VOL       = orasecondarydg,orasecondarydg2:datavol
CACHE_OBJECT             = cacheobj
Replicated data volumes ready for snapshot:
Volume/volume-set datavol of diskgroup orasecondarydg.
SNAPSHOT_VOL_PREFIX      = SNAP_
Database volume layout validated successfully.
```

```
$ opt/VRTS/bin/vxsfadm -s sos -a oracle  -o snap -c config1
Creating snapshot volumes for data volumes ...                 Done
```

After validating the snapshot configuration on the secondary host, you can perform operations such as clone and destroy, locally on the secondary host.

4 On the secondary host, create a clone using the space-optimized snapshot.

```
$ /opt/VRTS/bin/vxsfadm -s sos -a oracle -o clone -c config1
Generated mount path: /var/tmp/az8EHWQpVe
Mounting snapshot volumes ...                            Done
Preparing parameter file for clone database ...          Done
Mounting clone database ...                              Done
Starting database recovery ...
Clone database sfaHrtyd is open.
```

5 On the secondary host, unmount and destroy the clone database, when you no longer need it.

```
$ /opt/VRTS/bin/vxsfadm -s sos -a oracle -o unmount -c config1
Shutting down clone database sfaHrtyd ...
Database shut down.
Unmounting snapshot volumes ...                          Done

$ /opt/VRTS/bin/vxsfadm -s sos -a oracle -o destroy -c config2
Destroying snapshot volumes ...                          Done
```

6 Restart the clone database.

```
bash-2.05b$ /opt/VRTS/bin/vxsfadm -s sos -a oracle -c config1\
-o clone --clone_name clones --clone_path /tmp/clones
Mounting snapshot volumes ...                                    Done
Starting clone database clones ...
Clone database started.
```

You can use the `vxsfadm` command after the replication role transfer.

See "Considerations for using vxsfadm after replication role transfer" on page 321.

See "vxsfadm command reference" on page 323.

See "Space-optimized snapshots configuration parameters" on page 331.

See "Space-optimized snapshots supported operations" on page 333.

Creating third-mirror break-off snapshots in a VVR setup

To use online third-mirror break-off (FlashSnap) snapshots in a Volume Replicator (VVR) based replication environment, you need to generate a snapshot configuration on the primary host and then use that configuration on the secondary host.

To create third-mirror break-off snapshots in a VVR setup

1 On the primary host, generate the snapshot configuration for the secondary host.

```
$ opt/VRTSdbed/bin/vxsfadm -a oracle -o getvvrinfo
Gathering offline data...                        Done

ORACLE_SID           = sfaedb
REPLICATION_ROLE     = secondary
REPLICATION_DG_VOL   = orasecondarydg,orasecondarydg2:datavol
SNAPSHOT_TIME        = Wed Feb 15 09:50:17 2012

Configuration file /tmp/config.aTKYn is ready.
Use this configuration file on replication
secondary site for cloning application instance.
```

Note: You do not need to specify the configuration name and the replication role. The same configuration file can be used on any secondary host for online mode or instant mode snapshots.

2 Copy the newly generated snapshot configuration from the primary host to the secondary host.

3 On the secondary host, validate the snapshot configuration.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap -a oracle \
--name flash3 -o valid -c config.aTKYn
Validating database configuration for third-mirror-break-off snapshot:
ORACLE_SID                  = sfaedb
ORACLE_HOME                 = /home/ora11gr2/app/ora11gr2/product
                              /11.2.0/dbhome_1
APP_MODE                    = online
SNAPSHOT_ARCHIVE_LOG        = yes
Validating database volume layout for space-optimized snapshot:
REPLICATION_DG_VOL          = orasecondarydg:datavol
CACHE_OBJECT                = co
Replicated volumes ready for snapshot:
Volume/volume-set datavol of diskgroup orasecondarydg.
Volume/volume-set archvol of diskgroup orasecondarydg.
SNAPSHOT_VOL_PREFIX         = SNAP_
SNAPSHOT_DG_PREFIX          = SNAP_
Database volume layout validated successfully.
```

After validating the snapshot configuration on the secondary host, you can perform operations such as clone and destroy, locally on the secondary host.

4 On the secondary host, create a third-mirror break-off snapshot.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap -a oracle \
--name flash3 -o snap -c config.aTKYn
snapshot started at Wed Feb 15 14:02:34 2012.
Initiating snapshot from Primary site ...
Message from Primary:
Putting database in backup mode ...                          Done
Message from Primary:
Taking database out of backup mode ...                       Done
Flushing online redo logs ...                                Done
Sending IBC message to Secondary ...                         Done
Message from Host system1.example.com:
Creating snapshot volumes for data volumes ...               Done
Message from Host system1.example.com:
Receiving IBC message from Primary ...                       Done
Creating snapshot volume for archivelog volume ...           Done
Copying snapshot information to snapshot volume ...          Done
Creating snapshot diskgroups ...                             Done
Deporting snapshot diskgroups ...                            Done
SNAP_orasecondarydg
snapshot ended at Wed Feb 15 14:02:55 2012.
```

You can use the following command to resynchronize snapshot volumes to original volumes.

```
$ /opt/VRTS/bin/vxsfadm -s flashsnap -a oracle \
--name flash3 -o resync -c config.aTKYn
resync started at Wed Feb 15 14:16:47 2012.
Importing snapshot diskgroups ...                            Done
Joining snapshot diskgroups to original diskgroups ...       Done
Reattaching snapshot volumes to original volumes ...         Done
resync ended at Wed Feb 15 14:16:58 2012.
```

Note: You cannot resynchronize a FlashSnap snapshot while there exist space-optimized snapshots created by using that FlashSnap snapshot as a golden image.

See "Creating multiple clones using FlashSnap snapshots" on page 214.

You can use the `vxsfadm` command after the replication role transfer.

See "Considerations for using vxsfadm after replication role transfer" on page 321.

See "vxsfadm command reference" on page 323.

See "FlashSnap configuration parameters" on page 327.

See "FlashSnap supported operations" on page 329.

See "Space-optimized snapshots configuration parameters" on page 331.

See "Space-optimized snapshots supported operations" on page 333.

Considerations for using vxsfadm after replication role transfer

The `vxsfadm` command can be used in case primary role is migrated or failed over to the secondary site. File system level snapshots, Storage Checkpoints, and FileSnap snapshots created on primary site are available on secondary site after transfer of replication primary role from primary to secondary. However, volume level snapshots (third-mirror break-off and space-optimized) are not usable on the new primary site. For the `vxsfadm` command to work after role transfer, you must validate the snapshot configuration generated on the primary site, on the replication or secondary site. The `vxsfadm` command can also be used in case primary role is transferred back to the original primary host.

The following requirements must be met for using the `vxsfadm` command after the primary role transfer:

- Application key (such as ORACLE_SID) is not modified during or after the replication role transfer.
- SFDB repository is available after replication role transfer. You can use repository backup tool to ensure repository availability.

See "vxsfadm command reference" on page 323.

Section 8

Storage Foundation for Databases administrative reference

- Chapter 28. Storage Foundation for Databases command reference
- Chapter 29. Tuning for Storage Foundation for Databases
- Chapter 30. Troubleshooting SFDB tools
- Chapter 31. Manual recovery of Oracle database
- Chapter 32. Storage Foundation for Databases command reference for the releases prior to 6.0

Chapter 28

Storage Foundation for Databases command reference

This chapter includes the following topics:

- vxsfadm command reference
- FlashSnap reference
- Space-optimized snapshots reference
- FileSnap reference
- Database Storage Checkpoints reference

vxsfadm command reference

`vxsfadm` is a command line utility that can be used to perform various point-in-time copy operations such as FlashSnap, FileSnap, space-optimized snapshots, and Database Storage Checkpoints for Oracle databases. This utility uses the underlying features of Storage Foundation to perform these operations.

Note: SFDB tools do not support taking point-in-time copies while structural changes to the database are in progress, such as adding or dropping tablespaces and adding or dropping data files. However, once a point-in-time copy is taken, a clone can be created at any time, regardless of the status of the database.

The `vxsfadm` utility is supported in multiple database configurations including:

- Oracle single instance
- Oracle single instance with off-host
- Oracle single instance in a highly available setup (VCS cluster)
- Oracle single instance with Storage Foundation Cluster File System High Availability
- Oracle RAC with Storage Foundation for Oracle RAC
- Oracle in a replicated setup with Veritas Volume Replicator or Oracle Data Guard

The syntax of the `vxsfadm` command is as follows:

```
vxsfadm -s <service_type> -a <application-name> -o <operation>
[ -c config-file ]
[ -r repository-host ]
[ service-specific parameters ]
[ application-specific parameters ]

vxsfadm -s <service-type> -a application-name> -o list
[ service-specific parameters ]
[ application-specific parameters ]

vxsfadm -s <service-type> -a application-name> -o setdefaults
[ service-specific parameters ]
[ application-specific parameters ]
```

The following are the `vxsfadm` parameters.

`-s service_type`

Specifies the service type for the point-in-time copy operation.

Supported service types are:

- flashsnap: Volume-level snapshots
- sos: Space-optimized volume-level snapshots
- filesnap: File-level snapshots
- checkpoint: File system checkpoints

`-o operation`

Specifies the point-in-time copy operation that is being performed. The supported operations vary for the different service types. For more information on the operations supported for a service type, refer to the respective `vxsfadm` man page.

The following operations are common to all service types:

- `-o list`: Lists all the configurations/services for the specified application. If *service_type* is specified then all existing configurations for the specified *service_type* are displayed.
- `-o setdefaults`: Generates the default configuration that can be saved to a file and then it can be used for the remainder of the operations.

`-a application_name`

Specifies the application for which the point-in-time-copy operation is being performed.

`-c config_file`

All the command line options specific to applications apart from service_type, application_name, and operation can be provided in the configuration file. The information in the configuration file should be in the format PARAMETER_NAME=VALUE. For information about the parameters supported for a service, refer to the respective `vxsfadm` man page.

`-r repository-host`

Specifies the host of the SFDB repository for off-host operations.

`service-specific parameters`

These parameters are required by a service for processing. The most important service-specific parameter is *--service*_name. For example, if you want to create a third-mirror break-off snapshot (flashsnap), the service_name parameter is flashsnap_name and you can specify a unique identifier for this parameter as follows: `--flashsnap_name=dailysnap`. This identifier can be used for all operations performed on that copy.

These parameters can be provided in the configuration file or they can be exported in the environment.

For more information, refer to the `vxsfadm` man pages.

`application-specific parameters`

These parameters are required by application for processing. The most important application-specific parameter is the one that uniquely identifies an application instance.

For example, if there are two Oracle databases running on a system, oracle_sid and oracle_home parameters uniquely identify the application instance. These parameter are used for all the point-in-time copy operations for the specific application instance.

These parameters can be provided in the configuration file or they can be exported in the environment.

For more information, refer to the `vxsfadm` man pages.

Note: The `vxsfadm` parameters specified on the command line override the parameters specified in the configuration file and the parameters exported in the environment. The parameters specified in the configuration file override the parameters exported in the environment.

You can use `vxsfadm` logs and error messages for troubleshooting.

See "Resources for troubleshooting SFDB tools" on page 369.

For more information, refer to the following man pages:

```
vxsfadm-flashsnap(1M), vxsfadm-checkpoint(1M)

vxsfadm-sos(1M), vxsfadm-filesnap(1M)
vxsfadm-oracle-flashsnap(1M), vxsfadm-oracle-sos(1M),
vxsfadm-oracle-checkpoint(1M), vxsfadm-oracle-filesnap(1M)
```

FlashSnap reference

This section describes FlashSnap configuration parameters and supported operations.

FlashSnap configuration parameters

Table 28-1 lists the FlashSnap configuration parameters that can be provided in the configuration file. If you want to specify the parameter on the command line, refer to the Command Line Option column of the table.

Table 28-1 FlashSnap - Configuration Parameters

Parameter	Description	Accepted Values	Command Line Option
VERSION	The version of the configuration file format.	n.n Example: 7	NA
FLASHSNAP_NAME (*)	A unique identifier of the FlashSnap configuration.	String Example: snap1	`--flashsnap_name snap1` OR `--name snap1`
ORACLE_SID (*)	The SID of the database of which the snapshot is to be taken.	String Example: orcl	`--oracle_sid orcl` OR `-S orcl`
ORACLE_HOME (*)	The Oracle Home directory of the SID of the database.	Valid path Example: `/oracle/`*`databaseHome`	`--oracle_home /oracle/`*`databaseHome` OR `-H /oracle/`*`databaseHome`
APP_MODE	The mode of the application when the snapshot operation is being performed.	offline online instant	`--app_mode offline` OR `online` OR `instant`
PFILE	The location of the database pfile, if the database is running with a pfile in a non-standard location.	pfile location	--pfile *location*

Table 28-1 FlashSnap - Configuration Parameters *(continued)*

Parameter	Description	Accepted Values	Command Line Option
SNAPSHOT_PLEX_TAG	The value of the putil2 attribute tag for the plexes that must be a part of the snapshot.	String Example: dbed_snap1	`--snapshot_plex_tag dbed_snap1` OR `--plex_tag dbed_snap1`
SNAPSHOT_VOL_PREFIX	The string prefixed to volume names to create snapshot volume names.	String Example: SNAPVOL_	`--snapshot_vol_prefix SNAPVOL_`
SNAPSHOT_DG_PREFIX	The string prefixed to disk group names to create snapshot disk group names.	String Example: SNAPDG_	`--snapshot_dg_prefix SNAPDG_`
SNAPSHOT_MIRROR	The number of mirrors that need to broken off to form the snapshot volumes.	Number	`--snapshot_mirror 2` OR `--n 2`
SNAPSHOT_ARCHIVE_LOG	If this parameter is set, the snapshot operation is also performed on the archive log volumes. This parameter is applicable for Oracle databases only.	Y or N	`--snapshot_archive_log` OR `--no_snapshot_archive_log`
SECONDARY_HOST	The host on which the snapshot can be mounted and the application can be cloned.	Host name	`--secondary_host sys4`
PFILE_MOD	The location of the file in the Oracle pfile format that is used to modify initialization parameters for the clone instance.	The location of the pfile.	`--pfile_mod` *`pfile_location`*
MAPPED_MOUNTS	The paths where the snapshot volumes should be mounted.	dg1:volume1=path1; dg2:volume2=path2 Example: mydg:datavol=/tmp/fsnp;	`--mapped_mounts mydg:datavol=/tmp/fsnp`
CLONE_PATH	The directory under which the snapshot volumes must be mounted.	Valid path `/tmp/so1`	`--clone_path /tmp/so1` OR `mountpath /tmp/so1`

Table 28-1 FlashSnap - Configuration Parameters *(continued)*

Parameter	Description	Accepted Values	Command Line Option
CLONE_NAME	The Oracle_SID of the clone instance.	String Example: clone1	`--clone_name clone1`
EXTRA_OBJECTS	The list of additional objects that must be included in the snapshot disk groups. Examples of additional objects: volumes, volume sets, cache objects.	Comma-separated list of objects such that each object is specified as <diskgroup>:<object>	`--extra_objects` *<dg1>*`:`*<vol1>*`,`*<dg2>*`:`*<vol2>* OR `--extra` *<dg1>*`:`*<co1>*
`REVERSE_RESYNC_ARCHIVE_LOG`	Whether archivelog volume is to be reverse resynchronized.	Yes or No (Default: No)	`--reverse_resync_archive_log <value>` OR `--rr_archive_log <value>`
`REVERSE_RESYNC_RECOVERY`	Method for recovering database during reverse resynchronization.	Auto, Manual or Auto_Until_SCN (Default: Auto)	`--reverse_resync_recovery <value>` OR `--rr_recovery <value>`

Note: The path that is specified with the PFILE_MOD parameter is local to the system on which the clone operation is performed. If the clone operation is performed on a host other than the one from which the command is invoked, then the parameter file should be copied to the host on which the clone operation is performed.

Note: (*) denotes that the parameter is mandatory.

FlashSnap supported operations

Table 28-2 lists the FlashSnap operations that you can perform by using the `vxsfadm` command.

Table 28-2 FlashSnap Supported Operations

Operation	Operation alias	Description
setdefaults	None	Generates the default configuration file for an application.

Table 28-2 FlashSnap Supported Operations *(continued)*

Operation	Operation alias	Description
valid	validate	Validates the configuration file and the environment.
snap	snapshot, break	Takes a snapshot by breaking the mirrors, and splits the disk groups.
clone	clonedb	Creates an application clone. You can also use the clone option to restart a clone if it is unmounted.
mount	snapmount	Mounts the mirror volumes.
unmount	umount, snapunmount, snapumount	Unmounts the snapshots and if a clone is created, shuts down the clone.
destroy	delete, remove	Destroys the FlashSnap configuration from the repository.
resync	None	Resynchronizes the mirror volumes with primary volumes.
reverse_resync_begin	rrbegin, rrstart, reverseresyncstart	Mounts the snapshot volumes and brings up the oracle instance.
reverse_resync_abort	rrabort, rrcancel, reverseresynccancel	Ends the reverse resync preview session and gets back to the original data volumes.
reverse_resync_commit	rrcommit, rrfinish, reverseresyncfinish	Commits the reverse resync changes that are previewed in the reverse_resync_begin operation.

Table 28-2 FlashSnap Supported Operations *(continued)*

Operation	Operation alias	Description
preclone	mountDB	Performs all the operations required for cloning a database and opens a database in the mount state. You need to perform the database recovery manually.
list	None	Lists the available snapshot configurations.

Space-optimized snapshots reference

This section describes FlashSnap configuration parameters and supported operations.

Space-optimized snapshots configuration parameters

Table 28-3 lists the space-optimized snapshots configuration parameters that can be provided in the configuration file. If you want to specify the parameter on the command line, refer to the Command Line Option column of the table.

Table 28-3 Space-Optimized Snapshots - Configuration Parameters

Parameter	Description	Accepted Values	Command Line Option
SOS_NAME (*)	A unique identifier of the snapshot configuration.	String Example: sossnap	`--sos_name sossnap` OR `--name sossnap`
VERSION	The version of the configuration file format.	n.n Example: 7	NA
ORACLE_SID (*)	The SID of the database of which the snapshot is being taken.	String Example: orcl	`--oracle_sid orcl` OR `--S orcl`
ORACLE_HOME (*)	The Oracle Home directory of the SID of which the snapshot is being taken.	Valid path Example: `/oracle/`*`databaseHome`*	`--oracle_home /oracle/`*`databaseHome`*

Table 28-3 Space-Optimized Snapshots - Configuration Parameters *(continued)*

Parameter	Description	Accepted Values	Command Line Option
APP_MODE	The mode of the application when the snapshot operation is being performed.	offline online instant	`--app_mode offline` OR `online` OR `instant`
MAPPED_MOUNTS	The volume-to-mountpoint mapping provided by the user.	dg1:volume1=path1; dg2:volume2=path2 Example: mydg:datavol=/tmp/so;	`--mapped_mounts mydg:datavol=/tmp/so`
CLONE_NAME	The name of the application instance that is created during the clone operation.	String Example: clone1	`--clone_name= clone1`
CLONE_PATH	The path to which the clone files are mounted.	Valid path `/tmp/so1`	`--clone_path /tmp/so1` OR `--mount_path /tmp/so1` OR `--on /tmp/so1` OR `--at /tmp/so1`
SNAPSHOT_VOL_PREFIX	The prefix to the mirror volume names.	String Example: SNAPVOL_ Default: SNAP_	`--snapshot_vol_prefix SNAPVOL_` OR `--vol_prefix SNAPVOL_`
SNAPSHOT_ARCHIVE_LOG	If this parameter is set, the snapshot operation is also performed on the archive log volumes. This parameter is applicable for Oracle databases only.	Y or N Default: Y	`--snapshot_archive_log` OR `--no_snapshot_archive_log`
PFILE_MOD	The location of the file in the Oracle pfile format that is used to modify initialization parameters for the clone instance. This parameter is applicable for Oracle databases only.	The location of the pfile.	`--pfile_mod` *`pfile_location`*

Table 28-3 Space-Optimized Snapshots - Configuration Parameters *(continued)*

Parameter	Description	Accepted Values	Command Line Option
PFILE	The location of the database pfile, if the database is running with a pfile in a non-standard location.	The location of the pfile.	`--pfile` *`pfile_location`*
CACHE_OBJECT	The comma separated values of the disk group and the corresponding cache object.	str1:str2,str3:str4,... Example: mydg1:C1,mydg2:c=C2,...	`--cache_object mydg1:C1,mydg2:c=C2`
SOURCE_SNAPSHOT	The name of the FlashSnap configuration that must be used as a golden image for space-optimized snapshots.	String Example: dailysnap	`--source_snapshot dailysnap` OR `--source dailysnap`

Note: The path that is specified with the PFILE_MOD parameter is local to the system on which the clone operation is performed. If the clone operation is performed on a host other than the one from which the command is invoked, then the parameter file should be copied to the host on which the clone operation is performed.

Note: (*) denotes that the parameter is mandatory.

Space-optimized snapshots supported operations

Table 28-4 lists the space-optimized snapshots operations that you can perform by using the `vxsfadm` command.

Table 28-4 Space-optimized Snapshots Supported Operations

Operation	Operation alias	Description
setdefaults	None	Generates the default configuration file for an application.
valid	validate	Validates the database environment and the system setup.

Table 28-4 Space-optimized Snapshots Supported Operations *(continued)*

Operation	Operation alias	Description
snap	snapshot	Creates a snapshot of the volumes by using the specified cache object.
clone	clonedb	Creates an application clone. You can also use this operation to restart a clone if it is unmounted.
mount	snapmount	Mounts the volumes at the specified location.
umount	snapunmount, umount, snapumount	Unmounts the mounted snapshots.
preclone	mountdb	Opens the database in the mount state. **Note:** This operation is specific to Oracle databases.
destroy	delete, remove	Destroys the snapshot.
getvvrinfo	None	Generates a snapshot configuration that can be used on a remote site for cloning replicated database in a VVR setup.

FileSnap reference

This section describes FileSnap configuration parameters and supported operations.

FileSnap configuration parameters

Table 28-5 lists the FileSnap configuration parameters that can be provided in the configuration file. If you want to specify the parameter on the command line, refer to the Command Line Option column of the table.

Table 28-5 FileSnap - Configuration Parameters

Parameter	Description	Accepted Values	Command Line Option
VERSION (*)	The version of the configuration file format.	n.n Example: 7	NA
FILESNAP_NAME (*)	A unique identifier of the snapshot configuration.	String Example: filesnap1	`--filesnap_name filesnap1`
ORACLE_SID (*)	The SID of the database of which the snapshot is being taken.	String Example: orcl	`--oracle_sid orcl`
ORACLE_HOME (*)	The Oracle Home directory of the SID of which the snapshot is being taken.	Valid path Example: `/oracle/`*`databaseHome`	`--oracle_home /oracle/`*`databaseHome` OR `--H /oracle/`*`databaseHome`
CLONE_NAME	The name of the application instance that is created during the clone operation.	String Example: clone1	`--clone_name clone1`
PFILE_MOD	The location of a pfile with modified parameters that can be used during the clone operation.	The location of the pfile.	`--pfile_mod` *`pfile_location`*
PFILE	The path to the Oracle PFILE, if the database is using a PFILE and the PFILE is in a location other than *ORACLE_HOME*/dbs.	The location of the pfile.	`--pfile` *`pfile_location`*

Note: The path that is specified with the PFILE_MOD parameter is local to the system on which the clone operation is performed. If the clone operation is performed on a host other than the one from which the command is invoked, then the parameter file should be copied to the host on which the clone operation is performed.

Note: (*) denotes that the parameter is mandatory.

FileSnap supported operations

Table 28-6 lists the FileSnap operations that you can perform by using the `vxsfadm` command.

Table 28-6 FileSnap Supported Operations

Operation	Operation alias	Description
setdefaults	None	Generates the default configuration file for an application.
snap	snapshot	Creates a snapshot of the volumes used by the application.
destroysnap	None	Destroys the snapshot.
clone	clonedb	Creates an application clone.
destroyclone	None	Destroys the application clone.
restore	None	Restores the files of which a snapshot was taken to the primary files
list	None	Lists the snapshots and clones with their status.

Database Storage Checkpoints reference

This section describes FlashSnap configuration parameters and supported operations.

Database Storage Checkpoints configuration parameters

Table 28-7 lists the Database Storage Checkpoints configuration parameters that can be provided in the configuration file. If you want to specify the parameter on the command line, refer to the Command Line Option column of the table.

Table 28-7 Database Storage Checkpoints - Configuration Parameters

Parameter	Description	Accepted Values	Command Line Option
VERSION	The version of the configuration file format.	n.n Example: 6.0	NA

Table 28-7 Database Storage Checkpoints - Configuration Parameters *(continued)*

Parameter	Description	Accepted Values	Command Line Option
CHECKPOINT_NAME (*)	A unique identifier of the storage checkpoint.	String Example: ckpt1	`--checkpoint_name ckpt1` OR `--checkpoint-name ckpt1` OR `-name ckpt1`
ORACLE_SID (*)	The SID of the database of which the storage storage checkpoint is being taken.	String Example: ora11gr2	`--oracle_sid ora11gr2` OR `-S ora11gr2`
ORACLE_HOME (*)	The Oracle Home directory of the SID of which the storage checkpoint is being taken.	String Example: /oracle_home	`--oracle_home /oracle_home` OR `--oracle-home /oracle_home` OR `-H /oracle_home`
APP_MODE	The mode of the application when the snapshot operation is being performed.	offline online instant Default: online	`--app_mode offline` OR `online` OR `instant`
CLONE_NAME	The name of the application instance that is created during the clone operation.	String Example: clone1	`--clone_name clone1` OR `--clone-name clone1`
CLONE_PATH	The path to which the clone files are mounted.	Valid path `/tmp/ckpt1`	`--clone_path /tmp/ckpt1` OR `--clone-path /tmp/ckpt1` OR `--mount-path /tmp/ckpt1` OR `--mount_path /tmp/ckpt1`
SNAPSHOT_REMOVABLE	A parameter to specify whether a removable or a non-removable storage checkpoint is being created.	Y or N Default: removable Y	`--snapshot_removable` OR `--no_snapshot_removable--snapshot-removable` or `--no-snapshot-removable`

Table 28-7 Database Storage Checkpoints - Configuration Parameters *(continued)*

Parameter	Description	Accepted Values	Command Line Option
DATAFILE_ROLLBACK	A comma separated list of data files that are to be rolled back.	A comma separated list of data files.	`--datafile_rollback` *`list`* `OR` `--datafile-rollback` *`list`*
DATAFILE_LIST_ROLLBACK	A file containing a list of data files, separated by new line, that are to be rolled back.	The path to the file containing the list of data files.	`--datafile_list_rollback` *`path`* `OR` `--datafile-list-rollback` *`path`*
TABLESPACE_ROLLBACK	The name of the table space that is to be rolled back.	Table space name	`--tablespace_rollback` *`name`* `OR` `--tablespace-rollback` *`name`*
PFILE_MOD	The location of the file in the Oracle pfile format that is used to modify initialization parameters for the clone instance. This parameter is applicable for Oracle databases only.	The location of the pfile.	`--pfile_mod` *`pfile_location`* OR `--pfile-mod` *`pfile_location`*
PFILE	The location of the database pfile, if the database is running with a pfile in a non-standard location.	The location of the pfile.	`--pfile` *`pfile_location`*

Note: The path that is specified with the PFILE_MOD parameter is local to the system on which the clone operation is performed. If the clone operation is performed on a host other than the one from which the command is invoked, then the parameter file should be copied to the host on which the clone operation is performed.

Note: (*) denotes that the parameter is mandatory.

Database Storage Checkpoints supported operations

Table 28-8 lists the Database Storage Checkpoints operations that you can perform by using the `vxsfadm` command.

Table 28-8 Database Storage Checkpoints Supported Operations

Operation	Operation alias	Description
setdefaults	None	Generates the default configuration file for an application.
ckpt	create, checkpoint, fsckpt, snap, snapshot	Creates a Database Storage Checkpoint for the application.
mount	snapmount, ckptmount	Mounts the Database Storage Checkpoint in the read-only mode or the read-write mode to the specified mount point.
mountrw	snapmountrw, ckptmountrw	Mounts the Database Storage Checkpoint in the read-write mode (a shadow checkpoint of the original storage checkpoint is created and it is mounted as read_write).
unmount/umount	ckptumount, unmount, snapunmount, snapumount	Unmounts the Database Storage Checkpoint.
delete	destroy, remove, ckptremove	Removes the Database Storage Checkpoint.
clone	ckptclone, clonedb	Creates a clone of the Database Storage Checkpoint.
restore	None	Restores the data files from the Database Storage Checkpoint.

Table 28-8 Database Storage Checkpoints Supported Operations *(continued)*

Operation	Operation alias	Description
list	None	Lists the Database Storage Checkpoints.
getappdata	None	Gathers database information when the database is online. This information is used for offline checkpoint processing.

Chapter 29

Tuning for Storage Foundation for Databases

This chapter includes the following topics:

- Additional documentation
- About tuning Veritas Volume Manager (VxVM)
- About tuning VxFS
- About tuning Oracle databases
- About tuning AIX Virtual Memory Manager
- About tuning Solaris for Oracle

Additional documentation

Use the tuning tips and information provided in this chapter in conjunction with other more in-depth publications, such as:

- *Oracle Performance Tuning Tips & Techniques* (Osborne McGraw-Hill)
- Oracle installation documents (Oracle Corporation)
- *Oracle Performance Tuning* (O'Reilly & Associates)
- *Oracle Performance Tuning and Optimization* (Sams Publishing)
- *Bullet-Proofing your Oracle Database with HACMP: A Guide to Implementing AIX Databases with HACMP* (IBM Corporation)
- *IBM Configuration and Performance RedBooks* (IBM Corporation)
- *Storage Foundation Administrator's Guide*, chapter on “VxVM Performance Monitoring”

About tuning Veritas Volume Manager (VxVM)

Veritas Volume Manager (VxVM) is tuned for most configurations ranging from small systems to larger servers. On smaller systems with less than a hundred drives, tuning should not be necessary and Veritas Volume Manager should be capable of adopting reasonable defaults for all configuration parameters. On very large systems, however, there may be configurations that require additional tuning of these parameters, both for capacity and performance reasons.

Various mechanisms exist for tuning VxVM. Many parameters can be tuned using AIX's System Management Interface Tool (SMIT). Other values can only be tuned using the command line interface for VxVM.

For more information on tuning VxVM, see the *Storage Foundation Administrator's Guide*.

About obtaining volume I/O statistics

If your database is created on a single file system that is on a single volume, there is typically no need to monitor the volume I/O statistics. If your database is created on multiple file systems on multiple volumes, or the volume configurations have changed over time, it may be necessary to monitor the volume I/O statistics for the databases.

Use the `vxstat` command to access information about activity on volumes, plexes, subdisks, and disks under VxVM control, and to print summary statistics to the standard output. These statistics represent VxVM activity from the time the system initially booted or from the last time the counters were reset to zero. If no VxVM object name is specified, statistics from all volumes in the configuration database are reported. Use the `-g` option to specify the database disk group to report statistics for objects in that database disk group.

VxVM records the following I/O statistics:

- count of operations
- number of blocks transferred (one operation can involve more than one block)
- average operation time (which reflects the total time through the VxVM interface and is not suitable for comparison against other statistics programs)

VxVM records the preceding three pieces of information for logical I/Os, including reads, writes, atomic copies, verified reads, verified writes, plex reads, and plex writes for each volume. VxVM also maintains other statistical data such as read failures, write failures, corrected read failures, corrected write failures, and so on. In addition to displaying volume statistics, the `vxstat` command is capable of

displaying more detailed statistics on the components that form the volume. For detailed information on available options, refer to the `vxstat`(1M) manual page.

To reset the statistics information to zero, use the `-r` option. You can reset the statistics information for all objects or for only those objects that are specified. Resetting just prior to an operation makes it possible to measure the impact of that particular operation.

The following is an example of output produced using the `vxstat` command:

```
  OPERATIONS           BLOCKS         AVG TIME(ms)

TYP  NAME        READ   WRITE     READ     WRITE    READ     WRITE

vol  blop           0       0        0         0     0.0       0.0

vol  foobarvol      0       0        0         0     0.0       0.0

vol  rootvol    73017  181735   718528   1114227    26.8      27.9

vol  swapvol    13197   20252   105569    162009    25.8     397.0

vol  testvol        0       0        0         0     0.0       0.0
```

Additional information is available on how to use the `vxstat` output to identify volumes that have excessive activity and how to reorganize, change to a different layout, or move these volumes.

Additional volume statistics are available for RAID-5 configurations.

See the `vxstat`(1M) manual page.

See the “Performance Monitoring” section of the “Performance Monitoring and Tuning” chapter in the *Storage Foundation Administrator's Guide*.

About tuning VxFS

Veritas File System provides a set of tuning options to optimize file system performance for different application workloads. VxFS provides a set of tunable I/O parameters that control some of its behavior. These I/O parameters help the file system adjust to striped or RAID-5 volumes that could yield performance far superior to a single disk. Typically, data streaming applications that access large files see the largest benefit from tuning the file system.

Most of these tuning options have little or no impact on database performance when using Quick I/O. However, you can gather file system performance data when using Quick I/O, and use this information to adjust the system configuration to make the most efficient use of system resources.

How monitoring free space works

In general, VxFS works best if the percentage of free space in the file system is greater than 10 percent. This is because file systems with 10 percent or more of free space have less fragmentation and better extent allocation. Regular use of the `df` command to monitor free space is desirable. Full file systems may have an adverse effect on file system performance. Full file systems should therefore have some files removed or should be expanded.

See the `fsadm_vxfs`(1M) manual page.

About monitoring fragmentation

Fragmentation reduces performance and availability. Regular use of `fsadm`'s fragmentation reporting and reorganization facilities is therefore advisable.

The easiest way to ensure that fragmentation does not become a problem is to schedule regular defragmentation runs using the `cron` command.

Defragmentation scheduling should range from weekly (for frequently used file systems) to monthly (for infrequently used file systems). Extent fragmentation should be monitored with `fsadm` or the `df -os` commands.

Defragmentation scheduling should range from weekly (for frequently used file systems) to monthly (for infrequently used file systems). Extent fragmentation should be monitored with `fsadm`command.

There are three factors that can be used to determine the degree of fragmentation:

- Percentage of free space in extents that are less than eight blocks in length
- Percentage of free space in extents that are less than 64 blocks in length
- Percentage of free space in extents that are 64 or more blocks in length

An unfragmented file system will have the following characteristics:

- Less than 1 percent of free space in extents that are less than eight blocks in length
- Less than 5 percent of free space in extents that are less than 64 blocks in length
- More than 5 percent of the total file system size available as free extents that are 64 or more blocks in length

A badly fragmented file system will have one or more of the following characteristics:

- More than 5 percent of free space in extents that are less than 8 blocks in length
- More than 50 percent of free space in extents that are less than 64 blocks in length

- Less than 5 percent of the total file system size available as free extents that are 64 or more blocks in length

The optimal period for scheduling extent reorganization runs can be determined by choosing a reasonable interval, scheduling `fsadm` runs at the initial interval, and running the extent fragmentation report feature of `fsadm` before and after the reorganization.

The “before” result is the degree of fragmentation prior to the reorganization. If the degree of fragmentation approaches the percentages for bad fragmentation, reduce the interval between `fsadm`. If the degree of fragmentation is low, increase the interval between `fsadm` runs.

How tuning VxFS I/O parameters works

VxFS provides a set of tunable I/O parameters that control some of its behavior. These I/O parameters are useful to help the file system adjust to striped or RAID-5 volumes that could yield performance far superior to a single disk. Typically, data streaming applications that access large files see the biggest benefit from tuning the file system.

If VxFS is being used with Veritas Volume Manager, the file system queries VxVM to determine the geometry of the underlying volume and automatically sets the I/O parameters. VxVM is queried by `mkfs` when the file system is created to automatically align the file system to the volume geometry. If the default alignment from `mkfs` is not acceptable, the `-o align=n` option can be used to override alignment information obtained from VxVM. The `mount` command also queries VxVM when the file system is mounted and downloads the I/O parameters.

If the default parameters are not acceptable or the file system is being used without VxVM, then the `/etc/vx/tunefstab` file can be used to set values for I/O parameters. The `mount` command reads the `/etc/vx/tunefstab` file and downloads any parameters specified for a file system. The `tunefstab` file overrides any values obtained from VxVM. While the file system is mounted, any I/O parameters can be changed using the `vxtunefs` command, which can have tunables specified on the command line or can read them from the `/etc/vx/tunefstab` file.

The `vxtunefs` command can be used to print the current values of the I/O parameters.

See the `vxtunefs`(1M) and `tunefstab`(4) manual pages.

About tunable VxFS I/O parameters

The following are tunable VxFS I/O parameters:

`read_pref_io`

The preferred read request size. The file system uses this parameter in conjunction with the `read_nstream` value to determine how much data to read ahead. The default value is 64K.

`write_pref_io`

The preferred write request size. The file system uses this parameter in conjunction with the `write_nstream` value to determine how to do flush behind on writes. The default value is 64K.

`read_nstream`

The number of parallel read requests of size `read_pref_io` that you can have outstanding at one time. The file system uses the product of `read_nstream` multiplied by `read_pref_io` to determine its read ahead size. The default value for `read_nstream` is 1.

`write_nstream`

The number of parallel write requests of size `write_pref_io` that you can have outstanding at one time. The file system uses the product of `write_nstream` multiplied by `write_pref_io` to determine when to do flush behind on writes. The default value for `write_nstream` is 1.

`default_indir_ size`

On VxFS, files can have up to ten variably sized direct extents stored in the inode. After these extents are used, the file must use indirect extents that are a fixed size. The size is set when the file first uses indirect extents. These indirect extents are 8K by default. The file system does not use larger indirect extents because it must fail a write and return `ENOSPC` if there are no extents available that are the indirect extent size. For file systems with many large files, the 8K indirect extent size is too small. Large files that require indirect extents use many smaller extents instead of a few larger ones. By using this parameter, the default indirect extent size can be increased so that large files in indirects use fewer large extents.

Be careful using this tunable. If it is too large, then writes fail when they are unable to allocate extents of the indirect extent size to a file. In general, the fewer and the larger the files on a file system, the larger the `default_indir_size` parameter can be. The value of this parameter is generally a multiple of the `read_pref_io` parameter.

This tunable is not applicable on Version 4 disk layouts.

`discovered_direct_iosz` Any file I/O requests larger than the `discovered_direct_iosz` are handled as discovered direct I/O. A discovered direct I/O is unbuffered similar to direct I/O, but does not require a synchronous commit of the inode when the file is extended or blocks are allocated. For larger I/O requests, the CPU time for copying the data into the page cache and the cost of using memory to buffer the I/O data becomes more expensive than the cost of doing the disk I/O. For these I/O requests, using discovered direct I/O is more efficient than regular I/O. The default value of this parameter is 256K.

`initial_extent_ size` Changes the default initial extent size. VxFS determines the size of the first extent to be allocated to the file based on the first write to a new file. Normally, the first extent is the smallest power of 2 that is larger than the size of the first write. If that power of 2 is less than 8K, the first extent allocated is 8K. After the initial extent, the file system increases the size of subsequent extents (see `max_seqio_extent_size`) with each allocation. Since most applications write to files using a buffer size of 8K or less, the increasing extents start doubling from a small initial extent. `initial_extent_size` can change the default initial extent size to be larger, so the doubling policy will start from a much larger initial size and the file system will not allocate a set of small extents at the start of file. Use this parameter only on file systems that will have a very large average file size. On these file systems, it will result in fewer extents per file and less fragmentation. `initial_extent_size` is measured in file system blocks.

`max_direct_iosz` The maximum size of a direct I/O request that will be issued by the file system. If a larger I/O request comes in, then it is broken up into `max_direct_iosz` chunks. This parameter defines how much memory an I/O request can lock at once, so it should not be set to more than 20 percent of memory.

`max_diskq` Limits the maximum disk queue generated by a single file. When the file system is flushing data for a file and the number of pages being flushed exceeds `max_diskq`, processes will block until the amount of data being flushed decreases. Although this doesn't limit the actual disk queue, it prevents flushing processes from making the system unresponsive. The default value is 1MB.

`max_seqio_extent_size`
: Increases or decreases the maximum size of an extent. When the file system is following its default allocation policy for sequential writes to a file, it allocates an initial extent that is large enough for the first write to the file. When additional extents are allocated, they are progressively larger (the algorithm tries to double the size of the file with each new extent) so each extent can hold several writes' worth of data. This is done to reduce the total number of extents in anticipation of continued sequential writes. When the file stops being written, any unused space is freed for other files to use. Normally, this allocation stops increasing the size of extents at 2048 blocks, which prevents one file from holding too much unused space. `max_seqio_extent_size` is measured in file system blocks.

`qio_cache_enable`
: Enables or disables caching on Quick I/O files. The default behavior is to disable caching. To enable caching, set `qio_cache_enable` to 1. On systems with large memories, the database cannot always use all of the memory as a cache. By enabling file system caching as a second level cache, performance may be improved. If the database is performing sequential scans of tables, the scans may run faster by enabling file system caching so the file system will perform aggressive read-ahead on the files.

`write_throttle`

Warning: The `write_throttle` parameter is useful in special situations where a computer system has a combination of a lot of memory and slow storage devices. In this configuration, sync operations (such as `fsync()`) may take so long to complete that the system appears to hang. This behavior occurs because the file system is creating dirty pages (in-memory updates) faster than they can be asynchronously flushed to disk without slowing system performance.

Lowering the value of `write_throttle` limits the number of dirty pages per file that a file system will generate before flushing the pages to disk. After the number of dirty pages for a file reaches the `write_throttle` threshold, the file system starts flushing pages to disk even if free memory is still available. The default value of `write_throttle` typically generates a lot of dirty pages, but maintains fast user writes. Depending on the speed of the storage device, if you lower `write_throttle`, user write performance may suffer, but the number of dirty pages is limited, so sync operations will complete much faster.

Because lowering `write_throttle` can delay write requests (for example, lowering `write_throttle` may increase the file disk queue to the `max_diskq` value, delaying user writes until the disk queue decreases), it is recommended that you avoid changing the value of `write_throttle` unless your system has a a large amount of physical memory and slow storage devices.

If the file system is being used with VxVM, it is recommended that you set the VxFS I/O parameters to default values based on the volume geometry.

If the file system is being used with a hardware disk array or volume manager other than VxVM, align the parameters to match the geometry of the logical disk. With striping or RAID-5, it is common to set `read_pref_io` to the stripe unit size and `read_nstream` to the number of columns in the stripe. For striping arrays, use the same values for `write_pref_io` and `write_nstream`, but for RAID-5 arrays, set `write_pref_io` to the full stripe size and `write_nstream` to 1.

For an application to do efficient disk I/O, it should issue read requests that are equal to the product of `read_nstream` multiplied by `read_pref_io`. Generally, any multiple or factor of `read_nstream` multiplied by `read_pref_io` should be a good size for performance. For writing, the same rule of thumb applies to the `write_pref_io` and `write_nstream` parameters. When tuning a file system, the best thing to do is try out the tuning parameters under a real-life workload.

If an application is doing sequential I/O to large files, it should issue requests larger than the `discovered_direct_iosz`. This causes the I/O requests to be performed as discovered direct I/O requests, which are unbuffered like direct I/O but do not require synchronous inode updates when extending the file. If the file is too large to fit in the cache, then using unbuffered I/O avoids throwing useful data out of the cache and lessons CPU overhead.

About obtaining file I/O statistics using the Quick I/O interface

The `qiostat`command provides access to activity information on Quick I/O files on VxFS file systems. The command reports statistics on the activity levels of files from the time the files are first opened using their Quick I/O interface. The accumulated `qiostat` statistics are reset once the last open reference to the Quick I/O file is closed.

The qiostat command displays the following I/O statistics:

- Number of read and write operations
- Number of data blocks (sectors) transferred
- Average time spent on read and write operations

When Cached Quick I/O is used, `qiostat` also displays the caching statistics when the `-l` (the long format) option is selected.

The following is an example of `qiostat` output:

```
                    OPERATIONS        FILE BLOCKS             AVG TIME(ms)

 FILENAME          READ      WRITE     READ      WRITE       READ      WRITE

 /db01/file1          0         00        0        0.0       0.0

 /db01/file2          0         00        0        0.0       0.0

 /db01/file3      73017     181735   718528    1114227      26.8       27.9

 /db01/file4      13197      20252   105569     162009      25.8      397.0

 /db01/file5          0         00        0        0.0       0.0
```

For detailed information on available options, see the `qiostat`(1M) manual page.

About I/O statistics data

Once you gather the file I/O performance data, you can use it to adjust the system configuration to make the most efficient use of system resources.

There are three primary statistics to consider:

- file I/O activity
- volume I/O activity
- raw disk I/O activity

If your database is using one file system on a striped volume, you may only need to pay attention to the file I/O activity statistics. If you have more than one file system, you may need to monitor volume I/O activity as well.

First, use the `qiostat -r` command to clear all existing statistics. After clearing the statistics, let the database run for a while during a typical database workload period. For example, if you are monitoring a database with many users, let the statistics accumulate for a few hours during prime working time before displaying the accumulated I/O statistics.

To display active file I/O statistics, use the `qiostat` command and specify an interval (using `-i`) for displaying the statistics for a period of time. This command displays a list of statistics such as:

```
                    OPERATIONS        FILE BLOCKS          AVG TIME(ms)

 FILENAME         READ      WRITE     READ      WRITE      READ      WRITE

 /db01/cust1       218         36      872        144      22.8       55.6

 /db01/hist1         0         10        4        0.0      10.0

 /db01/nord1        10         14       40         56      21.0       75.0

 /db01/ord1         19         16       76         64      17.4       56.2

 /db01/ordl1       189         41      756        164      21.1       50.0

 /db01/roll1         0         50        0        200       0.0       49.0

 /db01/stk1       1614        238     6456        952      19.3       46.5

 /db01/sys1          0         00        0        0.0       0.0

 /db01/temp1         0         00        0        0.0       0.0

 /db01/ware1         3         14       12         56      23.3       44.3

 /logs/log1          0         00        0        0.0       0.0

 /logs/log2          0        217 0   2255        0.0       6.8
```

File I/O statistics help identify files with an unusually large number of operations or excessive read or write times. When this happens, try moving the “hot” files or busy file systems to different disks or changing the layout to balance the I/O load.

```
Mon May 11 16:21:20 2015

/db/dbfile01               813     0      813       0     0.3   0.0

/db/dbfile02                 0   813        0     813     0.0   5.5

Mon May 11 16:21:25 2015

/db/dbfile01               816     0      816       0     0.3    0.0

/db/dbfile02                 0   816        0             816    0.0  5.3

Mon May 11 16:21:30 2015

/db/dbfile01                 0     0        0               0    0.0  0.0

/db/dbfile02                 0     0        0               0    0.0  0.0
```

Obtaining file I/O statistics using Veritas extension for Oracle Disk Manager

The `odmstat` command provides access to activity information on Oracle Disk Manager files on VxFS systems. Refer to the `odmstat`(1M) manual page for more information. The command reports statistics on the activity from the time that the files were opened by the Oracle Disk Manager interface. The command has an option for zeroing the statistics. When the file is closed, the statistics are discarded.

The odmstat command displays the following I/O statistics:

- Number of read and write operations
- Number of data blocks read and written
- Average time spent on read and write operations

To obtain i/o statistics

- Use the `odmstat` command as follows:

  ```
  # /opt/VRTS/bin/odmstat -i 5 /mnt/odmfile*
  ```

  For example:

  ```
  # /opt/VRTS/bin/odmstat -i 5 /db/odmfile*

                          OPERATIONS       FILE BLOCKS     AVG TIME(ms)

FILE NAME                READ     WRITE   READ     WRITE   READ   WRITE

Mon May 11 16:21:10 2015

/db/cust.dbf                0         0      0         0    0.0     0.0

/db/system.dbf              0         0      0         0    0.0     0.0

Mon May 11 16:21:15 2015

/db/cust.dbf              371         0    371         0    0.2     0.0

/db/system.dbf              0       371      0       371    0.0     5.7

Mon May 11 16:21:20 2015

/db/cust.dbf              813         0    813         0    0.3     0.0

/db/system.dbf              0       813      0       813    0.0     5.5

Mon May 11 16:21:25 2015

/db/cust.dbf              816         0    816         0    0.3     0.0

/db/system.dbf              0       816      0       816    0.0     5.3

Mon May 11 16:21:30 2015

/db/cust.dbf                0         0      0         0    0.0     0.0

/db/system.dbf              0         0      0         0    0.0     0.0
  ```

About I/O statistics

When running your database through the file system, the read-write lock on each file allows only one active write per file. When you look at the disk statistics using `iostat`, the disk reports queueing time and service time. The service time is the

time that I/O spends on the disk, and the queueing time is how long it waits for all of the other I/Os ahead of it. At the volume level or the file system level, there is no queueing, so `vxstat` and `qiostat` do not show queueing time.

For example, if you send 100 I/Os at the same time and each takes 10 milliseconds, the disk reports an average of 10 milliseconds of service and 490 milliseconds of queueing time. The `vxstat`, `odmstat`, and `qiostat` report an average of 500 milliseconds service time.

About tuning Oracle databases

To achieve optimal performance on your Oracle database, the database needs to be tuned to work with VxFS. There are a number of Oracle parameters that you can tune to improve your Oracle database performance.

Sequential table scans

Quick I/O in its default mode performs all I/O as direct I/O. In the case of single-threaded sequential scans (common in decision support system (DSS) workloads), using buffered reads can yield better performance. Because the file system detects these sequential reads and performs read-aheads, the next few blocks that Oracle requests are readily available in the system buffer cache and are simply copied to the Oracle system global area (SGA). Because access from memory is inherently faster than access from disk, this achieves a significant reduction in response time.

To handle large sequential scans when using Quick I/O, one of two methods is available to improve performance:

- Use the Oracle Parallel Query Option to break the single large scan into multiple smaller scans.

  Note: Consult the Oracle documentation for your system and version of Oracle, and use the settings recommended for these parameters when provided.

- The second method is to enable Cached Quick I/O for the files that would be read by the Oracle sequential scan process. Cached Quick I/O enables buffered reads, and the automatic file system read-ahead helps lower response times by pre-loading data.

  Note: Do not use this option if you are using a 64-bit version of Oracle.

Sequential table scans

Quick I/O performs all I/O as direct I/O. In the case of single-threaded sequential scans (common in decision support system (DSS) workloads), using buffered reads can yield better performance. Because the file system detects these sequential reads and performs read-aheads, the next few blocks that Oracle requests are readily available in the system buffer cache and are simply copied to the Oracle system global area (SGA). Because access from memory is inherently faster than access from disk, this achieves a significant reduction in response time.

To handle large sequential scans when using Quick I/O, one of two methods is available to improve performance:

- Use the Oracle Parallel Query Option to break the single large scan into multiple smaller scans.

  Note: Consult the Oracle documentation for your system and version of Oracle, and use the settings recommended for these parameters when provided.

- The second method is applicable only if you are using Oracle8. In this method, Cached Quick I/O is enabled for the files that would be read by the Oracle sequential scan process. Cached Quick I/O enables buffered reads, and the automatic file system read-ahead helps lower response times by pre-loading data.

Asynchronous I/O

Quick I/O and Oracle Disk Manager support kernel asynchronous I/O, which reduces CPU utilization and improves transaction throughput.

Oracle Disk Manager supports kernel asynchronous I/O, which reduces CPU utilization and improves transaction throughput.

Enabling the following parameters lets Oracle take advantage of asynchronous I/O and avoids having to configure multiple DBWR slaves:

- If you are using Quick I/O datafiles with Oracle9, set `DISK_ASYNCH_IO`to `TRUE`in `init.ora`.
- If you are using ODM on Oracle9, you do not need to change any `init.ora` parameters.

Your Oracle Installation Guide provides detailed instructions on implementing asynchronous I/O on your system.

Tuning buffer cache

The UNIX buffer cache plays an important role in performance when using JFS in buffered I/O mode.

The UNIX buffer cache plays an important role in performance when using UFS in buffered I/O mode.

When using Quick I/O, however, the database cache must be tuned as if raw devices are being used. You can allocate more memory to the database buffer cache because Quick I/O bypasses the file system cache to improve database performance. Memory pages normally allocated to the file system cache can be allocated to the database buffer cache (SGA). You can adjust the SGA size without shutting down the database.

Tuning buffer cache

The UNIX buffer cache plays an important role in performance when using UFS in buffered I/O mode.

When using Quick I/O, however, the database cache must be tuned as if raw devices are being used. You can allocate more memory to the database buffer cache because Quick I/O bypasses the file system cache to improve database performance. Memory pages normally allocated to the file system cache can be allocated to the database buffer cache (SGA). With Oracle9i, you can adjust the SGA size without shutting down the database.

Setting Oracle block reads during sequential scans

The `DB_FILE_MULTIBLOCK_READ_COUNT` parameter specifies the maximum number of blocks Oracle reads in one I/O operation during a sequential scan. The `/etc/system` tunable parameter `maxphys` establishes the maximum physical I/O transfer size at the operating system level. To take advantage of the maximum transfer size, the Oracle `init.ora` parameter `DB_FILE_MULTIBLOCK_READ_COUNT` should be set to `maxphys/DB_BLOCK_SIZE`.

For example, if `maxphys` is set to 1048576 and `DB_BLOCK_SIZE` is set to 4096, then `DB_FILE_MULTIBLOCK_READ_COUNT` should be set to 256.

The `DB_FILE_MULTIBLOCK_READ_COUNT` parameter specifies the maximum number of blocks Oracle reads in one I/O operation during a sequential scan. A large value for the `DB_FILE_MULTIBLOCK_READ_COUNT` initialization parameter usually yields better I/O throughput. On AIX, this parameter ranges from 1 to 512, but using a value higher than 16 usually does not provide additional performance gain.

When the file system is created on a striped volume, set this parameter to a value that is a multiple of the full stripe size divided by `DB_BLOCK_SIZE`. Using a full stripe size allows the read operations to take advantage of the full bandwidth of the striped disks during sequential table scan.

Set the `DB_FILE_MULTIBLOCK_READ_COUNT` to a value that is a multiple of `(read_pref_io*read_nstream)/DB_BLOCK_SIZE`, but the value should not exceed the value of `max_direct_iosz/DB_BLOCK_SIZE`.

Use the `vxtunefs` command to display the value of `read_pref_io`, `read_nstream`, and `max_direct_iosz`, for example:

```
# vxtunefs /db01
```

The `vxtunefs` command displays output similar to the following:

```
Filesystem i/o parameters for /db01

read_pref_io  = 65536
read_nstream  = 1
read_unit_io  = 65536
write_pref_io  = 65536
write_nstream  = 1
write_unit_io  = 65536
pref_strength  = 10
buf_breakup_size  = 262144
discovered_direct_iosz  = 262144
max_direct_iosz  = 1048576
default_indir_size = 8192

qio_cache_enable = 0

write_throttle = 0

max_diskq = 1048576

initial_extent_size = 8

max_seqio_extent_size = 2048

max_buf_data_size = 8192

hsm_write_prealloc = 0

read_ahead = 1

inode_aging_size = 0
```

```
inode_aging_count = 0

fcl_maxalloc = 130150400

fcl_keeptime = 0

fcl_winterval = 3600

oltp_load = 0
```

For a description of these parameters and the tuning instructions, refer to the `vxtunefs`(1M) manual page.

The `DB_FILE_MULTIBLOCK_READ_COUNT` parameter specifies the maximum number of blocks Oracle reads in one I/O operation during a sequential scan. The `/etc/system` tunable parameter `maxphys` establishes the maximum physical I/O transfer size at the operating system level. To take advantage of the maximum transfer size, the Oracle `init.ora` parameter `DB_FILE_MULTIBLOCK_READ_COUNT` should be set to `maxphys/DB_BLOCK_SIZE`.

For example, if `maxphys` is set to 1048576 and `DB_BLOCK_SIZE` is set to 4096, then `DB_FILE_MULTIBLOCK_READ_COUNT` should be set to 256.

Setting slave parameters

Quick I/O and ODM provide support for kernel asynchronous I/O, eliminating the need for multiple logwriter slaves or database writer slaves. This parameter is set to `0` by default.

It is not necessary to set the DBWR_IO_SLAVES settings if you are using Quick I/O. The number of `DBWR` writer processes is set within DB_WRITER_PROCESSES, which performs asynchronous I/O.

ODM provides support for asynchronous I/O, eliminating the need for multiple logwriter slaves or database writer slaves. This parameter is set to `0` by default.

Configuring memory allocation

Never configure Oracle to make use of more memory than is physically available on the system. Oracle may have to compete with other processes for system memory resources, and all of these potential processes must be considered when sizing and allocating memory. In the ideal configuration, a system that is dedicated to Oracle simplifies the tuning and monitoring issues and ensures best performance.

About tuning AIX Virtual Memory Manager

If you are using either Cached Quick I/O or buffered I/O (that is, plain VxFS files without Quick I/O or mount options specified), it is recommended that you monitor any paging activity to the swap device on your database servers. To monitor swap device paging, use the `vmstat -I` command. Swap device paging information appears in the `vmstat -I` output under the columns labeled `pi` and `po` (for paging in and paging out from the swap device, respectively). Any nonzero values in these columns indicates swap device paging activity.

For example:

```
# /usr/bin/vmstat -I

kthr      memory                 page                            faults     cpu
--------  --------------------- ----------------------------- ---------- -----------
r  b  p   avm     fre      fi   fo pi  po  fr    sr    in    sy     cs    us sy id wa

5  1  0   443602  1566524  661  20 0   0   7     28    4760  37401  7580  11 7  43 38

1  1  0   505780  1503791  18   6  0   0   0     0     1465  5176   848   1  1  97 1

1  1  0   592093  1373498  1464 1  0   0   0     0     4261  10703  7154  5  5  27 62

3  0  0   682693  1165463  3912 2  0   0   0     0     7984  19117  15672 16 13 1  70

4  0  0   775730  937562   4650 0  0   0   0     0     10082 24634  20048 22 15 0  63

6  0  0   864097  715214   4618 1  0   0   0     0     9762  26195  19666 23 16 1  61

5  0  0   951657  489668   4756 0  0   0   0     0     9926  27601  20116 24 15 1  60

4  1  0   1037864 266164   4733 5  0   0   0     0     9849  28748  20064 25 15 1  59

4  0  0   1122539 47155    4476 0  0   0   0     0     9473  29191  19490 26 16 1  57

5  4  0   1200050 247      4179 4  70  554 5300  27420 10793 31564  22500 30 18 1  52

6  10 0   1252543 98       2745 0  138 694 4625  12406 16190 30373  31312 35 14 2  49

7  14 0   1292402 220      2086 0  153 530 3559  17661 21343 32946  40525 43 12 1  44

7  18 0   1319988 183      1510 2  130 564 2587  14648 21011 28808  39800 38 9  3  49
```

If there is evidence of swap device paging, proper AIX Virtual Memory Manager (VMM) tuning is required to improve database performance. VMM tuning limits the amount of memory pages allocated to the file system cache. This prevents the file system cache from stealing memory pages from applications (which causes swap device page-out) when the VMM is running low on free memory pages.

The command to tune the AIX VMM subsystem is:

```
# /usr/samples/kernel/vmtune
```

Changes made by `vmtune` last until the next system reboot. The VMM kernel parameters to tune include: `maxperm`, `maxclient`, and `minperm`. The `maxperm` and `maxclient` parameters specify the maximum amount of memory (as a percentage of total memory) that can be used for file system caching. The maximum amount of memory for file system caching should not exceed the amount of unused memory left by the AIX kernel and all active applications. Therefore, it can be calculated as:

100*(T-A)/T

where T is the total number of memory pages in the system and A is the maximum number of memory pages used by all active applications.

The `minperm` parameter should be set to a value that is less than or equal to `maxperm`, but greater than or equal to 5.

For more information on AIX VMM tuning, see the `vmtune`(1) manual page and the performance management documentation provided with AIX.

The following is a tunable VxFS I/O parameter:

VMM Buffer Count

(`-b` <value> option)

Sets the virtual memory manager (VMM) buffer count. There are two values for the VMM: a default value based on the amount of memory, and a current value. You can display these two values using `vxtunefs -b`. Initially, the default value and the current value are the same. The `-b` value option specifies an increase, from zero to 100 per cent, in the VMM buffer count from its default. The specified value is saved in the file `/etc/vx/vxfssystem` to make it persistent across VxFS module loads or system reboots.

In most instances, the default value is suitable for good performance, but there are counters in the kernel that you can monitor to determine if there are delays waiting for VMM buffers. If there appears to be a performance issue related to VMM, the buffer count can be increased. If there is better response time on the system, it is a good indication that VMM buffers were a bottleneck.

The following fields displayed by the `kdb vmker` command can be useful in determining bottlenecks.

`THRPGIO buf wait (_waitcnt) value`

This field may indicate that there were no VMM buffers available for pagein or pageout. The thread was blocked waiting for a VMM buffer to become available. The count is the total number of waits since cold load. This field, together with pages “paged in” and pages “paged out” displayed by the kdb vmstat command can be used to determine if there are an adequate number of VMM buffers. The ratio:

waitcnt / pageins+pageouts

is an indicator of waits for VMM buffers, but cannot be exact because pageins + pageouts includes page I/Os to other file systems and pageing space. It is not possible to give a typical value for this ratio because it depends on the amount of memory and page I/Os to file systems other than VxFS. A number greater than 0.1 may indicate a VMM buffer count bottleneck. Other relevant fields displayed by `kdb vmker` are:

- `THRPGIO partial cnt (_partialcnt) value`
  This field indicates page I/O was done in two or more steps because there were fewer VMM buffers available than the number of pages requiring I/O.
- `THRPGIO full cnt (_fullcnt) value`
  All the VMM buffers were found for all the pages requiring I/O.

About tuning Solaris for Oracle

To achieve optimal performance using Storage Foundation for Oracle, certain Solaris parameters need to be tuned. Changing these parameters requires modifying the Solaris kernel settings (specified in the `/etc/system` file) and rebooting the system.

You can add or change these tuning parameters in the `/etc/system` file using a text editor. The following example shows the contents of an `/etc/system` file:

```
* start Oracle *
set shmsys:shminfo_shmmax=0xffffffff
set shmsys:shminfo_shmmin=1
set shmsys:shminfo_shmmni=100
set shmsys:shminfo_shmseg=200
*
set semsys:seminfo_semmap=100
set semsys:seminfo_semmni=1000
set semsys:seminfo_semmns=4000
set semsys:seminfo_semmnu=800
set semsys:seminfo_semmsl=512
* end Oracle *
```

Note: The settings for all tunable parameters depend on such factors as the size of your system and database, the database load, and the number of users. In some cases, we make suggestions for setting the parameters; however, you should always consult the Oracle Installation Guide for your system and version, and use the settings recommended by Oracle when provided.

---

maxuprc

This parameter sets the maximum number of processes that can be run concurrently by any one user. If you anticipate having a large number of users accessing the database concurrently, you may need to increase this parameter.

To increase the maxuprc parameter

1. Check the current setting for `maxuprc` as follows:

   ```
   # echo "maxuprc/D" | adb -k
   ```

2. Modify or add the `maxuprc` setting in the `/etc/system` file as follows:

   ```
   # set maxuprc=some_integer
   ```

shmmax

This parameter sets the maximum size (in bytes) of a single shared memory segment. See your database documentation for the recommended value.

shmmin

This parameter sets the minimum size (in bytes) of a single shared memory segment. See your Oracle documentation for the recommended value.

shmmni

This parameter sets the number of shared memory identifiers. See your database documentation for the recommended value.

shmseg

This parameter sets the maximum number of shared memory segments that can be attached by a process. See your database documentation for the recommended value.

semmap

This parameter sets the number of entries in semaphore map. The memory space given to the creation of semaphores is taken from `semmap`, which is initialized with a fixed number of map entries based on the value of `semmap`. The value of `semmap` should never be larger than `semmni`. See your database documentation for the recommended value.

semmni

This parameter sets the number of semaphore set identifiers in the system. The `semmni` parameter determines the number of semaphore sets that can be created at any one time, and may need to be set higher for a large database. See your database documentation for the recommended value.

semmns

This parameter sets the maximum number of semaphores in the system. The `semmns` parameter may need to be set higher for a large database. See your database documentation for the recommended value.

semmnu

This parameter sets the system-wide maximum number of undo structures. Setting this parameter value equal to `semmni` provides for an undo structure for every semaphore set. Semaphore operations performed using `semop`(2) can be undone if the process terminates, but an undo structure is required to guarantee it. See your database documentation for the recommended value of `semmnu`.

semmsl

This parameter sets the maximum number of semaphores that can be in one semaphore set. The `semmsl`parameter should be equal to the maximum number of Oracle processes. See your Oracle documentation for the recommended value.

Chapter 30

Troubleshooting SFDB tools

This chapter includes the following topics:

- About troubleshooting Storage Foundation for Databases (SFDB) tools
- About the vxdbd daemon
- Troubleshooting vxdbd
- Resources for troubleshooting SFDB tools
- Troubleshooting SmartTier for Oracle
- Upgrading Storage Foundation for Databases (SFDB) tools from 5.0.x to 7.4.2 (2184482)
- Troubleshooting Reverse Resynchronization

About troubleshooting Storage Foundation for Databases (SFDB) tools

Storage Foundation for Databases (SFDB) tools are deployed with several Storage Foundation products, and as a result can be affected by any issue with those products. The first step in case of trouble should be to identify the source of the problem. It is rare to encounter problems in Storage Foundation for Databases (SFDB) tools; more commonly the problem can be traced to setup issues or problems in the base products.

Use the information in this chapter to diagnose the source of problems. Indications may point to base product set up or configuration issues, in which case solutions may require reference to other Storage Foundation documentation. In cases where

indications point to a component product or to Oracle as the source of a problem, it may be necessary to refer to the appropriate documentation to resolve it.

For troubleshooting Storage Foundation product issues:

- *Storage Foundation Administrator's Guide*
- *Storage Foundation for Cluster File System High Availability Administrator's Guide*
- *Storage Foundation for Oracle RAC Administrator's Guide*

Running scripts for engineering support analysis for SFDB tools

Troubleshooting scripts gather information about the configuration and status of your product and its modules. The scripts identify package information, debugging messages, console messages, and information about disk groups and volumes. Forwarding the output of these scripts to Veritas Tech Support can assist with analyzing and solving any problems.

To obtain SFDB repository and log information

◆ Run:

```
# /opt/VRTSspt/VRTSexplorer/VRTSexplorer
```

Send the output to Support.

Storage Foundation for Databases (SFDB) tools log files

Checking the following log files can provide useful diagnostic information.

SFDB tools commands log files are located in the /var/vx/vxdba/logs directory.

About the vxdbd daemon

The SFDB commands are run as the DBA user. DBA users need to perform several operations, such as creating snapshots and mounting file systems, as the root user. The `vxdbd` daemon is used by the SFDB commands to run privileged commands, or when communicating with the SFDB repository on a different host.

Starting and stopping vxdbd

The `vxdbd` daemon is configured to automatically start when the system boots up. The script at /opt/VRTS/bin/sfae_config can be used to stop and start the daemon, and to query its status. Only the root user can start and stop the daemon.

To query the daemon status

- Run the command:

```
# /opt/VRTS/bin/sfae_config status
```

To start the daemon

- Run the command:

```
# /opt/VRTS/bin/sfae_config enable
```

To stop the daemon

- Run the command:

```
# /opt/VRTS/bin/sfae_config disable
```

Note: Most SFDB commands require that the `vxdbd` daemon be running.

Configuring listening port for the vxdbd daemon

The `vxdbd` daemon listens on TCP port 3233, by default. If this port is in use by some other application, `vxdbd` can be configured to listen on an alternate port. In cluster environments, `vxdbd` must be configured to listen on the same port on all the nodes.

To configure listening port for the vxdbd daemon

1. Stop the `vxdbd` daemon:

```
# /opt/VRTS/bin/sfae_config disable
```

2. Set VXDBD_PORT to the desired port number by editing the `/etc/vx/vxdbed/admin.properties` configuration file.
3. Start the `vxdbd` daemon:

```
# /opt/VRTS/bin/sfae_config enable
```

Limiting vxdbd resource usage

Although the `vxdbd` daemon is light-weight in terms of resource usage, system administrators might want to additionally restrict its resource usage. This can be

controlled by using two configuration parameters in `/etc/vx/vxdbed/admin.properties`:

- MAX_CONNECTIONS: This parameter controls the maximum number of simultaneous requests that vxdbd should allow.
- MAX_REQUEST_SIZE: This parameter controls the maximum size of a single request (in bytes).

Setting these parameters too low may cause SFDB commands to fail. The following are the suggested minimum values for these parameters:

- MAX_CONNECTIONS: 5
- MAX_REQUEST_SIZE: 1048576 (1 MB)

Note: Restart vxdbd after making any changes to these parameters for the changes to take effect.

Configuring encryption ciphers for vxdbd

Communication with the vxdbd daemon is encrypted. The encryption algorithms used for communication can be configured by editing the `/var/vx/vxdba/auth/vxdbd/root/.VRTSat/profile/VRTSatlocal.conf` configuration file. The SSLCipherSuite configuration parameter specifies the ciphers that are allowed when establishing a secure connection. Refer to the OpenSSL ciphers(1) man page for details on the acceptable values of this parameter.

For example, to configure `vxdbd` to deny communication using the medium-strength and low-strength ciphers, set the SSLCipherSuite parameter as follows:

```
"SSLCipherSuite"="HIGH:!MEDIUM:!eNULL:!aNULL:!SSLv2:!LOW"
```

Troubleshooting vxdbd

If the `vxdbd` daemon is not running, SFDB commands may fail with an error message. The following steps can help in troubleshooting the instances of the daemon that are down:

- Use the `/opt/VRTS/bin/sfae_config status` to verify that the daemon is running.
- If the output states that the daemon is not running, use the `/opt/VRTS/bin/sfae_config enable` command to start the daemon.

- If the daemon fails to start, verify that no other process is listening on the same port. If there is such a process, stop that process, or configure `vxdbd` to listen on a different port.
- The daemon generates logs in the file at `/var/vx/vxdba/logs/vxsfaed.log`. To enable verbose logging, edit the `/etc/vx/vxdbed/admin.properties` configuration file and set LOG_LEVEL to DEBUG.

Resources for troubleshooting SFDB tools

If Storage Foundation for Databases (SFDB) commands fail, use the following resources to troubleshoot.

See "SFDB logs" on page 369.

See "SFDB error messages" on page 370.

See "SFDB repository and repository files" on page 370.

SFDB logs

The SFDB commands generate logs that can be used to narrow down to the actual problem.

Log files:

- Log files are generated in the location `/var/vx/vxdba/logs`.
- There are two kind of logs:
  - User logs are generated in the <user> folder.
  - Logs from vxdbd and other root operations are generated in the logs folder.
- The user log files have the naming convention:
  log_<service>_<app>_<service_id><app_id>.log.
  A system.log is also present until `vxsfadm` can recognize the service and the application identifiers.
- The vxdbd logs have the name vxsfaed.log.
  A system.log also exists for all root operations performed.
- The log files are archived after they reach a threshold of 1MB and are backed up as
  log_<service><application><application_identifier><service_identifier>.log.<randomnumber>
  Every log file has a pointer to the previously archived log.

Log levels:

- Log levels can be set using the environment variable SFAE_LOG_LEVEL.

- The following additional environment variables can be set that override SFAE_LOG_LEVEL:
  - APP_LOG_LEVEL: Log application-specific operations.
  - SER_LOG_LEVEL: Log VxFS/VxVM stack specific operations.
  - REP_LOG_LEVEL: Log repository operations.
  - FSM_LOG_LEVEL: Log vxsfadm engine-specific operations.
- The log levels can be set to the following levels:
  - FATAL: Logs only fatal messages.
  - ERROR: Logs errors and above messages.
  - WARN: Logs warning and above messages.
  - INFO: Logs info and above messages.
  - DEBUG: Logs debug and above messages.
- The default log level is DEBUG.

Log messages:

- The actual log messages appear in the following format:

  ```
  yyyy/mm/dd hh:mm:ss: <loglevel> : <module> : <message>
  ```

  For example:

SFDB error messages

Each error message is based on a standard template wherein the following fields are displayed:

- MESSAGE: The actual error message.
- REASON: The reason for the error.
- ACTION: The action to be taken to correct the error.

These fields provide you with precise information about the cause of a problem.

SFDB repository and repository files

Caution: Any troubleshooting that involves operations related to the SFDB repository must be done under the supervision of a trained Veritas Engineer.

The name of the repository database is in the following format:
`dbed-<application>-<application identifier>-repository.db`.

For example: `dbed-oracle-SFAEDB-repository.db`.

The repository database can be queried to view a variety of information about an application instance. This includes the following tables:

- `_fsm_state_`: Displays the progress of various services.
- `_operational_data_`: Displays the various configuration values set for various services.
- `files`: Displays the files used by the services.

SFDB tools create files under `<repositorylocation>/files`.

- These files are used for various processing activities for all services.
- The `files` table from the repository points to the various file names used by a service under the `files` directory.

Troubleshooting SmartTier for Oracle

If the SmartTier for Oracle commands fail as in the following example, review the tsdb_debug.log.

The tsdb_debug.log is located at:

```
/var/vx/vxdba/logs/tsdb_debug.log
```

For example, when the following message appears after issuing a `dbdst_addvol` command review the tsdb_debug.log:

```
$ /opt/VRTS/bin/dbdst_addvol -S BLM21 -M /dev/vx/dsk/nobody_data1/ \
data1vol -v new_vol1:fast_storage,new_vol2:slow_storage,new_vol3:slow_storage

SFORA dbdst_addvol ERROR V-81-6222 Could not add volume new_vol1 to vset
```

The tsdb_debug.log file contains information that resembles the following:

```
# view /var/vx/vxdba/logs/tsdb_debug.log
1216606 Tue May 13 10:11:05 2008
 /opt/VRTS/bin/dbdst_addvol -S BLM21 -M /dev/vx/dsk/nobody_data1/data1vol
-v new _vol1:fast_storage,new_vol2:slow_storage,
new_vol3:slow_storage
1216606 Tue May 13 10:11:09 2008
RACmaster = system1
1216606 Tue May 13 10:11:09 2008
```

```
system1:/opt/VRTSdbed/common/.dba/tsdb_setup.sh -g nobody_data1 -o addvol
 -d\ data1vol - v new_vol1 -m /oradata1 -t vxfs.placement_class.FAST_STORAGE
1216606 Tue May 13 10:11:09 2008
command failed, ret=1

1216606 Tue May 13 10:11:09 2008
tsdb_setup.sh arguments -g nobody_data1 -o addvol -d data1vol -v
 new_vol1 -m /or
adata1 -t vxfs.placement_class.FAST_STORAGE
05/13/08@17:08:11
size of volume  new_vol1 is 204800
VxVM vxvset ERROR V-5-1-10035 Volume set data1vol contains volume(s)
in snapshot chain.
This can cause inconsistencies in the snapshot hierarchy. Specify "-f"
option to force the operation.
^^^^ NOTE: here is the reason for the failure, barried in this log file:
^^^^ /var/vx/vxdba/logs/tsdb_debug.log
Can not add  to data1vol, ERR 1
ERROR:1
```

Upgrading Storage Foundation for Databases (SFDB) tools from 5.0.x to 7.4.2 (2184482)

The `sfua_rept_migrate`command results in an error message after upgrading SFHA or SF for Oracle RAC version 5.0 or 5.0MP3 to SFHA or SF for Oracle RAC 7.4.2.

The `sfua_rept_migrate`command results in an error message after upgrading SFHA or SF for Oracle RAC version 5.0 to SFHA or SF for Oracle RAC 7.4.2.

When upgrading from SF Enterprise products version 5.0 or 5.0MP3 to SF Enterprise products 7.4.2 the S*vxdbms3 startup script is renamed to NO_S*vxdbms3. The S*vxdbms3 startup script is required by `sfua_rept_upgrade`. Thus when `sfua_rept_upgrade` is run, it is unable to find the S*vxdbms3 startup script and gives the error message:

```
/sbin/rc3.d/S*vxdbms3 not found
SFORA sfua_rept_migrate ERROR V-81-3558 File:  is missing.
SFORA sfua_rept_migrate ERROR V-81-9160 Failed to mount repository.
```

Workaround: Before running `sfua_rept_migrate`, rename the startup script NO_S*vxdbms3 to S*vxdbms3.

Troubleshooting Reverse Resynchronization

Before preparing a preview session for a backup database instance the Reverse Resync Begin operation always creates a recovery point. If this recovery point is lost, then you cannot transit from the Reverse Resync Begin state.

The following steps can help in troubleshooting the instance and let you transit to the Reverse Resync Begin state:

1. Make sure to stop the Oracle database instance using **SQL** > **Shutdown immediate**.
2. Unmount all the snapshot volumes. The snapshot volume names are prefixed with a snapshot prefix as specified in the snapshot configuration file. The default value of the prefix is "SNAP_".

   For example:

   ```
   # umount /data/prodDB
   ```

3. Deport snapshot disk group(s). The snapshot disk group names are prefixed with a snapshot prefix as specified in the snapshot configuration file.

   For example:

   ```
   # vxdg deport SNAP_proddg
   ```

4 Mount the primary data volumes.

For example:

```
#mount -t vxfs /dev/vx/dsk/proddg/dbvol1 /data/prodDB
```

5 Restore the backup repository to switch to a previous state.

The RRBegin operation automatically backs up the SFDB repository.

```
[oracle@testbox ~]$ sfua_rept_util -S $ORACLE_SID -o list
Listing Repository Backups for SID='RR'
TIME                    SIZE            NAME
2013-07-30-13-43-55     18160           rept_RR_2013-07-30-13-43-55.tgz
2013-07-30-16-16-25     18159           rept_RR_2013-07-30-16-16-25.tgz
2013-07-30-16-16-49     18161           rept_RR_2013-07-30-16-16-49.tgz
2013-07-30-16-17-56     18163           rept_RR_2013-07-30-16-17-56.tgz
2013-07-30-16-18-01     18163           rept_RR_2013-07-30-16-18-01.tgz
2013-07-30-16-18-07     18161           rept_RR_2013-07-30-16-18-07.tgz

[oracle@testbox ~]$ sfua_rept_util -S $ORACLE_SID -o restore -f
rept_RR_2013-07-30-16-18-07.tgz
Restoring repository for SID='RR'
Target directory='/etc/vx/vxdba/logs/oracle/backup'
Restore Repository for SID='RR' Completed
```

Chapter 31

Manual recovery of Oracle database

This chapter includes the following topics:

- Manual recovery of a clone database
- Manual recovery after a restore operation

Manual recovery of a clone database

Storage Foundation for Databases (SFDB) tools let you clone an application based on a point-in-time copy.

Application-level recovery required to clone an application, in the case of online and instant mode snapshots, is normally automatically done whenever a clone operation is performed. However, you may want to manually perform Oracle recovery for the clone database instance. Manual recovery might be required for reasons such as performing a point-in-time recovery of the clone database to a point after the time when the snapshot was taken.

To do manual recovery, use the preclone or the mountdb operation instead of the clone operation. The preclone operation mounts the clone database and lets you perform database recovery manually to open the database.

Note: A manual recovery option is only available, and advised, when using the online snapshot mode. Offline snapshots do not require any recovery and the database can be directly opened with RESETLOGS.

Use the following series of SQL statements to perform manual recovery:

```
1) ALTER DATABASE RECOVER UNTIL CANCEL USING BACKUP CONTROLFILE;
 OR

 ALTER DATABASE RECOVER UNTIL TIME <time> USING BACKUP CONTROLFILE;
 OR

 ALTER DATABASE RECOVER UNTIL CHANGE <SCN> USING BACKUP CONTROLFILE;

2) ALTER DATABASE RECOVER LOGFILE '<archived redo log file>';
{ Repeat this step until the desired recovery point/target is reached.}
{The recovery target must be after the time the snapshot was taken.}

3) ALTER DATABASE RECOVER CANCEL;
{ Use if cancel-based recovery was initiated in step 1.}
```

Once manual recovery is done, you can open the database with the RESETLOGS option. The data files belonging to read-only or offline tablespaces can then be added so that read-only or offline tablespaces are added back into the clone database. Temp files may also be added to the clone database to provide storage for the temporary tablespaces.

Manual recovery after a restore operation

Storage Checkpoints and FileSnap services of Storage Foundation for Databases (SFDB) tools provide a restore operation that restores application files from a point-in-time copy. In the case of Oracle, a restore operation restores the database files from the point-in-time copy.

The restore operation restores only the data files onto the primary database data files, thus overwriting them. It lets you then perform a database point-in-time recovery manually to recover the database to the desired recovery target or time. The restore operation also places a copy of the control file from the snapshot in the `/var/tmp` directory to you to perform point-in-time recovery to the snapshot time.

Note: You can perform a restore operation only with an online or an offline mode snapshot.

After performing a restore operation, use the following SQL statements to recover the database to any point-in-time, starting from the time the snapshot was taken:

```
{ To perform complete recovery:}
1) SET AUTORECOVERY ON;
```

```
2) RECOVER DATABASE;

{ To perform incomplete recovery:}

1) SET AUTORECOVERY ON;

2) RECOVER DATABASE UNTIL CANCEL;

OR

RECOVER DATABASE UNTIL TIME <time>;

OR

RECOVER DATABASE UNTIL CHANGE <SCN>;

{ Note: The 'USING BACKUP CONTROLFILE' clause must be used
if recovery is performed using a backup control file, or if using
the control file from the snapshot.}

3) ALTER DATABASE OPEN RESETLOGS;
```

Chapter 32

Storage Foundation for Databases command reference for the releases prior to 6.0

This chapter includes the following topics:

- About SFDB commands backward compatibility
- Storage Foundation for Databases (SFDB) tools features which are no longer supported
- Preparing storage for Database FlashSnap
- About creating database snapshots
- FlashSnap commands
- Guidelines for Oracle recovery
- Database Storage Checkpoint Commands

About SFDB commands backward compatibility

The following sections contain information about the Storage Foundation for Databases (SFDB) tools commands from the previous release. Veritas recommends using the `vxsfadm` CLI. The commands from the previous release are supported for backward compatibility and they will be deprecated in a subsequent release.

Storage Foundation for Databases (SFDB) tools features which are no longer supported

The following Storage Foundation for Databases (SFDB) tools features are not supported in this release:

- Storage Checkpoint policy and Storage Checkpoint quotas
- Interactive modes in clone and rollback

Preparing storage for Database FlashSnap

The following sections contain information about preparing storage for Database FlashSnap.

Example procedure to create a snapshot mirror of a volume

This example shows the steps involved in creating a snapshot mirror for the volume data_vol belonging to the disk group PRODdg.

Note: You must be logged in as superuser (root) to issue the commands in the following procedure.

To create a snapshot mirror of the volume data_vol

1. Prepare the volume data_vol for mirroring:

```
# vxsnap -g PRODdg prepare data_vol alloc=PRODdg01,PRODdg02
```

2. Verify that FastResync is enabled:

```
# vxprint -g PRODdg -F%fastresync data_vol
on
```

3. Verify that a DCO and a DCO log are attached to the volume:

```
# vxprint -g PRODdg -F%hasdcolog data_vol
on
```

4. Create a snapshot mirror of data_vol:

```
# vxsnap -g PRODdg addmir data_vol alloc=PRODdg02
```

5 List the data plexes:

```
# vxprint -g PRODdg -F%name -e"pl_v_name in \"data_vol\""

data_vol-01

data_vol-02
```

6 Choose the plex that is in the SNAPDONE state. Use the `vxprint -g diskgroup` command to identify the plex that is in the SNAPDONE state.

7 Identify the plex name in the above step and set the dbed_flashsnap tag for it:

```
# vxedit -g PRODdg set putil2=dbed_flashsnap data_vol-02
```

8 Verify that the `dbed_flashsnap` tag has been set to the desired data plex, data_vol-02:

```
# vxprint -g PRODdg -F%name -e"pl_v_name in \"data_vol\" \
&& p2 in \"dbed_flashsnap\""
data_vol-02
```

9 To verify that the snapshot volume was created successfully, use the `vxprint -g dg` command as follows:

```
# vxprint -g PRODdg

# vxprint -g flashdgso

TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg flashdgso    flashdgso    -        -        -        -        -       -

dm hdisk5       hdisk5       -        10411776 -        -        -       -
dm hdisk6       hdisk6       -        10411776 -        -        -       -
dm hdisk8       hdisk8       -        10411776 -        -        -       -
dm hdisk10      hdisk10      -        10411776 -        -        -       -
dm hdisk14      hdisk14      -        10411776 -        -        -       -
dm hdisk17      hdisk17      -        10411776 -        -        -       -

v  archvol      fsgen        ENABLED  10237952 -        ACTIVE   -       -
pl archvol-01   archvol      ENABLED  10237952 -        ACTIVE   -       -
sd hdisk6-01    archvol-01   ENABLED  10237952 0        -        -       -
pl archvol-02   archvol      ENABLED  10237952 -        SNAPDONE -       -
sd hdisk8-01    archvol-02   ENABLED  10237952 0        -        -       -
dc archvol_dco  archvol      -        -        -        -        -       -
v  archvol_dcl  gen          ENABLED  1120     -        ACTIVE   -       -
pl archvol_dcl-01 archvol_dcl ENABLED 1120     -        ACTIVE   -       -
sd hdisk14-01   archvol_dcl-01 ENABLED 1120    0        -        -       -
pl archvol_dcl-02 archvol_dcl DISABLED 1120    -        DCOSNP   -       -
sd hdisk8-02    archvol_dcl-02 ENABLED 1120    0        -        -       -

v  datavol      fsgen        ENABLED  10237952 -        ACTIVE   -       -
pl datavol-01   datavol      ENABLED  10237952 -        ACTIVE   -       -
sd hdisk5-01    datavol-01   ENABLED  10237952 0        -        -       -
pl datavol-02   datavol      ENABLED  10237952 -        SNAPDONE -       -
sd hdisk10-01   datavol-02   ENABLED  10237952 0        -        -       -
dc datavol_dco  datavol      -        -        -        -        -       -
v  datavol_dcl  gen          ENABLED  1120     -        ACTIVE   -       -
pl datavol_dcl-01 datavol_dcl ENABLED 1120     -        ACTIVE   -       -
sd hdisk17-01   datavol_dcl-01 ENABLED 1120    0        -        -       -
pl datavol_dcl-02 datavol_dcl DISABLED 1120    -        DCOSNP   -       -
sd hdisk10-02   datavol_dcl-02 ENABLED 1120    0        -        -       -

TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0 PUTIL0
dg PRODdg       PRODdg       -        -        -        -        -       -
dm PRODdg01     Disk_1       -        71117760 -        -        -       -
```

```
dm PRODdg02     Disk_2       -        71117760 -        -        -       -
dm PRODdg03     Disk_3       -        71117760 -        -        -       -

TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg PRODdg       PRODdg       -        -        -        -        -       -
dm PRODdg01     c1t2d0s2     -        35358848 -        -        -       -
dm PRODdg02     c1t3d0s2     -        17674896 -        -        -       -
dm PRODdg03     c1t1d0s2     -        17674896 -        -        -       -

      v  data_vol     fsgen
      ENABLED        4194304             -        ACTIVE   -       -
      pl data_vol-01  data_vol
      ENABLED        4194304             -        ACTIVE   -       -
      sd PRODdg03-01  data_vol-01
      ENABLED        4194304             0        -        -       -
      pl data_vol-02  data_vol
      ENABLED        4194304             -        SNAPDONE -       -
      sd PRODdg02-01  data_vol-02
      ENABLED        4194304             0        -        -       -
      dc data_vol_dco data_vol
      -              -                   -        -        -       -
      v  data_vol_dcl gen
      ENABLED        560                 -        ACTIVE   -       -
      pl data_vol_dcl-01 data_vol_dcl ENABLED
      560         -           ACTIVE   -       -
      sd PRODdg01-01  data_vol_dcl-01 ENABLED
      560         0           -        -       -
      pl data_vol_dcl-02 data_vol_dcl DISABLED
      560         -           DCOSNP   -       -
      sd PRODdg02-02  data_vol_dcl-02 ENABLED
      560         0           -        -       -
```

Identify that the specified plex is in the SNAPDONE state. In this example, it is data_vol-02.

The snapshot mirror is now ready to be used.

Upgrading existing volumes to use Veritas Volume Manager 7.4.2

The procedure in this section describes how to upgrade a volume created using a version older than VxVM 5.0 so that it can take advantage of FlashSnap and space-optimized snapshots.

Note the following requirements and caveats for this procedure:

- The plexes of the DCO volume require persistent storage space on disk to be available. To make room for the DCO plexes, you may need to add extra disks to the disk group, or reconfigure existing volumes to free up space in the disk group. Another way to add disk space is to use the disk group move feature to bring in spare disks from a different disk group.
- Existing snapshot volumes created by the `vxassist` command are not supported. A combination of snapshot volumes created by `vxassist` and `vxsnap` are also not supported.
- You must be logged in as superuser (root) to issue the commands in the following procedure.

To upgrade an existing volume created with an earlier version of VxVM

1. Upgrade the disk group that contains the volume to version 120 or higher before performing the remainder of the procedure described in this section. Use the following command to check the version of a disk group:

   ```
   # vxdg list diskgroup
   ```

   To upgrade a disk group to the latest version, use the following command:

   ```
   # vxdg upgrade diskgroup
   ```

2. If the volume to be upgraded has a DRL plex or subdisk from an earlier version of VxVM, use the following command to remove this:

   ```
   # vxassist [-g diskgroup] remove log volume [nlog=n]
   ```

   Use the optional attribute nlog=*n* to specify the number, *n*, of logs to be removed. By default, the `vxassist` command removes one log.

3. For a volume that has one or more associated snapshot volumes, use the following command to reattach and resynchronize each snapshot:

   ```
   # vxsnap [-g diskgroup] snapback snapvol
   ```

   If persistent FastResync was enabled on the volume before the snapshot was taken, the data in the snapshot plexes is quickly resynchronized from the original volume. If persistent FastResync was not enabled, a full resynchronization is performed.

4 Use the following command to turn off persistent FastResync for the volume:

```
# vxvol [-g diskgroup] set fastresync=off volume
```

5 Use the following command to dissociate a DCO object from an earlier version of VxVM, DCO volume and snap objects from the volume:

```
# vxassist [-g diskgroup] remove log volume logtype=dco
```

6 Use the following command on the volume to upgrade it:

```
# vxsnap [-g diskgroup] prepare volume \
alloc="disk_name1,disk_name2"
```

Provide two disk names to avoid overlapping the storage of the snapshot DCO plex with any other non-moving data or DCO plexes.

The `vxsnap prepare` command automatically enables persistent FastResync on the volume and on any snapshots that are generated from it. It also associates a DCO and DCO log volume with the volume to be snapshot.

7 To view the existing DCO plexes and see whether there are enough for the existing data plexes, enter:

```
# vxprint -g diskgroup
```

There needs to be one DCO plex for each existing data plex.

8 If there are not enough DCO plexes for the existing data plexes, create more DCO plexes:

```
# vxsnap [-g diskgroup] addmir dco_volume_name \
[alloc=disk_name]
```

where *dco_volume_name* is the name of the DCO volume you are creating.

9 If the plex is in a SNAPDONE state, convert it to an ACTIVE state:

```
# vxplex [-g diskgroup] convert state=ACTIVE data_plex
```

10 Convert the data plexes to a SNAPDONE state and associate a DCO plex with the data plex that will be used for snapshot operations:

```
# vxplex [-g diskgroup] -o dcoplex=dco_plex_name convert \
state=SNAPDONE data_plex
```

where *dco_plex_name* is the name of the DCO plex you are creating.

Example procedure to upgrade existing volumes to use Veritas Volume Manager 7.4.2

Note: You must be logged in as superuser (root) to issue the commands in the following procedure.

In this example, the volume, `data_vol`, is upgraded to make use of VxVM 7.4.2 features.

To upgrade an existing volume created with an earlier version of VxVM

1 Upgrade the disk group, PRODdg.

```
# vxdg upgrade PRODdg
```

2 Remove the DRL plexes or subdisks belonging to an earlier version of VxVM from the volume to be upgraded.

```
# vxassist -g PRODdg remove log data_vol logtype=drl
```

3 Reattach any snapshot volume back to the primary volume to be upgraded.

```
# vxsnap -g PRODdg snapback SNAP-data_vol
```

4 Turn off FastResync on the volume to be upgraded.

```
# vxvol -g PRODdg set fastresync=off data_vol
```

5 Disassociate and remove any older DCO object and DCO volumes.

```
# vxassist -g PRODdg remove log data_vol logtype=dco
```

6 Upgrade the volume by associating a new DCO object and DCO volume.

```
# vxsnap -g PRODdg prepare data_vol alloc="PRODdg01,PRODdg02"
```

7 View the existing DCO plexes and plex state.

Scenario 1

In this scenario, there are enough DCO plexes for the data plexes. Also, no data plex is associated with a DCO plex.

```
# vxprint -g PRODdg
```

```
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg PRODdg       PRODdg       -        -        -        -        -       -
dm PRODdg01     c1t2d0s2     -        35358848 -        -        -       -
dm PRODdg02     c1t3d0s2     -        17674896 -        -        -       -
dm PRODdg03     c1t1d0s2     -        17674896 -        -        -       -

v  data_vol     fsgen
ENABLED        4194304          -        ACTIVE   -       -
pl data_vol-01  data_vol
ENABLED        4194304          -        ACTIVE   -       -
sd PRODdg01-01  data_vol-01
ENABLED        4194304          0        -        -       -
pl data_vol-04  data_vol
ENABLED        4194304          -        SNAPDONE -       -
sd PRODdg02-03  data_vol-04
ENABLED        4194304          0        -        -       -
dc data_vol_dco data_vol
-              -                -        -        -       -
v  data_vol_dcl gen
ENABLED       560               -        ACTIVE   -       -
pl data_vol_dcl-01 data_vol_dcl
ENABLED       560               -        ACTIVE   -       -
sd PRODdg01-02  data_vol_dcl-01
ENABLED       560               0        -        -       -
pl data_vol_dcl-02 data_vol_dcl
ENABLED       560               -        ACTIVE   -       -
```

```
sd PRODdg02-02  data_vol_dcl-02
ENABLED       560               0        -        -       -
```

- Convert the data plex state from SNAPDONE to ACTIVE.

```
# vxplex -g PRODdg convert state=ACTIVE data_vol-04
```

- Associate the data plex with a new DCO plex and convert it back to a SNAPDONE state.

```
# vxplex -g PRODdg -o dcoplex=data_vol_dcl-02 \
convert state=SNAPDONE data_vol-04

# vxprint -g PRODdg
```

```
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg PRODdg       PRODdg       -        -        -        -        -       -
dm PRODdg01     c1t2d0s2     -        35358848 -        -        -       -
dm PRODdg02     c1t3d0s2     -        17674896 -        -        -       -
dm PRODdg03     c1t1d0s2     -        17674896 -        -        -       -

pl data_vol-03  -
DISABLED       4194304          -         -        -       -
sd PRODdg02-01  data_vol-03
ENABLED        4194304          0         -        -       -

v  data_vol     fsgen
ENABLED        4194304          -         ACTIVE   -       -
pl data_vol-01  data_vol
ENABLED        4194304          -         ACTIVE   -       -
sd PRODdg01-01  data_vol-01
ENABLED        4194304          0         -        -       -
pl data_vol-04  data_vol
ENABLED        4194304          -         SNAPDONE -       -
sd PRODdg02-03  data_vol-04
ENABLED        4194304          0         -        -       -
dc data_vol_dco data_vol
-              -                -         -        -       -
v  data_vol_dcl gen
ENABLED        560              -         ACTIVE   -       -
pl data_vol_dcl-01 data_vol_dcl
ENABLED        560              -         ACTIVE   -       -
```

```
sd PRODdg01-02  data_vol_dcl-01
ENABLED        560              0         -        -       -
pl data_vol_dcl-02 data_vol_dcl
DISABLED       560              -         DCOSNP   -       -
sd PRODdg02-02  data_vol_dcl-02
ENABLED        560              0         -        -       -
```

Scenario 2

In this scenario, there are fewer DCO plexes than data plexes.

```
# vxprint -g PRODdg


TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg PRODdg       PRODdg       -        -        -        -        -       -
dm PRODdg01     c1t2d0s2     -        35358848 -        -        -       -
dm PRODdg02     c1t3d0s2     -        17674896 -        -        -       -
dm PRODdg03     c1t1d0s2     -        17674896 -        -        -       -

pl data_vol-03  -
DISABLED       4194304       -        -        -       -
sd PRODdg02-01  data_vol-03
ENABLED        4194304       0        -        -       -
v  data_vol     fsgen
ENABLED        4194304       -        ACTIVE   -       -
pl data_vol-01  data_vol
ENABLED        4194304       -        ACTIVE   -       -
sd PRODdg01-01  data_vol-01
ENABLED        4194304       0        -        -       -
pl data_vol-04  data_vol
ENABLED        4194304       -        ACTIVE   -       -
sd PRODdg02-03  data_vol-04
ENABLED        4194304       0        -        -       -
dc data_vol_dco data_vol
-              -             -        -        -       -
v  data_vol_dcl gen
ENABLED        560           -        ACTIVE   -       -
pl data_vol_dcl-01 data_vol_dcl
ENABLED        560           -        ACTIVE   -       -
sd PRODdg01-02  data_vol_dcl-01
ENABLED        560           0        -        -       -
```

- Add a DCO plex to the DCO volume using the vxassist mirror command.

```
# vxsnap -g PRODdg addmir data_vol_dcl alloc=PRODdg02
```

- Associate the data plex with the new DCO plex and convert it to a SNAPDONE state.

```
# vxplex -g PRODdg -o dcoplex=data_vol_dcl-02 \
  convert state=SNAPDONE data_vol-04
```

```
TY NAME         ASSOC        KSTATE   LENGTH   PLOFFS   STATE    TUTIL0  PUTIL0
dg PRODdg       PRODdg       -        -        -        -        -       -
dm PRODdg01     c1t2d0s2     -        35358848 -        -        -       -
dm PRODdg02     c1t3d0s2     -        17674896 -        -        -       -
dm PRODdg03     c1t1d0s2     -        17674896 -        -        -       -

                         pl data_vol-03  -
                         DISABLED       4194304       -        -        -       -
                         v  data_vol     fsgen
                         ENABLED        4194304       -        ACTIVE   -       -
                         pl data_vol-01  data_vol
                         ENABLED        4194304       -        ACTIVE   -       -
                         sd PRODdg01-01  data_vol-01
                         ENABLED        4194304       0        -        -       -
                         pl data_vol-04  data_vol
                         ENABLED        4194304       -        SNAPDONE -       -
                         sd PRODdg02-03  data_vol-04
                         ENABLED        4194304       0        -        -       -
                         dc data_vol_dco data_vol
                         -              -             -        -        -       -
                         v  data_vol_dcl gen
                         ENABLED        560           -        ACTIVE   -       -
                         pl data_vol_dcl-01 data_vol_dcl
                         ENABLED        560           -        ACTIVE   -       -
                         sd PRODdg01-02  data_vol_dcl-01
                         ENABLED        560           0        -        -       -
                         pl data_vol_dcl-02 data_vol_dcl
                         DISABLED       560           -        DCOSNP   -       -
                         sd PRODdg02-02  data_vol_dcl-02
                         ENABLED        560           0        -        -       -
```

Updating after structural changes to the database

In case of any structural change to database, for instance adding or removing any database objects such as control files, data files , arch location etc., you must execute `dbed_update` on the node to refresh the SFDB repository. If there are any flashsnaps which are taken before the database modification, you must perform the following steps:

To update a flashsnap after structural change to a database

1. Resynchronize the snapshots using the `dbed_vmsnap` utility.
2. Run `dbed_update`.
3. Validate the snapplan again and take new snapshot.

About creating database snapshots

A snapshot can be a source for backing up the database or creating a clone database for decision-support purposes. You can use Database FlashSnap commands to create a snapshot of your entire database on the same host (node) or on a different one.

Online database snapshots

Table 32-1 describes the three types of snapshots that can be created.

Table 32-1 Database snapshot types

Database snapshot type	Description
online	If the SNAPSHOT_MODE specified in the snapplan is set to online, the `dbed_vmsnap` command first puts the tablespaces to be snapshot into backup mode. After the snapshot is created, the tablespaces are taken out of backup mode, the log files are switched to ensure that the extra redo logs are archived, and a snapshot of the archive logs is created. Both online and offline snapshots provide a valid backup copy of the database.
offline	If the SNAPSHOT_MODE is set to offline, the database must be shut down before the snapshot is created. Online redo logs and control files are required and will be used to ensure a full database recovery.

Table 32-1 Database snapshot types *(continued)*

Database snapshot type	Description
instant	If the SNAPSHOT_MODE is set to instant, tablespaces are not put into and out of backup mode. Online redo logs and control files are required and will be used to ensure a full database recovery. Instant snapshots do not represent a valid backup copy for point-in-time recovery.

Note: For Storage Foundation for Oracle RAC, only the online snapshot mode is supported.

Database FlashSnap supports online database snapshot types.

When the SNAPSHOT_MODE specified in the snapplan is set to online, the `dbed_vmsnap`command first puts the tablespaces to be snapshot into backup mode. After the snapshot is created, the tablespaces are taken out of backup mode, the log files are switched to ensure that the extra redo logs are archived, and a snapshot of the archive logs is created.

Online snapshots provide a valid backup copy of the database.

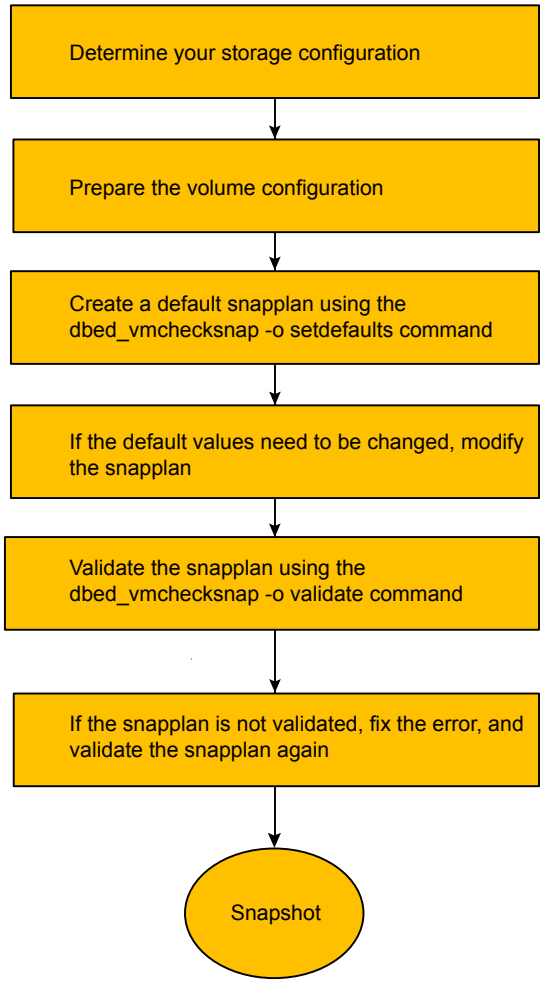
Tasks before creating a snapshot

Review the details on how to create snapshots of all volumes on a database using the snapplan.

Optionally, you can use the VxVM command (`vxsnap`) to create volume snapshots. However, unlike the Database FlashSnap commands, the `vxsnap` command does not automate disk group content reorganization functions.

Figure 32-1 depicts the sequence of steps leading up to taking a snapshot using Database FlashSnap.

Figure 32-1 Prerequisites for creating a snapshot of your database



Creating a snapshot

Make sure the volumes used by the database are configured properly before attempting to take a snapshot. This database configuration requires superuser (root) privileges.

Note: Database FlashSnap commands must be run by the Oracle database administrator.

Whenever you change the structure of the database (for example, by adding or deleting datafiles, converting PFILE to SPFILE, or converting SPFILE to PFILE), you must run `dbed_update`. For example:

```
$ /opt/VRTS/bin/dbed_update -S $ORACLE_SID -H $ORACLE_HOME
```

To create a snapshot image of a database

1. Create a snapshot mirror of a volume or volume set.

   See "To create a snapshot mirror of a volume or volume set" on page 191.

2. Use the `dbed_vmchecksnap` command to create a snapplan template and check the volume configuration to ensure that it is valid for creating volume snapshots of the database.

   The snapplan contains detailed database and volume configuration information that is needed for snapshot creation and resynchronization. You can modify the snapplan template with a text editor.

   The `dbed_vmchecksnap` command can also be used to:

List all snapplans associated with a specific ORACLE_SID	`dbed_vmchecksnap -o list`
Remove the snapplan from the SFDB repository	`dbed_vmchecksnap -o remove -f SNAPPLAN`
Copy a snapplan from the SFDB repository to your local directory	`dbed_vmchecksnap -o copy -f SNAPPLAN`

3. Use the `dbed_vmsnap` command to create snapshot volumes for the database.

   See "Creating a snapshot mirror of a volume or volume set used by the database" on page 189.

4. On the secondary host, use the `dbed_vmclonedb` command to create a clone database using the disk group deported from the primary host. For more information:

   See "Cloning a database on the secondary host" on page 204.

   If the primary and secondary hosts specified in the snapplan are different, the `dbed_vmclonedb` command takes the following actions:

   - Imports the disk group that was deported from the primary host
   - Recovers the snapshot volumes
   - Mounts the file systems

- Recovers the database
- Brings the database online with a different Oracle SID name than the primary host.

You can use the `-o recoverdb` option to let `dbed_vmclonedb` perform an automatic database recovery, or you can use the `-o mountdb` option to perform your own point-in-time recovery and bring up the database manually. For a point-in-time recovery, the snapshot mode must be online.

You can also create a clone on the primary host. Your snapplan settings specify whether a clone should be created on the primary or secondary host.

5. You can now use the clone database to perform database backup and other off-host processing work.
6. For single instance Oracle, the snapshot volumes can be used to reverse resynchronize the original volume from the data in the snapshot, or can be discarded by rejoining the snapshot volumes with the original volumes (that is, by resynchronizing the snapshot volumes) for future use.
7. The clone database can be discarded by rejoining the snapshot volumes with the original volumes (that is, by resynchronizing the snapshot volumes) for future use.

Tasks after creating a snapshot

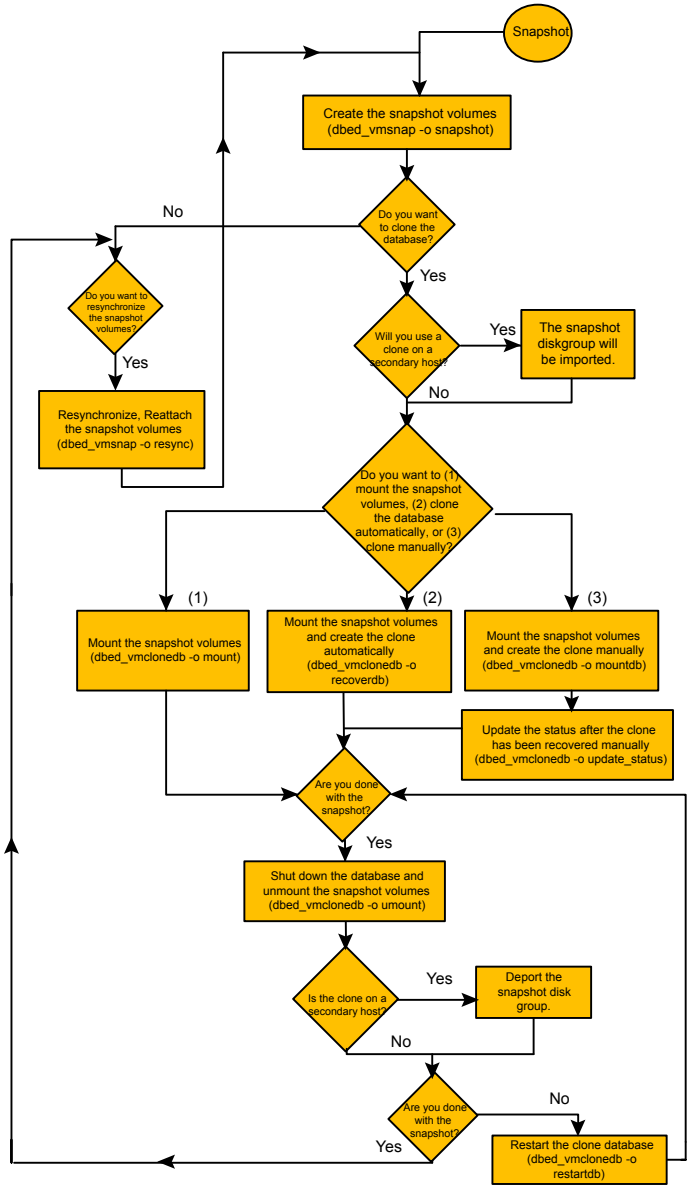
There are many actions you can take after creating a snapshot of your database using Database FlashSnap. You can create a clone of the database for backup and off-host processing purposes. You can resynchronize the snapshot volumes with the primary database.

For single instance Oracle, in the event of primary database failure, you can recover it by reverse resynchronizing the snapshot volumes.

The following flow chart depicts the actions you can perform after creating a snapshot of your database using Database FlashSnap, and involve the following four questions:

Figure 32-2 is a flow chart that depicts the actions you can perform after creating a snapshot of your database using Database FlashSnap.

Figure 32-2 Actions you can perform after creating a snapshot of your database

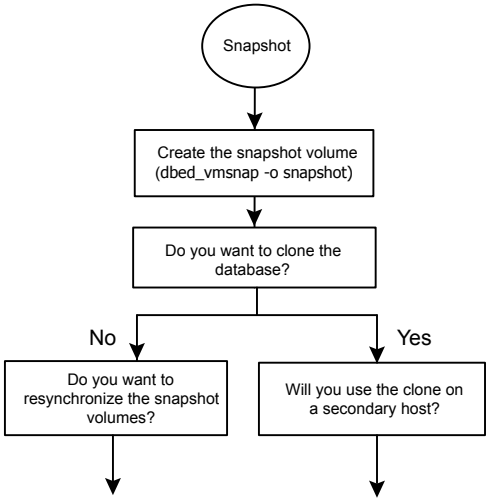


Do you want to clone the database?

After creating the snapshot volumes, you can proceed to clone the database and use the clone on a secondary host or resynchronize the snapshot volumes.

Figure 32-3 is a flow chart of this process.

Figure 32-3 Clone the database

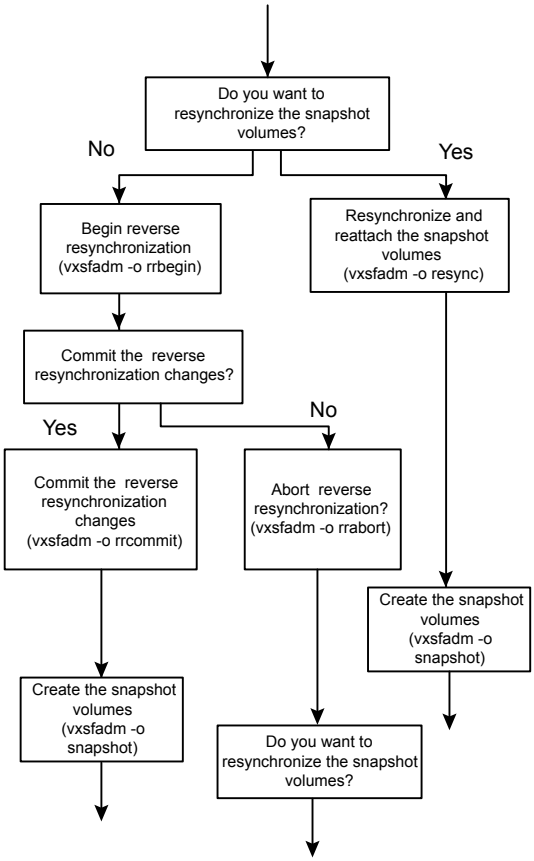


Do you want to resynchronize the snapshot volumes?

If you decide not to clone the database, then you can proceed to resynchronize and reattach the snapshot volumes, or begin a reverese resynchronization process.

Figure 32-4 is a flow chart of this process.

Figure 32-4 Resynchronize the database



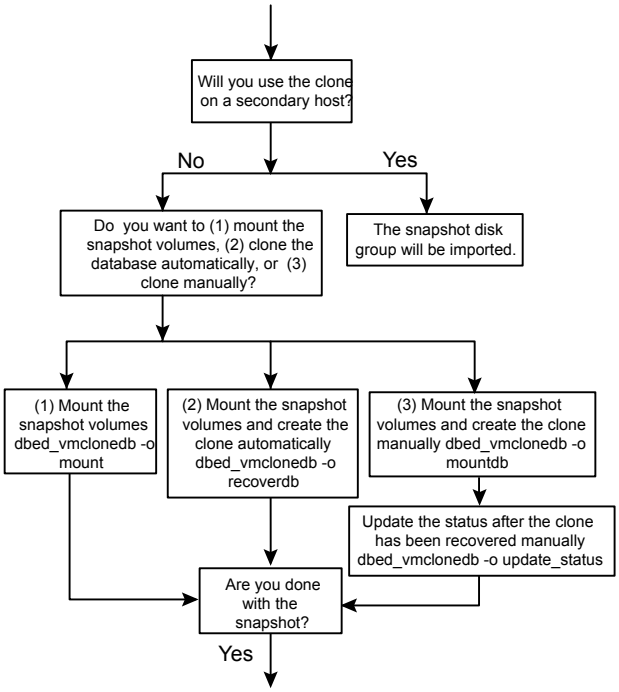
Will you use the clone on a secondary host?

If you decide to clone the database and use the clone on a secondary host, then the snapshot diskgroup will be imported. If not, then proceed with one of the following three options:

- Mount the snapshot volumes
- Mount the snapshot volumes and create the clone automatically
- Mount the snapshot volumes and create the clone manually

Figure 32-5 is a flow chart of this process.

Figure 32-5 Clone on a secondary host

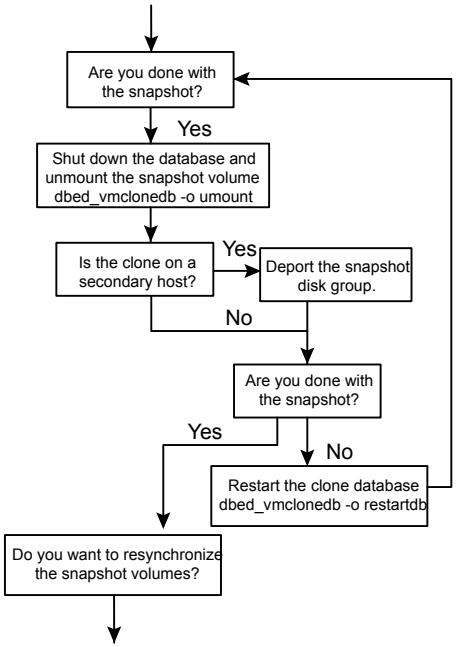


Are you done with the snapshot?

If you are done with the snapshot, then shut down the database and unmount the snapshot volumes. If the clone is on a secondary host, then deport the snapshot diskgroup.

Figure 32-6 is a flow chart of these processes.

Figure 32-6 Finished with the snapshot



FlashSnap commands

Database FlashSnap actions can be performed by using the following FlashSnap commands described in the following sections:

- Creating a snapplan (dbed_vmchecksnap)
- Validating a snapplan (dbed_vmchecksnap)
- Displaying, copying, and removing a snapplan (dbed_vmchecksnap)
- Creating a snapshot (dbed_vmsnap)
- Backing up the database from snapshot volumes (dbed_vmclonedb)
- Cloning a database (dbed_vmclonedb)
- Resynchronizing the snapshot to your database
- Removing a snapshot volume

Creating a snapplan (dbed_vmchecksnap)

The `dbed_vmchecksnap` command creates a snapplan that `dbed_vmsnap` uses to create a snapshot of an Oracle database.

The snapplan specifies snapshot scenarios: online, offline, or instant.

You can name a snapplan file whatever you choose. Each entry in the snapplan file is a line in parameter=argument format.

Table 32-2 describes the parameters that can be set when using the `dbed_vmchecksnap` command to create or validate a snapplan.

Table 32-2 Parameter values for dbed_vmchecksnap

Parameter	Value
SNAPSHOT_VERSION	Specifies the snapshot version for this major release of SF Enterprise products.
PRIMARY_HOST	The name of the host where the primary database resides.
SECONDARY_HOST	The name of the host where the database will be imported.
PRIMARY_DG	Colon-seperated list of names of the VxVM disk group's used by the primary database
SNAPSHOT_DG_PREFIX	The name of the prefix attached to the disk group name. A snapshot disk group's name is a concatination of SNAPSHOT_DG_PREFIX and the corresponding primary dg name. Its default value will be "SNAP_". The snapshot volumes will be put into this disk group on the primary host and deported. The secondary host will import this disk group to start a clone database. SNAPSHOT_DG is not supported in release 5.1. It is replaced by SNAPSHOT_DG_PREFIX.
ORACLE_SID	The name of the Oracle database. By default, the name of the Oracle database is included in the snapplan.

Table 32-2 Parameter values for dbed_vmchecksnap *(continued)*

Parameter	Value
ARCHIVELOG_DEST	The full path of the archive logs. There are several archive log destinations that can be used for database recovery if you are multiplexing the archive logs. You must specify which archive log destination to use. It is recommended that you have the archive log destination on a separate volume if SNAPSHOT_ARCHIVE_LOG is yes.
SNAPSHOT_ARCHIVE_LOG	yes or no Specifies whether to create a snapshot of the archive log volumes. Specify yes to split the archive log volume mirrors and deport them to the secondary host. When using the Oracle remote archive log destination feature to send the archive logs to the secondary host, you can specify no to save some space. Because the archive logs may not always be delivered to the secondary host reliably, it is recommended that you specify yes.
SNAPSHOT_MODE	Specifies the database snapshot mode. Values can be online, offline, or instant. ■ If the snapshot is created while the database is online, the `dbed_vmsnap` command will put the tablespaces into backup mode. After `dbed_vmsnap` finishes creating the snapshot, it will take the tablespaces out of backup mode, switch the log files to ensure that the extra redo logs are archived, and create a snapshot of the archived logs. ■ If the database is offline, it is not necessary to put the tablespaces into backup mode. The database must be shut down before creating an offline snapshot. ■ If the database snapshot is instant, `dbed_vmsnap` will skip putting the tablespace into backup mode. **Note:** If SNAPSHOT_MODE is set to offline or instant, an off-host configuration is required and the `-r` *`relocate_path`* option is not allowed.

Table 32-2 Parameter values for dbed_vmchecksnap *(continued)*

Parameter	Value
SNAPSHOT_PLAN_FOR	The default value is **database** and cannot be changed. Specifies the database object for which you want to create a snapshot.
SNAPSHOT_PLEX_TAG	Specifies the snapshot plex tag. Use this variable to specify a tag for the plexes to be snapshot. The maximum length of the `plex_tag` is 15 characters. The default plex tag is `dbed_flashsnap`.
SNAPSHOT_VOL_PREFIX	Specifies the snapshot volume prefix. Use this variable to specify a prefix for the snapshot volumes split from the primary disk group. A volume name cannot be more than 32 characters. You should consider the length of the volume name when assigning the prefix.
SNAPSHOT_MIRROR	Specifies the number of plexes to be snapshot. The default value is 1.
DG:*VOL*	Optional These entries are created by default if the `dbed_vmchecksnap` command is run with the `-o setdefaults -m` option. These entries specify the mount path for the associated snapshot volumes when we perform a clone operation using `dbed_vmclondb`. The values for these fields can be left blank, if they are blank then the `-r relocate_path` needs to be specified when performing a `dbed_vmclonedb` operation.

When you first run `dbed_vmchecksnap`, use the `-o setdefaults` option to create a snapplan using default values for variables. You may then edit the file manually to set the variables for different snapshot scenarios.

Before creating a snapplan, make sure the following conditions have been met:

Prerequisites

- Storage must be configured as specified:
  See "Preparing hosts and storage for Database FlashSnap" on page 187.
- You must be the Oracle database administrator.
- The disk group must be version 110 or later. For more information on disk group versions, see the `vxdg`(1M) manual page.
- Be sure that a DCO and DCO volume are associated with the volume for which you are creating the snapshot.
- Snapshot plexes and their associated DCO logs should be on different disks than the original plexes, and should be configured correctly for creating snapshots by the system administrator.
- Persistent FastResync must be enabled on the existing database volumes and disks must be assigned for the snapshot volumes.
- The database must be running in archive log mode. Archive log mode is set in the Oracle initialization parameter file.
- The Oracle database must have at least one mandatory archive destination.
- ORACLE_HOME cannot reside on a disk which will be used for snapshot.
- The Oracle database files and archive log files should use different volumes with unique disks in same disk group.

Usage Notes

- If you are performing offhost processing, the snapplan should be created on the primary host.
- After creating the snapplan using the `dbed_vmchecksnap` command, you can use a text editor to review and update the file, if necessary.
- It is recommended that you create a local working directory to store your snapplans in.
- See the `dbed_vmchecksnap` (1M) online manual page for more information.
- If the SNAPSHOT_MODE for the database is set to online, the primary and secondary hosts can be the same.
- If the SNAPSHOT_MODE is set to offline or instant, the primary and secondary hosts must be different.

To create a snapplan

1. Perform the following steps as an Oracle database administrator.
2. Change directories to the working directory you want to store your snapplan in.

```
$ cd /working_directory
```

3 Create a snapplan with default values using the `dbed_vmchecksnap` command:

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S  ORACLE_SID \
-H ORACLE_HOME -f SNAPPLAN -o setdefaults -t  host_name \
[-p PLEX_TAG] [-m]
```

Example output for using `multi-dg` and for when when you specify `-m` option for mapped mount point.

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S slave -H $ORACLE_HOME \
-f snapplan3 -o setdefaults -t mensa -m
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap will
be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an alternname
use /opt/VRTS/bin/vxsfadm
Snapplan snapplan3 for slave
SNAPSHOT_VERSION=5.0
PRIMARY_HOST=mensa
SECONDARY_HOST=mensa
PRIMARY_DG=datadg1:datadg2:archdg1
ORACLE_SID=slave
ARCHIVELOG_DEST=/arch/sfae10g
SNAPSHOT_ARCHIVE_LOG=yes
SNAPSHOT_MODE=online
SNAPSHOT_PLEX_TAG=dbed_flashsnap
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
archdg1:arch1vol1=
datadg1:data1vol2=
datadg1:data1vol1=
datadg2:data1vol1=
datadg2:data1vol2=
```

4 Open the snapplan file in a text editor and modify it as needed.

Example snapplans created for a snapshot image

In this example, a snapplan, snap1, is created for a snapshot image in a same-node configuration and default values are set. The host is named host1 and the working directory is /export/snap_dir.

The following is an example of the `dbed_vmchecksnap` command and sample output:

```
$ cd /export/snap_dir
```

```
$ $ /opt/VRTS/bin/dbed_vmchecksnap -S PROD \
-H /oracle/product/orahome -f snap1 -o setdefaults -t host1
```

Example ouput for a single disk group environment:

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap will be deprecated
in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername use
/opt/VRTS/bin/vxsfadm
Snapplan snap1 for PROD
SNAPSHOT_VERSION=5.0
PRIMARY_HOST=host1
SECONDARY_HOST=host1
PRIMARY_DG=proddg
ORACLE_SID=PROD
ARCHIVELOG_DEST=/arch/sfae10g
SNAPSHOT_ARCHIVE_LOG=yes
SNAPSHOT_MODE=online
SNAPSHOT_PLEX_TAG=dbed_flashsnap
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
```

Example output for an environment with multiple disk groups:

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Snapplan snap1 for PROD
SNAPSHOT_VERSION=5.0
PRIMARY_HOST=host1
SECONDARY_HOST=host2
PRIMARY_DG=proddg1:proddg2
ORACLE_SID=PROD
ARCHIVELOG_DEST=/arch
SNAPSHOT_ARCHIVE_LOG=yes
SNAPSHOT_MODE=online
SNAPSHOT_PLEX_TAG=my_tag
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
```

In this second example, a snapplan, snap2, is created for a snapshot image in a two-node in the cluster configuration, and default values are set. The primary host is host1, the secondary host is host2, and the working directory is /export/snap_dir.

The following is an example of the `dbed_vmchecksnap` command and sample output:

```
$cd /export/snap_dir

$/opt/VRTS/bin/dbed_vmchecksnap -S PROD \
-H /oracle/product/orahome -f snap2 -o setdefaults -t host2
```

Example ouput for a single disk group environment:

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Snapplan snap2 for PROD
SNAPSHOT_VERSION=5.0
PRIMARY_HOST=host1
SECONDARY_HOST=host2
PRIMARY_DG=proddg
ORACLE_SID=PROD
ARCHIVELOG_DEST=/arch/sfae10g
SNAPSHOT_ARCHIVE_LOG=yes
SNAPSHOT_MODE=online
SNAPSHOT_PLEX_TAG=dbed_flashsnap
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
```

Example output for an environment with multiple disk groups:

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Snapplan snap2 for PROD
SNAPSHOT_VERSION=5.0
PRIMARY_HOST=host1
SECONDARY_HOST=host2
PRIMARY_DG=proddg1:proddg2
ORACLE_SID=PROD
ARCHIVELOG_DEST=/arch/sfae10g
SNAPSHOT_ARCHIVE_LOG=yes
```

```
SNAPSHOT_MODE=online
SNAPSHOT_PLEX_TAG=dbed_flashsnap
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
```

By default, a snapplan's `SNAPSHOT_PLEX_TAG` value is set as dbed_flashsnap. You can use the `-p` option to assign a different tag name. Make use of the `-p` option when creating the snapplan with the `setdefaults` option.

In the following example, the `-p` option is used with `setdefaults` to assign *my_tag* as the `SNAPSHOT_PLEX_TAG` value.

```
$ dbed_vmchecksnap -S PROD -H $ORACLE_HOME -o setdefaults \
-p my_tag -f snap1 -t host2
```

Example ouput for a single disk group environment:

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Snapplan snap1 for PROD
SNAPSHOT_VERSION=5.0
PRIMARY_HOST=host1
SECONDARY_HOST=host2
PRIMARY_DG=proddg
ORACLE_SID=PROD
ARCHIVELOG_DEST=/arch
SNAPSHOT_ARCHIVE_LOG=yes
SNAPSHOT_MODE=online
SNAPSHOT_PLEX_TAG=my_tag
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
```

Example output for an environment with multiple disk groups:

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Snapplan snap1 for PROD
SNAPSHOT_VERSION=5.0
```

```
PRIMARY_HOST=host1
SECONDARY_HOST=host2
PRIMARY_DG=proddg1:proddg2
ORACLE_SID=PROD
ARCHIVELOG_DEST=/arch
SNAPSHOT_ARCHIVE_LOG=yes
SNAPSHOT_MODE=online
SNAPSHOT_PLEX_TAG=my_tag
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
```

Example for creating a snapplan with mapped mount option enabled

Example for creating a snapplan with mapped mount option enabled.

```
$/opt/VRTS/bin/dbed_vmchecksnap -S PROD \
   -H /oracle/product/orahome -f snap2 -o setdefaults -t host2 -m
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Snapplan snap2 for PROD
SNAPSHOT_VERSION=5.0
PRIMARY_HOST=host1
SECONDARY_HOST=host2
PRIMARY_DG=proddg
ORACLE_SID=PROD
ARCHIVELOG_DEST=/arch
SNAPSHOT_ARCHIVE_LOG=yes
SNAPSHOT_MODE=online
SNAPSHOT_PLEX_TAG=dbed_flashsnap
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
PRODdg:data_vol=/tmp/datadst
PRODdg:arch_vol=/tmp/archdst
```

Example output for an environment when `-m` option is used in `dbed_vmchecksnap` :

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S slave -H $ORACLE_HOME \
-f snapplan3 -o setdefaults -t mensa -m
```

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Snapplan snapplan3 for slave
SNAPSHOT_VERSION=5.0
PRIMARY_HOST=mensa
SECONDARY_HOST=mensa
PRIMARY_DG=datadg1:datadg2:archdg1
ORACLE_SID=PROD
ARCHIVELOG_DEST=/arch
SNAPSHOT_ARCHIVE_LOG=yes
SNAPSHOT_MODE=online
SNAPSHOT_PLEX_TAG=dbed_flashsnap
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
archdg1:arch1vol1=
datadg1:data1vol2=
datadg1:data1vol1=
datadg2:data1vol1=
datadg2:data1vol2=
```

Edit the snapplan and add the mountpoints to the *dg:volume* pairs before you validate the snapplan. The mountpoints entered would be the location where the clone database would be mounted after a `dbed_vmclonedb` operation.

Creating multi-mirror snapshots

To make the Database Snapshots highly available, the snapped snapshot volume should contain more than one mirror. This makes the snapshot volumes available even if one of the mirrors becomes disabled. Snapshot volumes can be mounted and the entire database snapshot is usable even if one of the mirrors becomes disabled. The multi-mirror snapshots are enabled by `SNAPSHOT_MIRROR=<n>` in the snapplan.

Note: There are no changes to the Command Line usage or arguments for the Flashsnap tools.

Before taking the snapshot, make sure all tagged snapshot mirrors are in SNAPDONE state.

For information about snapshot mirrors, refer to the *Storage Foundation Administrator's Guide*.

Validating a snapplan (dbed_vmchecksnap)

After creating a snapplan, the next steps are to validate the snapplan parameters and check whether the snapshot volumes have been configured correctly for creating snapshots. If validation is successful, the snapplan is copied to the repository. The snapplan is validated using the `dbed_vmchecksnap` command with the `-o validate` option.

Consider the following prerequisites and notes before validating a snapplan:

Prerequisites

- The database must be up and running while executing the `dbed_vmchecksnap` command.

Usage Notes

- The `dbed_vmchecksnap` command must be run as the Oracle database administrator.
- On AIX:
  The default behavior is to force validation. Use the -n option if you want to skip validation.
- When using `dbed_vmchecksnap -o validate` to validate the snapplan and storage, you can save the validation output. The system administrator can use this information to adjust the storage setup if the validation fails.
- If a snapplan is updated or modified, you must re-validate it. It is recommended that snapplans are revalidated when changes are made in the database disk group.
- See the `dbed_vmchecksnap`(1M) manual page for more information.

Note: You must issue commands as an Oracle database administrator in the following procedure.

To validate a snapplan

1 Change directories to the working directory your snapplan is stored in:

```
$ cd /working_directory
```

2 Validate the snapplan using the `dbed_vmchecksnap` command:

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S  ORACLE_SID \
-H ORACLE_HOME -f SNAPPLAN -o validate
```

Example to validate snapplan snap1 for a snapshot image

In the following example, a snapplan, snap1, is validated for a snapshot image in a same-node configuration. The primary host is host1 and the working directory is /export/snap_dir. Commands are issued as an Oracle database administrator.

The following is an example of the `dbed_vmchecksnap` command and sample output:

```
$ cd /export/snap_dir
$ /opt/VRTS/bin/dbed_vmchecksnap -S PROD -H /oracle/product/orahome \
-f snap1 -o validate
```

Example output:

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Validating database configuration for third-mirror-break-off snapshot:
ORACLE_SID = PROD
ORACLE_HOME = /oracle/product/orahome
APP_MODE = online
SNAPSHOT_ARCHIVE_LOG = yes
REVERSE_RESYNC_ARCHIVE_LOG = no
REVERSE_RESYNC_RECOVERY = auto
ARCHIVELOG_DEST = /arch
Database validation successful.
Validating database volume layout for third-mirror-break-off snapshot:
Data volumes ready for snapshot:
Volume/volume-set datavol of diskgroup datadg mounted on /data.
Archivelog volume ready for snapshot:
Volume/volume-set archvol of diskgroup datadg mounted on /arch.
Storage units to be used for snapshot from diskgroup datadg:
```

```
hitachi_usp-vm0_01c4 hitachi_usp-vm0_01c6
SNAPSHOT_VOL_PREFIX = SNAP_
SNAPSHOT_DG_PREFIX = SNAP_
Database volume layout validated successfully.
```

Displaying, copying, and removing a snapplan (dbed_vmchecksnap)

Consider the following usage notes before listing all snapplans for a specific Oracle database, displaying a snapplan file, or copying and removing snapplans.

Usage Notes

- If the local snapplan is updated or modified, you must revalidate it.
- If the database schema or disk group is modified, you must revalidate it after running `dbed_update`.

Displaying a snapplan

You can use the `dbed_vmchecksnap` command to list all available snapplans and to display detailed information for a particular snapplan.

To list all available snapplans for a specific Oracle database

◆ Use the `dbed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S ORACLE_SID -o list
```

In the following example, all available snapplans are listed for the database PROD.

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S PROD -o list
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an alternname
use /opt/VRTS/bin/vxsfadm
The following snapplan(s) are available for sfae10g:
SNAP_PLAN  SNAP_STATUS  DB_STATUS  SNAP_READY
snap1  init_full  -   yes
snap2  init_full  -   yes
```

The command output displays all available snapplans, their snapshot status (SNAP_STATUS), database status (DB_STATUS), and whether a snapshot may be taken (SNAP_READY).

For Database FlashSnap status information:

See “About Database FlashSnap status information” on page 477.

To display detailed information for a snapplan

- Use the `dbed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S \
ORACLE_SID -f SNAPPLAN -o list
```

In the following example, the snapplan snap1 is displayed.

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S PROD -f snap1 -o list

SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
SNAPSHOT_VERSION=5.0
ORACLE_SID=sfaedb1
ORACLE_HOME=/ora_base/db_home
SNAPSHOT_PLEX_TAG=dbed_flashsnap
SNAPSHOT_DG_PREFIX=SNAP_
SNAPSHOT_VOL_PREFIX=SNAP_
SNAPSHOT_MIRROR=1
SNAPSHOT_MODE=online
SNAPSHOT_ARCHIVE_LOG=yes

STORAGE_INFO
sdu
sdx
SNAP_PLEX=datavol-02 archvol-02 datavol_dcl-02 archvol_dcl-02

STATUS_INFO
SNAP_STATUS=init_full
```

Copying a snapplan

If you want to create a snapplan similar to an existing snapplan, you can simply create a copy of the existing snapplan and modify it. To copy a snapplan from the SFDB repository to your current directory, the snapplan must not already be present in the current directory.

To copy a snapplan from the SFDB repository to your current directory

- Use the `dbed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S  ORACLE_SID \
-f SNAPPLAN -o copy
```

In the following example, the snapplan, snap1, is copied from the VxDBA repository to the current directory.

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S PROD -f snap1 -o  copy
SFDB vxsfadm WARNING V-81-0000 dbed_vmchecksnap
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername use
/opt/VRTS/bin/vxsfadm
Copying 'snap1' to '/export/snap_dir'
```

Removing a snapplan

A snapplan can be removed from a local directory or repository if the snapplan is no longer needed.

To remove a snapplan from the SFDB repository

- Use the `dbed_vmchecksnap` command as follows:

```
$ /opt/VRTS/bin/dbed_vmchecksnap -S ORACLE_SID -f\
SNAPPLAN -o remove
```

Creating a snapshot (dbed_vmsnap)

The `dbed_vmsnap` command creates a snapshot of an Oracle database by splitting the mirror volumes used by the database into a snapshot database. You can use the snapshot image on either the same host as the database or on a secondary host provided storage is shared by the two hosts.

The snapshot image created by `dbed_vmsnap` is a frozen image of an Oracle database's datafiles. The `dbed_vmsnap` command ensures that a backup control file is created when the snapshot database is created, which allows for complete data recovery, if needed.

For Database FlashSnap status information:

See "About Database FlashSnap status information" on page 477.

Prerequisites

- You must be logged in as the Oracle database administrator.
- You must create and validate a snapplan using `dbed_vmchecksnap` before you can create a snapshot image with `dbed_vmsnap`.

Usage Notes

- If you are performing offhost processing, you should use the `dbed_vmsnap` command on the primary host.
- Do not share volumes between Oracle database files and other software.
- When creating a snapshot volume, create the snapshot on a separate controller and on separate disks from the primary volume.
- Make sure your archive log destination is separate from your Oracle database volumes.
- Do not place any datafiles, including control files, in the $ORACLE_HOME/dbs directory.
- Resynchronization speed varies based on the amount of data changed in both the primary and secondary volumes when the mirror is broken off.
- See the `dbed_vmsnap(1M)` manual page for more information.

Note the following points:

- To force snapshot creation, use the `-F` option. The `-F` option can be used after a snapshot operation has failed and the problem was fixed without using SFDB commands. (That is, the volumes were synchronized using VxVM commands.) In this situation, the status of the snapplan will appear as unavailable for creating a snapshot. The `-F` option ignores the unavailable status, checks for the availability of volumes, and creates the snapshot after the volumes pass the availability check.
- After the snapshot is created, `dbed_vmsnap` returns values you will need to run `dbed_vmclonedb`. These values include the snapshot disk group, the snapplan name, and the SFDB repository volume for an off-host configuration. Make a note of these values so you have them when running `dbed_vmclonedb`.
- You can also use the command `dbed_vmchecksnap -f snapplan -o list` to access the information regarding the snapshot disk group, the snapplan name, and the SFDB repository.

Note: You must issue commands as an Oracle database administrator in the following procedure.

To create a snapshot

1 Change directories to the working directory in which your snapplan is stored:

```
$ cd /working_directory
```

2 If SNAPSHOT_MODE is set to offline in the snapplan, shut down the database.

3 Create the snapshot image using the `dbed_vmsnap` command.

```
$ /opt/VRTS/bin/dbed_vmsnap -S ORACLE_SID -f  SNAPPLAN \
-o snapshot
```

```
$ /opt/VRTS/bin/dbed_vmsnap -S ORACLE_SID -f SNAPPLAN \
-o snapshot
```

```
$ /opt/VRTS/bin/dbed_vmsnap -S ORACLE_SID -f SNAPPLAN \
-o snapshot [-F]
```

The snapshot volumes now represent a consistent backup copy of the database. You can backup the database by copying the snapshot volumes to tape or other backup media.

See "Backing up the database from snapshot volumes (dbed_vmclonedb)" on page 418.

4 You can also create another Oracle database for decision-support purposes.

See "Cloning a database (dbed_vmclonedb)" on page 422.

Example to create a snapshot image of the database PROD

In this example, a snapshot image of the database, PROD, is created for a same-node configuration. In this case, the SECONDARY_HOST parameter is set the same as the PRIMARY_HOST parameter in the snapplan.

Note: You must issue commands as an Oracle database administrator in the following procedure.

```
$ /opt/VRTS/bin/dbed_vmsnap -S PROD -f snap1 -o snapshot
```

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmsnap
```

```
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an alternname
use /opt/VRTS/bin/vxsfadm
Validating database configuration for third-mirror-break-off snapshot:
ORACLE_SID = PROD
ORACLE_HOME = /ora_base/db_home
APP_MODE = online
ARCHIVELOG_DEST = /arch
SNAPSHOT_ARCHIVE_LOG = yes
REVERSE_RESYNC_ARCHIVE_LOG = no
REVERSE_RESYNC_RECOVERY = auto
Database validation successful.
snapshot started at Mon Aug 19 05:14:12 2013.
Putting database in backup mode ... Done
Validating database volume layout for third-mirror-break-off snapshot:
Data volumes ready for snapshot:
Volume/volume-set datavol of diskgroup datadg mounted on /data.
Archivelog volume ready for snapshot:
Volume/volume-set archvol of diskgroup datadg mounted on /arch.
Storage units to be used for snapshot from diskgroup datadg:
hitachi_usp-vm0_01c4 hitachi_usp-vm0_01c6
SNAPSHOT_VOL_PREFIX = SNAP_
SNAPSHOT_DG_PREFIX = SNAP_
Database volume layout validated successfully.
Creating snapshot volumes for data volumes ... Done
Taking database out of backup mode ... Done
Flushing online redo logs ... Done
Creating snapshot volume for archivelog volume ... Done
Copying snapshot information to snapshot volume ... Done
Creating snapshot diskgroups ... Done
Deporting snapshot diskgroups ... Done
SNAP_datadg
snapshot ended at Mon Aug 19 05:14:53 2013.
```

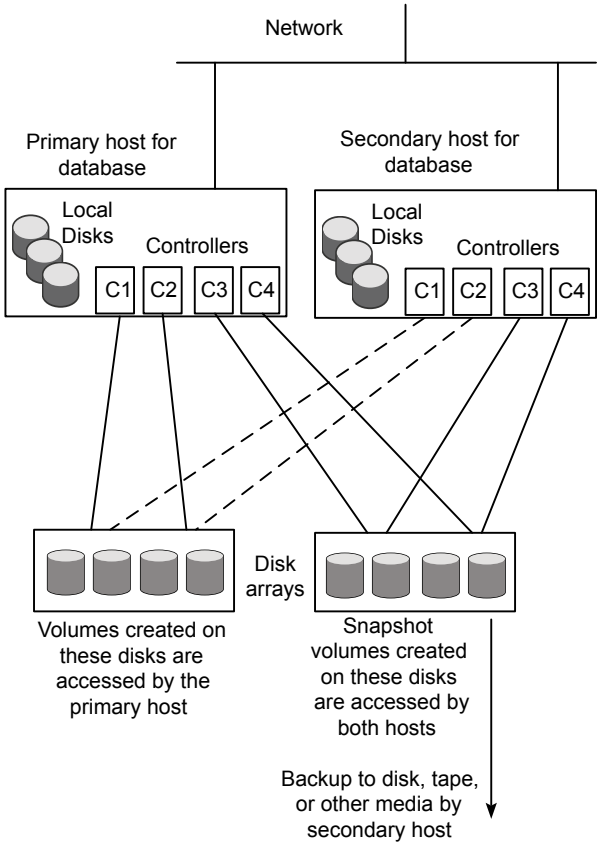
Backing up the database from snapshot volumes (dbed_vmclonedb)

Snapshots are most commonly used as a source for backing up a database. The advantage of using snapshot volumes is that the backup will not contest the I/O bandwidth of the physical devices. Making the snapshot volumes available on a secondary host will eliminate the extra loads put on processors and I/O adapters by the backup process on the primary host.

A clone database can also serve as a valid backup of the primary database. You can back up the primary database to tape using snapshot volumes.

Figure 32-7 shows a typical configuration when snapshot volumes are used on a secondary host.

Figure 32-7 Example system configuration for database backup on a secondary host



Prerequisites

- You must be logged in as the Oracle database administrator to use `dbed_vmclonedb` command.
- Before you can use the `dbed_vmclonedb` command, you must validate a snapplan and create a snapshot.
  See "Backing up the database from snapshot volumes (dbed_vmclonedb)" on page 418.
  See "Validating a snapplan (dbed_vmchecksnap)" on page 411.
  See "Creating a snapshot (dbed_vmsnap)" on page 415.
- The volume snapshot must contain the entire database.
- Before you can use the `dbed_vmclonedb` command with the `-r` *`relocate_path`* option (which specifies the initial mount point for the snapshot image), the system administrator must create the mount point and then change the owner to the Oracle database administrator.

Usage Notes

- The `dbed_vmclonedb` command can be used on the secondary host.
- In a same-node configuration, the primary and secondary hosts are the same.
- In a same-node configuration, `-r` *`relocate_path`* is required if no mapped mounts were specified or some of the mapped mount entries were left blank in the snapplan.
- See the `dbed_vmclonedb`(1M) manual page for more information.

Mounting the snapshot volumes and backing up

Before using the snapshot volumes to do a backup, you must first mount them.

Note: You must issue commands as an Oracle database administrator in the following procedure.

Note: If you use the Oracle online backup method, you must also back up all the archived log files in order to do a complete restore and recovery of the database.

To mount the snapshot volumes

- Use the `dbed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/dbed_vmclonedb -S ORACLE_SID \
  -o mount,new_sid=new_sid,server_name=svr_name\
  -f SNAPPLAN [-H ORACLE_HOME] \
  [-r relocate_path]
```

You can now back up an individual file or a group of files under a directory onto the backup media.

In this example, snapshot volumes are mounted.

```
$ /opt/VRTS/bin/dbed_vmclonedb -S PROD \
 -o mount,new_sid=NEWPROD,server_name=svr_name \
 -f snap1 -r /clone/single
```

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmclonedb
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Retrieving snapshot information ... Done
Importing snapshot diskgroups ... Done
Mounting snapshot volumes ... Done
```

Note: A usage error is displayed if the server_name is not given in the above command for offhost configurations.

To mount a Storage Checkpoint carried over from the snapshot volumes to a secondary host

1. On the secondary host, list the Storage Checkpoints carried over from the primary database using the `dbed_ckptdisplay` command.

   For example:

```
$ /opt/VRTS/bin/dbed_ckptdisplay -S ORACLE_SID -n
```

2. You can mount one of the listed Storage Checkpoints using the `dbed_ckptmount` command.

For example:

```
$ /opt/VRTS/bin/dbed_ckptmount -S ORACLE_SID -c CKPT_NAME \
  -m MOUNT_POINT
```

Note the following limitations:

- Any mounted Storage Checkpoints must be unmounted before running the following commands:

  ```
  $ /opt/VRTS/bin/dbed_ckptumount -S ORACLE_SID -c CKPT_NAME
  ```

- It is only possible to mount a Storage Checkpoint carried over with the snapshot volumes in an off-host configuration if the snapshot volumes were mounted with the `dbed_vmclonedb` command with the `-o mount` option without the use of `-r relocate_path`.
- Storage Checkpoints carried over with the snapshot volumes can be mounted before a clone database is created using `dbed_vmclonedb` with the `-o mount` option. After a clone database is created using `dbed_vmclonedb` with the `-o recoverdb` option, however, Storage Checkpoints are no longer present.

To back up the database using the snapshot

◆ Copy the snapshot volumes to tape or other appropriate backup media.

Restoring from backup

For single instance Oracle environments, backup copies are used to restore volumes lost due to disk failure, or data destroyed due to human error. If a volume's data is corrupted and you know that you need to restore it from backup, you can use Database FlashSnap's reverse resynchronization function to restore the database.

Cloning a database (dbed_vmclonedb)

The SFDB commands enable you to create a clone database using snapshot volumes. You can use snapshots of a primary database to create a clone of the database at a given point in time. You can then implement decision-support analysis and report generation operations that take their data from the database clone rather than from the primary database to avoid introducing additional burdens on the production database.

A clone database can also serve as a valid backup of the primary database.

See "Backing up the database from snapshot volumes (dbed_vmclonedb)" on page 418.

You can also back up the primary database to tape using snapshot volumes.

The resynchronization functionality of Database FlashSnap allows you to quickly refresh the clone database with up-to-date information from the primary database. Reducing the time taken to update decision-support data also lets you generate analysis reports more frequently.

Using Database FlashSnap to clone a database

In a same-node configuration, the `dbed_vmclonedb` command creates a clone database on the same host. The command can also be used to shut down the clone database and unmount its file systems. When creating or unmounting the clone database in a same-node configuration, `-r` *`relocate_path`* is required so that the clone database's file systems use different mount points than those used by the primary database.

When used in an off-host configuration, the `dbed_vmclonedb` command imports the snapshot disk groups, mounts the snapshot, and starts a clone database. It can also reverse the process by shutting down the clone database, unmounting the file systems, and deporting the snapshot disk group.

Warning: When creating a clone database, all Storage Checkpoints in the original database are discarded.

Prerequisites

- You must be logged in as the Oracle database administrator.
- Before you can use the dbed_vmclonedb command, you must validate a snapplan and create a snapshot.
  See "About creating database snapshots" on page 391.
  See "Validating a snapplan (dbed_vmchecksnap)" on page 411.
  See "Creating a snapshot (dbed_vmsnap)" on page 415.
- The volume snapshot must contain the entire database.
- The system administrator must provide the database administrator with access to the necessary volumes and mount points.
- Before you can use the `dbed_vmclonedb` command with the `-r` *`relocate_path`* option (which specifies the initial mount point for the snapshot image), the system administrator must create the mount point and then change the owner to the Oracle database administrator.
- The Oracle database must have at least one mandatory archive destination.
- In case of mapped mounts, the mapped mount points need to be created by the System Administrator.

Usage Notes

- The `dbed_vmclonedb` command can be used on the secondary host.
- In a same-node configuration, -r relocate_path is required if no mapped mounts were specified or some of the mapped mount entries were left blank in the snapplan.
- The initialization parameters for the clone database are copied from the primary database. This means that the clone database takes up the same memory and machine resources as the primary database. If you want to reduce the memory requirements for the clone database, shut down the clone database and then start it up again using a different init.ora file that has reduced memory requirements. If the host where `dbed_vmclonedb` is run has little available memory, you may not be able to start up the clone database and the cloning operation may fail.
- See the `dbed_vmclonedb`(1M) manual page for more information.

To mount a database and recover it manually

1. Start and mount the clone database using SFDB commands to allow manual database recovery:
2. Follow the Oracle recovery procedure to recover the database manually.
3. Update the snapshot status information for the clone database in the SFDB repository using SFDB commands:

Example: Mounting the file systems without bringing up the clone database

In this example, file systems are mounted without bringing up the clone database. The clone database must be manually created and recovered before it can be used. This example is for a clone created on the same host as the primary database.

```
$ /opt/VRTS/bin/dbed_vmclonedb -S FLAS11r2 \
 -o mountdb,new_sid=clone1,server_name=motmot -f snap1 \
 -r /cudb
```

```
dbed_vmclonedb -S PROD -H /ora_base/db_home -o mountdb,
new_sid=clons1, server_name=host1 -f snap1
SFDB vxsfadm WARNING V-81-0000 dbed_vmclonedb
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Retrieving snapshot information ...                              Done
Importing snapshot diskgroups ...                                Done
Mounting snapshot volumes ...                                    Done
Preparing parameter file for clone database ...                  Done
Mounting clone database clons1 ...                               Done
```

The database status (database_recovered) needs to be updated for a clone database on the primary host after manual recovery has been completed.

```
$ /opt/VRTS/bin/dbed_vmclonedb -o update_status,\
  new_sid=NEWPROD,server_name=svr_name -f snap1 -r /clone
```

To clone the database automatically

- Use the `dbed_vmclonedb` command as follows:

  ```
  $ /opt/VRTS/bin/dbed_vmclonedb -S ORACLE_SID \
    -o recoverdb,new_sid=new_sid,server_name=svr_name -f SNAPPLAN \
    [-H ORACLE_HOME] [-r relocate_path]
  ```

  Where:

ORACLE_SID	Represents the name of the Oracle database used to create the snapshot.
new_sid	Specifies the ORACLE_SID for the clone database.
server_name	Specifies the server name as svr_name.
SNAPPLAN	Represents the name of the snapplan file.

ORACLE_HOME	Represents the ORACLE_HOME setting for the ORACLE_SID database.
relocate_path	Represents the name of the initial mount point for the snapshot image.

When cloning a database on a secondary host, ensure that PRIMARY_HOST and SECONDARY_HOST parameters in the snapplan file are different.

When the `-o recoverdb` option is used with `dbed_vmclonedb`, the clone database is recovered automatically using all available archive logs. If the `-o recoverdb` option is not used, you can perform point-in-time recovery manually.

In the following example, a clone of the primary database is automatically created.

```
$ /opt/VRTS/bin/dbed_vmclonedb -S FLAS11r2 \
  -o recoverdb,new_sid=clone1,server_name=motmot.veritas.com\
 -f snap1 -r /cudb

SFDB vxsfadm WARNING V-81-0000 dbed_vmclonedb
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Retrieving snapshot information ... Done
Importing snapshot diskgroups ... Done
Mounting snapshot volumes ... Done
Preparing parameter file for clone database ... Done
Mounting clone database clone1 ... Done
Starting database recovery ...
Clone database clone1 is open.
```

Shutting down the clone database and unmounting file systems

When you are done using the clone database, you can shut it down and unmount all snapshot file systems with the `dbed_vmclonedb -o umount` command. If the clone database is used on a secondary host that has shared disks with the primary host, the `-o umount` option also deports the snapshot disk group.

Note: Any mounted Storage Checkpoints mounted need to be unmounted before running `dbed_vmclonedb -o umount`.

To shut down the clone database and unmount all snapshot file systems

- Use the `dbed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/dbed_vmclonedb \
-o umount,new_sid=NEWPROD,server_name=svr_name -f snap1 -r /clone
SFDB vxsfadm WARNING V-81-0000 dbed_vmclonedb
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Shutting down clone database clone1 ...
Database shut down.
Retrieving snapshot information ... Done
Unmounting snapshot volumes ... Done
Deporting snapshot diskgroups ... Done
```

Restarting a Clone Database

If the clone database is down as a result of using `dbed_vmclonedb -o umount` or rebooting the system, you can restart it with the `-o restartdb` option.

Note: This option can only be used when a clone database is created successfully. If the clone database is recovered manually, `-o update_status` must be run to update the status before `-o restartdb` will work.

To start the clone database

- Use the `dbed_vmclonedb` command as follows:

```
$ /opt/VRTS/bin/dbed_vmclonedb -S ORACLE_SID \
  -o restartdb,new_sid=new_sid,server_name=svr_name \
  -f SNAPPLAN [-H ORACLE_HOME] \
  [-r relocate_path]
```

In this example, the clone database is re-started on the same host as the primary database (same-node configuration).

```
$ /opt/VRTS/bin/dbed_vmclonedb -S FLAS11r2 \
  -o restartdb,new_sid=clone1,server_name=motmot \
  -f snap1 -r /cudb
```

```
SFDB vxsfadm WARNING V-81-0000 dbed_vmclonedb
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername
use /opt/VRTS/bin/vxsfadm
Retrieving snapshot information ... Done
Importing snapshot diskgroups ... Done
Mounting snapshot volumes ... Done
Starting clone database clone1 ...
Clone database started.
```

Recreating Oracle tempfiles

After a clone database is created and opened, the tempfiles are added if they were residing on the snapshot volumes. If the tempfiles were not residing on the same file systems as the datafiles, `dbed_vmsnap` does not include the underlying volumes in the snapshot. In this situation, `dbed_vmclonedb` issues a warning message and you can then recreate any needed tempfiles on the clone database as described in the following procedure.

To recreate the Oracle tempfiles

1. If the tempfiles were not residing on the same file systems as the datafiles, the `dbed_vmclonedb` command displays WARNING and INFO messages similar to the following:

   ```
   WARNING: Not all tempfiles were included in snapshot for
   $ORACLE_SID, there is no snapshot volume for
   /clone_path/temp02.dbf.
   WARNING: Could not recreate tempfiles for $ORACLE_SID due to
   lack of free space.INFO: The sql script for adding tempfiles to
   $ORACLE_SID is at /tmp/add_tf.$ORACLE_SID.sql.
   ```

   where $ORACLE_SID is the name of the clone database.

2. A script named `add_tf.`*`$ORACLE_SID`*`.sql` is provided in the /tmp directory for the purpose of recreating Oracle tempfiles. This script contains the SQL*Plus commands to recreate the missing tempfiles.

3. Make a copy of the /tmp/add_tf.*$ORACLE_SID*.sql script and open it to view the list of missing tempfiles.

   An example of the add_tf.*$ORACLE_SID*.sql script is shown below:

   ```
   $ cat add_tf.mnt2.sql
   -- Commands to add tempfiles to temporary tablespaces.
   -- Online tempfiles have complete space information.
   -- Other tempfiles may require adjustment.
   ALTER TABLESPACE TEMP ADD TEMPFILE
    'DBED: ADJUST_BEFORE_RUN/snap_data11r2/mnt2/temp01.dbf'
   SIZE 20971520  REUSE AUTOEXTEND ON NEXT 655360  MAXSIZE 32767M;
   ```

4. Evaluate whether you need to recreate any temp files. If you want to recreate tempfiles, proceed to the next step.

5. In the add_tf.*$ORACLE_SID*.sql file, edit the sizes and default path names of the tempfiles as needed to reside on cloned volumes configured for database storage.

   Warning: Do not run the script without first editing it because path names may not exist and the specified mount points may not contain sufficient space.

6. After you have modified the `add_tf.$ORACLE_SID.sql` script, execute it against your clone database.

7. After you have successfully run the script, you may delete it.

Resynchronizing the snapshot to your database

When you have finished using a clone database or want to refresh it, you can resynchronize it with the original database. This is also known as refreshing the snapshot volume or merging the split snapshot image back to the current database image. After resynchronizing, the snapshot can be retaken for backup or decision-support purposes.

When resynchronizing the data in a volume:

- Resynchronize the snapshot from the original volume. This procedure is explained in this section.
- Resynchronizing the original volume from the snapshot. This choice is known as reverse resynchronization. Reverse resynchronization may be necessary to restore a corrupted database and is usually much quicker than using alternative approaches such as full restoration from backup media.
  See "About Reverse Resynchronization in volume-level snapshots (FlashSnap)" on page 164.
  For single instance oracle, the clone database can be used to reverse resynchronize the original volume from the data in the snapshot, or can be discarded by rejoining the snapshot volumes with the original volumes (that is, by resynchronizing the snapshot volumes) for future use.

Note: The reverse resynchronization option is not available for Oracle RAC environments.

You can resynchronize the snapshot from the original volume.

Prerequisites

- You must be logged in as the Oracle database administrator.
- Before you can resynchronize the snapshot image, you must validate a snapplan and create a snapshot.
  See "About creating database snapshots" on page 391.
  See "Validating a snapplan (dbed_vmchecksnap)" on page 411.
  See "Creating a snapshot (dbed_vmsnap)" on page 415.
- If a clone database has been created, shut it down and unmount the file systems using the `dbed_vmclonedb -o umount` command. This command also deports the disk group if the primary and secondary hosts are different.
  See "Shutting down the clone database and unmounting file systems" on page 426.
- The Oracle database must have at least one mandatory archive destination.

Usage Notes

- The `dbed_vmsnap` command can be executed on either host.
- In an off-host configuration, the `dbed_vmsnap` command imports the disk group that was deported from the secondary host and joins the disk group back to the original disk group. The snapshot volumes again become plexes of the original volumes. The snapshot is then resynchronized.
- See the `dbed_vmsnap`(1M) manual page for more information.

Note: You must issue commands as an Oracle database administrator in the following procedure.

To resynchronize the snapshot image

◆ Use the `dbed_vmsnap` command as follows:

```
$ /opt/VRTS/bin/dbed_vmsnap -S ORACLE_SID -f SNAPPLAN -o resync
```

In this example, the snapshot image is resynchronized with the primary database.

```
$ dbed_vmsnap -S FLAS11r2 -f snap -o resync

SFDB vxsfadm WARNING V-81-0000 dbed_vmsnap will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0000 As an altername use /opt/VRTS/bin/vxsfadm
resync started at Mon Aug 19 07:30:42 2013.
Importing snapshot diskgroups ... Done
Joining snapshot diskgroups to original diskgroups ... Done
Reattaching snapshot volumes to original volumes ... Done
resync ended at Mon Aug 19 07:31:13 2013.
```

Now, you can again start creating snapshots.

Removing a snapshot volume

If a snapshot volume is no longer needed, you can remove it and free up the disk space for other uses by using the `vxedit rm` command.

Prerequisites

- You must be logged in as superuser.
- If the volume is on a mounted file system, you must unmount it before removing the volume.

To remove a snapplan and snapshot volume

1 To remove the snapshot and free up the storage used by it:

- If the snapshot has been taken, remove the snapshot as follows:

```
# vxsnap -g diskgroup dis snapshot_volume
# vxvol -g diskgroup stop snapshot_volume
# vxedit -g diskgroup -rf rm snapshot_volume
```

- If the snapshot has not been taken and the snapshot plex (mirror) exists, remove the snapshot as follows:

```
# vxsnap -g diskgroup rmmir volume
```

2 Remove the DCO and DCO volume:

```
# vxsnap -g diskgroup unprepare volume
```

3 Remove the snapplan.

```
# /opt/VRTS/bin/dbed_vmchecksnap -S PROD -f snapplan -o remove
```

For example, the following commands will remove a snapshot volume from disk group PRODdg:

```
# vxsnap -g PRODdg dis snap_v1
# vxvol -g PRODdg stop snap_v1
# vxedit -g PRODdg -rf rm snap_v1
```

Guidelines for Oracle recovery

For an optimal Oracle recovery, the following steps should be taken:

- Back up all control files before Storage Rollback
- Ensure that the control files are not rolled back
- Ensure that all archived redo logs are available
- Media recovery procedures

Back up all control files before Storage Rollback

This guideline is recommended in case the subsequent Oracle recovery is not successful.

Oracle recommends that you keep at least two copies of the control files for each Oracle database and that you store the copies on different disks. Control files should also be backed up before and after making structural changes to databases.

Note: The `dbed_ckptcreate` command automatically saves control file and log information when you create a Storage Checkpoint.

Ensure that the control files are not rolled back

A control file is a small binary file that describes the structure of the database and must be available to mount, open, and maintain the database. The control file stores all necessary database file information, log file information, the name of the database, the timestamp of database creation, and synchronization information, such as the Storage Checkpoint and log-sequence information needed for recovery.

Rolling back the control file will result in an inconsistency between the physical database structure and the control file.

Ensure that all archived redo logs are available

A database backup with online and archived logs is required for a complete database recovery.

Query V$ARCHIVED_LOG to list all the archived log information and V$ARCHIVE_DEST to list the location of archive destinations.

Note: Refer to your Oracle documentation for information about querying archived information.

For Storage Foundation for Oracle RAC, the archive log destination must be on a Veritas cluster file system.

To restore the necessary archived redo log files, you can query V$LOG_HISTORY to list all the archived redo log history or query V$RECOVERY_LOG to list only the archived redo logs needed for recovery. The required archived redo log files can be restored to the destination specified in the LOG_ARCHIVE_DEST parameter or to an alternate location. If the archived redo logs were restored to an alternate location, use the `ALTER DATABASE RECOVER ... FROM` statement during media recovery.

After Storage Rollback, perform Oracle recovery, applying some or all of the archived redo logs.

Note: After rolling back the database (including control files and redo logs) to a Storage Checkpoint, you need to recover the Oracle database instance. Rolling the database forward is not supported; that is, you cannot apply archived redo logs.

Media recovery procedures

The following are the procedures for performing either a complete or incomplete media recovery.

Media recovery procedures

- To perform a complete media recovery:

```
SQL> SET AUTORECOVERY ON;

SQL> RECOVER DATABASE;
```

- To perform an incomplete media recovery, use one of the following:

```
SQL> RECOVER DATABASE UNTIL CANCEL;
```

or

```
SQL> RECOVER DATABASE UNTIL TIME 'yyyy-mm-dd:hh:mm:ss';
```

(You can confirm the time of error by checking the ../bdump/alert*.log file.)
or

```
SQL> RECOVER DATABASE UNTIL TIME 'yyyy-mm-dd:hh:mm:ss'\
using backup controlfile;
```

or

```
SQL> RECOVER DATABASE UNTIL CHANGE scn;
```

- To open the database after an incomplete media recovery, use the following:

```
SQL> ALTER DATABASE OPEN RESETLOGS;
```

RESETLOGS resets the log sequence. The RESETLOGS option is required after an incomplete media recovery. After opening the database with the RESETLOGS option, remove the Storage Checkpoint you just rolled back to as

well as any Storage Checkpoints that were taken before that one. These earlier Storage Checkpoints can no longer be used for storage rollback. After removing these Storage Checkpoints, be sure to create a new Storage Checkpoint.

Warning: Attempting to roll back to the same Storage Checkpoint more than once can result in data corruption. After rolling back, be sure to delete the Storage Checkpoint that you rolled back to and then create a new one.

See your Oracle documentation for complete information on recovery.

Database Storage Checkpoint Commands

The Storage Foundation for Databases (SFDB) tools provide a command line interface to many key operations. The command line interface enables you to incorporate command operations into scripts and other administrative processes.

SFDB tools commands supported are located in the /opt/VRTS/bin directory.

The online manual pages for these commands are located in the /opt/VRTS/man directory.

Note: For a Storage Foundation product database, when you issue the commands, replace `$ORACLE_SID` with `$ORACLE_SID=instance_name` and provide the instance name on which the instance is running.

Table 32-3 summarizes the commands available to you from the command line.

Table 32-3 Database Checkpoint commands

Command	Description
`dbed_ckptcreate`	Command that creates a Storage Checkpoint for an Oracle database. See "Creating Storage Checkpoints using dbed_ckptcreate" on page 437.
`dbed_ckptdisplay`	Command that displays the Storage Checkpoints associated with an Oracle instance. See "Displaying Storage Checkpoints using dbed_ckptdisplay" on page 439.

Table 32-3 Database Checkpoint commands *(continued)*

Command	Description
`dbed_ckptmount`	Command that mounts a Storage Checkpoint for an Oracle instance. See "Mounting Storage Checkpoints using dbed_ckptmount" on page 441.
`dbed_ckptumount`	Command that unmounts a Storage Checkpoint for an Oracle instance. See "Unmounting Storage Checkpoints using dbed_ckptumount" on page 442.
`dbed_ckptrollback`	Command that rolls back an Oracle instance to a Storage Checkpoint point-in-time image. See "Performing Storage Rollback using dbed_ckptrollback" on page 443.
`dbed_ckptremove`	Command that removes a Storage Checkpoint for an Oracle instance. See "Removing Storage Checkpoints using dbed_ckptremove" on page 444.
`dbed_clonedb`	Command that creates a copy of an Oracle database by cloning all existing database files and recreating the control file accordingly. This cloned database can only be started on the same host as the existing database as long as it uses a different SID. See "Cloning the Oracle instance using dbed_clonedb" on page 445.

This section displays examples of SFDB commands that are used to perform administrative operations for Storage Checkpoints and Storage Rollbacks. Review the prerequisites and usage notes listed for each command before using that command.

Note: For detailed information about these commands, their command syntax, and available options, see the individual manual pages.

Creating or updating the repository using dbed_update

You can use the `dbed_update` command to create or update the repository.

Note: Any time you change the structure of the database (for example, by adding or deleting data files, converting PFILE to SPFILE, or converting SPFILE to PFILE), you must run the `dbed_update` command.

Before creating or updating the repository, the following conditions must be met:

Prerequisites

- You must be logged on as the database administrator (typically, the user ID oracle).

Usage notes

- If the repository already exists, the `dbed_update` command will refresh the information.
- The database must be up and running, and the *ORACLE_SID* and the *ORACLE_HOME* variable arguments must be specified with the `-S` and `-H` options, respectively.
- See the `dbed_update`(1M) manual page for more information.

To update the repository

◆ Use the dbed_update command as follows:

```
$ /opt/VRTS/bin/dbed_update -S PROD -H /oracle/product/ORA_HOME
```

Note: If the user switches from SPFILE to PFILE, then the `dbed_update` command must be issued using the -P option. The <ORACLE_PFILE > option specifies the fully qualified path of the Oracle PFILE, if it is not located under ORACLE_HOME/dbs directory.

Creating Storage Checkpoints using dbed_ckptcreate

You can use the `dbed_ckptcreate` command to create a Storage Checkpoint for an Oracle database from the command line.

Storage Checkpoints can be online, offline, or instant. By default, Storage Checkpoints are online. For online Storage Checkpoints, the database is put into hot-backup mode when the Storage Checkpoint is created. If offline is specified, the database is expected to be down.

If instant is specified, the database must be online and a Storage Checkpoint will be taken for a "crash recovery"-type recovery.

Before creating a Storage Checkpoint, the following conditions must be met:

Prerequisites

- You must be logged on as the database administrator (typically, the user ID oracle).
- For best recoverability, always keep ARCHIVELOG mode enabled when you create Storage Checkpoints.

Usage notes

- dbed_ckptcreate stores Storage Checkpoint information in the SFDB repository.
- See the `dbed_ckptcreate`(1M) manual page for more information.

To create Storage Checkpoints while the database is online

◆ Use the `dbed_ckptcreate` command as follows:

```
$/opt/VRTS/bin/dbed_ckptcreate -S PROD \
-H /oracle/product/ORA_HOME -o online

SFDB vxsfadm WARNING V-81-0962 dbed_ckptcreate
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an altername
use /opt/VRTS/bin/vxsfadm
Putting database in backup mode...                          Done
Creating Storage Checkpoint Ckpt_1378514849 ...             Done
Storage Checkpoint Ckpt_1378514849 created
Removing the database from backup mode...                   Done
```

To create Storage Checkpoints while the database is offline

◆ Use the `dbed_ckptcreate` command as follows:

```
$/opt/VRTS/bin/dbed_ckptcreate -S PROD \
-H /oracle/product/ORA_HOME -o offline

SFDB vxsfadm WARNING V-81-0962 dbed_ckptcreate will be
 deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative use
 /opt/VRTS/bin/vxsfadm
Creating Storage Checkpoint Ckpt_1378465225 ...             Done
Storage Checkpoint Ckpt_1378465225 created
```

To create an instant Storage Checkpoints

- Ensure that the database is online and use the `dbed_ckptcreate` command as follows:

```
$ /opt/VRTS/bin/dbed_ckptcreate -S PROD \
-H /oracle/product/oracle -o instant

SFDB vxsfadm WARNING V-81-0962 dbed_ckptcreate
will be deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative use
 /opt/VRTS/bin/vxsfadm
Creating Storage Checkpoint Ckpt_1378465458 ...            Done
Storage Checkpoint Ckpt_1378465458 created
```

Displaying Storage Checkpoints using dbed_ckptdisplay

You can use the `dbed_ckptdisplay` command to display the Storage Checkpoints associated with an Oracle database from the command line.

Before displaying Storage Checkpoints, the following conditions must be met:

Prerequisites

- You must be logged on as the database administrator.

Usage Notes

- In addition to displaying the Storage Checkpoints created by SF Enterprise products, `dbed_ckptdisplay` also displays other Storage Checkpoints (for example, Storage Checkpoints created by the Capacity Planning Utility and NetBackup).
- The Status field identifies if the Storage Checkpoint is partial (P), complete ©), error (E), mounted (M), read only ®), writable (W), or of type online (ON), offline (OF), instant (IN), or unknown (UN).
- Database FlashSnap commands are integrated with Storage Checkpoint functionality. It is possible to display and mount Storage Checkpoints carried over with snapshot volumes to a secondary host.
- See the `dbed_ckptdisplay`(1M) manual page for more information.

To display Database Checkpoints

- Use the `dbed_ckptdisplay` command as follows to display information for Storage Checkpoints created by SF Enterprise products:

```
 $ /opt/VRTS/bin/dbed_ckptdisplay -S PROD \
-H /oracle/product/ORA_HOME


SFDB vxsfadm WARNING V-81-0962 dbed_ckptdisplay will be deprecated
 in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative use
/opt/VRTS/bin/vxsfadm
Storage Checkpoint       Creation time             Status
============================================================
Ckpt_1378463767          Fri Sep  6 14:58:16 2013  C+R+ON
Ckpt_1378465225          Fri Sep  6 15:00:32 2013  C+R+OF
Ckpt_1378465458          Fri Sep  6 14:58:29 2013  C+R+IN
```

To display other Storage Checkpoints

- Use the `dbed_ckptdisplay` command as follows:

```
$ /opt/VRTS/bin/dbed_ckptdisplay -S PROD \
-H /oracle/product/ORA_HOME -o other

SFDB vxsfadm WARNING V-81-0962 dbed_ckptdisplay will be deprecated
 in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative use
 /opt/VRTS/bin/vxsfadm
Storage Checkpoint                   Creator  Location
-----------------------------------  -------  --------
NetBackup_incr_PROD_95518740         NBU      /data
NetBackup_incr_PROD_95518673         NBU      /data
```

To display all Storage Checkpoints

- Use the `dbed_ckptdisplay` command as follows:

```
$ /opt/VRTS/bin/dbed_ckptdisplay -S PROD \
-H /oracle/product/ORA_HOME -o all


SFDB vxsfadm WARNING V-81-0962 dbed_ckptdisplay will be
 deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative use
 /opt/VRTS/bin/vxsfadm
Storage Checkpoint       Creation time             Status
==========================================================
Ckpt_1378463767          Fri Sep  6 14:58:16 2013  C+R+ON
Ckpt_1378465225          Fri Sep  6 15:00:32 2013  C+R+OF
Ckpt_1378465458          Fri Sep  6 14:58:29 2013  C+R+IN

Storage Checkpoint                   Creator  Location
-----------------------------------  -------  --------
NetBackup_incr_PROD_95518740         NBU      /data
NetBackup_incr_PROD_95518673         NBU      /data
```

Mounting Storage Checkpoints using dbed_ckptmount

You can use the `dbed_ckptmount` command to mount a Storage Checkpoint for the database from the command line.

Before mounting Storage Checkpoints, the following conditions must be met:

Prerequisites

- You must be logged on as the database administrator.

Usage notes

- The `dbed_ckptmount` command is used to mount a Storage Checkpoint into the file system namespace. Mounted Storage Checkpoints appear as any other file system on the machine and can be accessed using all normal file system based commands.
- Storage Checkpoints can be mounted as read only or read-write. By default, Storage Checkpoints are mounted as read only.
- If the rw (read-write) option is used, rw_xxxx, where xxx is an integer, will be appended to the Storage Checkpoint name.
- If the specified mount point directory does not exist, then `dbed_ckptmount` creates it before mounting the Storage Checkpoint, as long as the Oracle database owner has permission to create it.
- Database FlashSnap commands are integrated with Storage Checkpoint functionality. It is possible to display and mount Storage Checkpoints carried over with snapshot volumes to a secondary host.
- See the `dbed_ckptmount`(1M) manual page for more information.

To mount Storage Checkpoints with the read/write option

◆ Use the `dbed_ckptmount` command as follows:

```
$ /opt/VRTS/bin/dbed_ckptmount -S PROD -c Ckpt_971672042 \
-m /tmp/ckpt_rw -o rw
SFDB vxsfadm WARNING V-81-0962 dbed_ckptmount will be
deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative use
/opt/VRTS/bin/vxsfadm
Creating Storage Checkpoint Ckpt_1378463767_rw_1378469718 ...  Done
Storage Checkpoint Ckpt_1378463767_rw_1378469718 created
Mounting Checkpoint to /tmp/ckptrw...                          Done
```

To mount Storage Checkpoints with the read only option

◆ Use the dbed_ckptmount command as follows:

```
$ /opt/VRTS/bin/dbed_ckptmount -S PROD -c Ckpt_971672042 \
-m /tmp/ckpt_ro -o ro
```

Unmounting Storage Checkpoints using dbed_ckptumount

You can use the `dbed_ckptumount` command to unmount a Storage Checkpoint for an Oracle database from the command line.

Before unmounting Storage Checkpoints, the following conditions must be met:

Prerequisites
- You must be logged on as the database administrator.

Usage notes
- The `dbed_ckptumount` command is used to unmount a mounted Storage Checkpoint from the file system namespace. Mounted Storage Checkpoints appear as any other file system on the machine and can be accessed using all normal file system based commands. When mounted Storage Checkpoints are not required, they can be unmounted.
- See the `dbed_ckptumount`(1M) manual page for more information.

To unmount Storage Checkpoints

◆ Use the `dbed_ckptumount` command as follows:

```
$ /opt/VRTS/bin/dbed_ckptumount -S PROD -c Ckpt_97167042
SFDB vxsfadm WARNING V-81-0962 dbed_ckptumount will be
deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative use
/opt/VRTS/bin/vxsfadm
Unmounting the checkpoint...                                  Done
```

Performing Storage Rollback using dbed_ckptrollback

You can use the `dbed_ckptrollback` command to rollback an Oracle database to a Storage Checkpoint.

Before performing a Storage Rollback, the following conditions must be met:

Prerequisites
- You must be logged on as the database administrator.

Usage notes
- The `dbed_ckptrollback` command rolls an Oracle database back to a specified Storage Checkpoint. You can perform a Storage Rollback for the entire database, a specific tablespace, or list of datafiles.
  Database rollback for the entire database requires that the database be inactive before Storage Rollback commences. The `dbed_ckptrollback` command will not commence if the Oracle database is active. However, to perform a Storage Rollback of a tablespace or datafile, only the tablespace or datafile to be rolled back must be offline (not the entire database).
- See the `dbed_ckptrollback`(1M) manual page for more information.

To roll back an Oracle database to a Storage Checkpoint

- Use the `dbed_ckptrollback` command as follows:

```
$ /opt/VRTS/bin/dbed_ckptrollback -S PROD \
-H /oracle/product/ORA_HOME -c Ckpt_903937870
```

To rollback a tablespace to a Storage Checkpoint

- Use the `dbed_ckptrollback` command with the `-T` option as follows:

```
$ /opt/VRTS/bin/dbed_ckptrollback -S PROD \
-H /oracle/product/ORA_HOME -T DATA01 -c Ckpt_903937870
```

If the Oracle database is running, you must take the tablespace offline before running this command. If the tablespace is online, the command will fail.

Note: The TEMP tablespaces cannot be rolled back.

To rollback datafiles to a Storage Checkpoint

- Use the `dbed_ckptrollback` command with the `-F` option as follows:

```
$ /opt/VRTS/bin/dbed_ckptrollback -S PROD \
-H /oracle/product/ORA_HOME \
-F /share/oradata1/data01.dbf,/share/oradata2/index01.dbf \
-c Checkpoint_903937870
```

If the Oracle database is running, you must take the datafile offline before running this command. If the datafile is online, the command will fail.

Removing Storage Checkpoints using dbed_ckptremove

You can use the `dbed_ckptremove` command to remove a Storage Checkpoint for an Oracle database at the command line.

Before removing Storage Checkpoints, the following conditions must be met:

Prerequisites
- You must be logged on as the database administrator.

Usage notes
- The `dbed_ckptremove` command is used to remove a Storage Checkpoint from the file system, or file systems, it is associated with. The Storage Checkpoint must have been created using the `dbed_ckptcreate`(1M) command.
- See the `dbed_ckptremove`(1M) manual page for more information.

To remove Storage Checkpoints

- Use the dbed_ckptremove command as follows:

```
$ /opt/VRTS/bin/dbed_ckptremove -S PROD \
-c Ckpt_971672042
```

Cloning the Oracle instance using dbed_clonedb

You can use the `dbed_clonedb` command to clone an Oracle instance using a Storage Checkpoint.

Cloning an existing database using a Storage Checkpoint must be done on the same host.

Warning: On AIX:

If the `-o cio` option was used with the `mount` command to mount your primary database file systems, Concurrent I/O will not be preserved when using `dbed_clonedb` to create a clone database.

Before cloning the Oracle instance, the following conditions must be met:

Prerequisites

- You must first create a Storage Checkpoint.
  See "Creating Storage Checkpoints using dbed_ckptcreate" on page 437.
- You must be logged in as the database administrator.
- Make sure you have enough space and system resources to create a clone database on your system.
- A clone database takes up as much memory and machine resources as the primary database.

Usage notes

- The `dbed_clonedb` command is used to create a copy of a database, cloning all existing database files to new locations.
- The *ORACLE_SID* and *ORACLE_HOME* environment variables must be set to the primary database.
- See the `dbed_clonedb`(1M) manual page for more information.

Limitations for Oracle RAC

- When you clone the database by using Storage Checkpoint, the node can be any node in the same Oracle RAC cluster but the archive log destination is required to be on CFS file system. Otherwise, you must manually copy the archive log files.

Table 32-4 lists the options for the dbed_clonedb command.

Table 32-4 dbed_clonedb command options

Option	Description
`-S` *`CLONE_SID`*	Specifies the name of the new Oracle SID, which will be the name of the new database instance.
`-m` *`MOUNT_POINT`*	Indicates the new mount point of the Storage Checkpoint.
`-c` *`CKPT_NAME`*	Indicates the name of the Storage Checkpoint.
`-o umount`	Shuts down the clone database and unmounts the Storage Checkpoint file system.
`-o restartdb`	Mounts the Storage Checkpoint file system and starts the clone database. The `-o restartdb` option will not attempt to recover the clone database.
`-d`	Used with the `-o umount` option. If the `-d` option is specified, the Storage Checkpoint used to create the clone database will be removed along with the clone database.
`-p`	Specifies a file containing initialization parameters to be modified or added to the clone database's initialization parameter file prior to startup. The format of the pfile_modification_file is the same as that of the Oracle initialization parameter file.

To clone an Oracle instance

- Use the `dbed_clonedb` command as follows:

```
$/opt/VRTS/bin/dbed_clonedb -S new2 -m /tmp/new2 \
-c onCkpt
SFDB vxsfadm WARNING V-81-0962 dbed_clonedb will
 be deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative
use /opt/VRTS/bin/vxsfadm
Creating Storage Checkpoint onCkpt_rw_1378467816 ... Done
Storage Checkpoint onCkpt_rw_1378467816 created
Mounting Checkpoint to /tmp/new2...                          Done
Cloning the Pfile ...                                        Done
Mounting the database...                                     Done
Recovering the clone database...
```

To shut down the clone database and unmount the Storage Checkpoint

- Use the `dbed_clonedb` command as follows:

```
$ /opt/VRTS/bin/dbed_clonedb -S new2 -o umount
SFDB vxsfadm WARNING V-81-0962 dbed_clonedb will
be deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative
use /opt/VRTS/bin/vxsfadm
Shutting down the clone database new2... Done
Unmounting the checkpoint... Done
```

To mount a Storage Checkpoint file system and start the clone database

- Use the `dbed_clonedb` command as follows:

```
$/opt/VRTS/bin/dbed_clonedb -S new2 -o restartdb
SFDB vxsfadm WARNING V-81-0962 dbed_clonedb will be
 deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative use
 /opt/VRTS/bin/vxsfadm
Mounting Checkpoint to /tmp/new2... Done
Mounting the database... Done
Starting the Clone database ... Done
```

To delete a clone database and the Storage Checkpoint used to create it

- Use the dbed_clonedb command as follows:

```
$ /opt/VRTS/bin/dbed_clonedb -S new2 -o umount -d
SFDB vxsfadm WARNING V-81-0962 dbed_clonedb will be
 deprecated in the next release
SFDB vxsfadm WARNING V-81-0963 As an alternative use
 /opt/VRTS/bin/vxsfadm
Shutting down the clone database new2... Done
Unmounting the checkpoint... Done
Deleting the Checkpoint onCkpt_rw_1316003740...
Deleting Checkpoint from /db01... Done
```

Section 9

Reference

- Appendix A. VCS Oracle agents
- Appendix B. Sample configuration files for clustered deployments
- Appendix C. Database FlashSnap status information
- Appendix D. Using third party software to back up files

Appendix A

VCS Oracle agents

This appendix includes the following topics:

- VCS agents for Oracle

VCS agents for Oracle

The VCS agents for Oracle include the following agents that work together to make Oracle highly available:

- The Oracle agent monitors the database processes.
  See "Oracle agent functions" on page 449.
  See "Resource type definition for the Oracle agent" on page 453.
- The Netlsnr agent monitors the listener process.
  See "Netlsnr agent functions" on page 458.
- The ASMDG agent monitors the Oracle ASM disk groups.

Refer to the *Cluster Server Agent for Oracle Installation and Configuration Guide* for more details on the agent functions and the resource types.

Oracle agent functions

The Oracle agent monitors the database processes.

Table A-1 lists the Oracle agent functions.

Table A-1 Oracle agent functions

Agent operation	Description
Monitor	Verifies the status of the Oracle processes. The Oracle agent provides two levels of monitoring: basic and detail. See "Monitor options for the Oracle agent in traditional database and container database" on page 450.
Info	Provides the static and dynamic information about the state of the database. **Note:** This attribute is not applicable PDB resources. See "Info entry point for Cluster Server agent for Oracle" on page 451.
Action	Performs the predefined actions on a resource. See "Action entry point for Cluster Server agent for Oracle" on page 452.

Monitor options for the Oracle agent in traditional database and container database

The Oracle agent provides two levels of monitoring: basic and detail. By default, the agent does a basic monitoring.

The basic monitoring mode has the following options:

- Process check
- Health check

The MonitorOption attribute of the Oracle resource determines whether the agent must perform basic monitoring in Process check or Health check mode.

Table A-2 describes the basic monitoring options.

Table A-2 Basic monitoring options

Option	Description
0 (Default)	Process check The agent scans the process table for the ora_dbw0, ora_smon, ora_pmon, ora_lmon, and ora_lgwr processes to verify that Oracle is running.

Table A-2 Basic monitoring options *(continued)*

Option	Description
1	Health check The agent uses the Health Check APIs from Oracle to monitor the SGA and retrieve the information about the instance.

Review the following considerations if you want to configure basic monitoring:

- Basic monitoring of Oracle processes is user-specific. As a result, an Oracle instance started under the context of another user cannot be detected as online. For example, if an Oracle instance is started under the user "oraVRT" and the agent is configured for a user "oracle", the agent will not detect the instance started by "oraVRT" as online.
  This could lead to situations where issuing a command to online a resource on a node might online an already running instance on that node (or any other node).
  So, Veritas recommends that instances started outside Veritas InfoScale products control be configured with the correct Owner attribute corresponding to the OS user for that instance.

In the detail monitoring mode, the agent performs a transaction on a test table in the database to ensure that Oracle database functions properly. The agent uses this test table for internal purposes. Veritas recommends that you do not perform any other transaction on the test table.

Info entry point for Cluster Server agent for Oracle

The supports the Info entry point, which provides static and dynamic information about the state of the database.

To invoke the Info entry point, type the following command:

```
# hares -value resource ResourceInfo [system]\
 [-clus cluster | -localclus]
```

For the above command to be executed successfully, you must perform the following command:

- Execute the resource refreshinfo using the following command:
  ```
  # hares -refreshinfo <res-name> -sys <sys-name>
  ```

The entry point retrieves the following static information:

- Version
- InstanceNo
- InstanceName

- DatabaseName
- HostName
- StartupTime
- Parallel
- Thread
- InstanceRole

The entry point retrieves the following dynamic information:

- InstanceStatus
- Logins
- OpenMode
- LogMode
- ShutdownPending
- DatabaseStatus
- Shared Pool Percent free
- Buffer Hits Percent

You can add additional attributes by adding sql statements to the file /opt/VRTSagents/ha/bin/Oracle/resinfo.sql. For example:

```
select 'static:HostName:'||host_name from v$instance;
select 'dynamic:ShutdownPending:'||shutdown_pending from
v$instance;
```

The format of the selected record must be as follows:

```
attribute_type:userkey_name:userkey_value
```

The variable *attribute_type* can take the value static and/or dynamic.

Action entry point for Cluster Server agent for Oracle

The supports the Action entry point, which enables you to perform predefined actions on a resource.

To perform an action on a resource, type the following command:

You can also add custom actions for the agent.

Table A-3 describes the agent's predefined actions.

Table A-3 Predefined agent actions

Action	Description
VRTS_GetInstanceName	Retrieves the name of the configured instance. You can use this option for the Oracle and the Netlsnr resources.
VRTS_GetRunningServices	Retrieves the list of processes that the agent monitors. You can use this option for the Oracle and the Netlsnr resources.

Table A-3 Predefined agent actions *(continued)*

Action	Description
DBRestrict	Changes the database session to enable the RESTRICTED mode.
DBUndoRestrict	Changes the database session to disable the RESTRICTED mode.
DBSuspend	Suspends a database.
DBResume	Resumes a suspended database.
DBTbspBackup	Backs up a tablespace; `actionargs` contains name of the tablespace to be backed up.

Note: VRTS_GetRunningServices, DBRestrict, DBResume, DBSuspend, DBTbspBackup, and DBUndoRestrict action entry points are not supported for Oracle PDB resources.

Resource type definition for the Oracle agent

The Oracle agent of the is represented by the Oracle resource type in Veritas InfoScale products.

The following extract shows the type definition of the Oracle resource in the OracleTypes.cf file.

Attribute definitions for the Oracle agent

The Oracle agent attributes are classified as required, optional, and internal.

You must assign values to the following required attributes.

Table A-4 Required attributes for Oracle agent

Required attributes	Type and dimension	Definition
Sid	string-scalar	The variable $ORACLE_SID that represents the Oracle instance. The Sid is considered case-sensitive by the Oracle agent and by the Oracle database server. For a policy managed database, the Sid attribute should be set to Sid prefix.

Table A-4 Required attributes for Oracle agent *(continued)*

Required attributes	Type and dimension	Definition
Owner	string-scalar	The Oracle user who has privileges to start or stop the database instance. The agent also supports LDAP users as Oracle user.
Home	string-scalar	The $ORACLE_HOME path to Oracle binaries and configuration files. For example, you could specify the path as /opt/ora_home. **Note:** Do not append a slash (/) at the end of the path.

You can configure the following optional attributes if they are necessary for your configuration.

Table A-5 Optional attributes for Oracle agent

Optional Attributes	Type and Dimension	Definition
StartUpOpt	string-scalar	Startup options for the Oracle instance. This attribute can take the following values for traditional database and container database: ■ STARTUP ■ STARTUP_FORCE ■ RESTRICTED ■ RECOVERDB ■ SRVCTLSTART ■ CUSTOM ■ SRVCTLSTART_FORCE Default is STARTUP_FORCE. Startup options for the Oracle instance in pluggable database. This attribute can take the following values: ■ STARTUP ■ STARTUP_FORCE ■ RESTRICTED ■ OPEN_RO

Table A-5 Optional attributes for Oracle agent *(continued)*

Optional Attributes	Type and Dimension	Definition
ShutDownOpt	string-scalar	Shut down options for the Oracle instance. This attribute can take the following values for traditional database and container database: ■ IMMEDIATE ■ TRANSACTIONAL ■ SRVCTLSTOP ■ CUSTOM ■ SRVCTLSTOP_TRANSACT ■ SRVCTLSTOP_ABORT ■ SRVCTLSTOP_IMMEDIATE Default is IMMEDIATE. Shut down options for the Oracle instance in pluggable database is IMMEDIATE.
EnvFile	string-scalar	The full path name of the file that is sourced by the entry point scripts. This file contains the environment variables set by the user for the Oracle database server environment such as LD_LIBRARY_PATH, NLS_DATE_FORMAT, and so on. The syntax for the contents of the file depends on the login shell of Owner. File must be readable by Owner. The file must not contain any prompts for user input.
Pfile	string-scalar	The name of the initialization parameter file with the complete path of the startup profile. You can also use the server parameter file. Create a one-line text initialization parameter file that contains only the SPFILE parameter. See the Oracle documentation for more information. **Note:** This attribute is applicable only for traditional and container databases.
AutoEndBkup	boolean-scalar	Setting the AutoEndBkup attribute to a non-zero value takes the datafiles in the database out of the backup mode, during Online. Default = 1 **Note:** If a node fails during a hot backup of container database or pluggable database, you must set AutoEndBkup attribute of the corresponding CDB resource to 1. When the AutoEndBkup is set to 1 for the CDB, it also ends the backup of both CDB and PDB during Online.

Table A-5 Optional attributes for Oracle agent *(continued)*

Optional Attributes	Type and Dimension	Definition
MonitorOption	integer-scalar	Monitor options for the Oracle instance. This attribute can take values 0 or 1. For traditional and container databases: ■ 0—Process check monitoring (recommended) ■ 1—Health check monitoring For pluggable databases: ■ 0—Basic monitoring Default = 0 The agent supports intelligent resource monitoring only for traditional and CDBs when the attribute value is set to 0. See "Monitor options for the Oracle agent in traditional database and container database" on page 450.
MonScript	string-scalar	Pathname to the script provided for detail monitoring. The default (basic monitoring) is to monitor the database PIDs only. **Note:** Second-level monitoring is disabled if the value of the attribute MonScript is invalid or is set to an empty string. The pathname to the supplied detail monitor script is /opt/VRTSagents/ha/bin/Oracle/SqlTest.pl. MonScript also accepts a pathname relative to /opt/VRTSagents/ha. A relative pathname should start with "./", as in the path ./bin/Oracle/SqlTest.pl.
User	string-scalar	Internal database user. Connects to the database for detail monitoring.
Pword	string-scalar	Encrypted password for internal database-user authentication. Encrypt passwords only when entering them using the command-line. Passwords must be encrypted using the VCS Encrypt Utility (`/opt/VRTSvcs/bin/vcsencrypt`). **Note:** This attribute is not applicable for PDB resources.
Table	string-scalar	Table for update by `User/Pword`.

Table A-5 Optional attributes for Oracle agent *(continued)*

Optional Attributes	Type and Dimension	Definition
DBName	string-scalar	Set this attribute only when the database is a policy managed RAC database. The value of this attribute must be set to the database unique name.
ManagedBy	string-scalar	Default value for this attribute is ADMIN. In a policy managed RAC database this attribute must be set to POLICY.
PDBName	string-scalar	This attribute must be configured for a PDB. And the value should be set for a PDB database name. For traditional and CDB database, do not set this attribute.
UseSystemD	Boolean	systemD is a system and service manager for Linux operating systems. It helps manage applications across Linux distributions that support systemD feature. When Online Entry point is invoked with `UseSystemD` attribute enabled in the supported RHEL-compatible distributions, the Oracle resource comes online as a service in system.slice. Without the `UseSystemD` attribute enabled, a typical online entry point starts the resource in user.slice. Consideration: Upon a system reboot, if application resource has been started under user.slice, there is a possibility that the processes are not stopped gracefully. This may cause the database to corrupt. This behavior is different for each Oracle application version. Hence, enable UseSystemD attribute for Oracle on you system. The application process can be viewed as `vcs-<resource_name>.service`, in /etc/systemd/system. Set the `UseSystemD` value to 0 or 1.
SystemDAttrList	string-scalar	Ensure that `UseSystemD` is enabled before you set the key-value pairs in the `SystemDAttrList` attribute. For more information, refer to the Red Hat website to know more about the different limits you can apply to the application. For example, `# hares -modify oracle SystemDAttrList LimitLOCKS 44433` Where, LimitLOCKS is the key and 44433 is the value for the key.

Table A-6 lists the internal attribute for Oracle agent. This attribute is for internal use only. recommends not to modify the value of this attribute.

Table A-6 Internal attributes for Oracle agent

Optional Attributes	Type and Dimension	Definition
AgentDirectory	static-string	Specifies the location of binaries, scripts, and other files related to the Oracle agent. Default is /opt/VRTSagents/ha/bin/Oracle.

Netlsnr agent functions

The listener is a server process that listens to incoming client connection requests and manages traffic to the database. The Netlsnr agent brings the listener services online, monitors their status, and takes them offline.

Table A-7 lists the Netlsnr agent functions.

Table A-7 Netlsnr agent functions

Agent operation	Description
Online	Starts the listener process by using the following command: `lsnrctl start $LISTENER`
Offline	Stops the listener process by using the following command: `lsnrctl stop $LISTENER` If the listener is configured with a password, the agent uses the password to stop the listener.
Monitor	Verifies the status of the listener process. The Netlsnr agent provides two levels of monitoring, basic and detail: ■ In the basic monitoring mode, the agent scans the process table for the `tnslsnr` process to verify that the listener process is running. ■ In the detail monitoring mode, the agent uses the `lsnrctl status $LISTENER` command to verify the status of the Listener process. (Default)
Clean	Scans the process table for `tnslsnr $LISTENER` and kills it.
Action	Performs the predefined actions on a resource. See "Action entry point for Cluster Server agent for Oracle" on page 452.

Resource type definition for the Netlsnr agent

The Netlsnr agent of the is represented by the Netlsnr resource type in Veritas InfoScale products.

```
type Netlsnr (
 static str AgentDirectory = "/opt/VRTSagents/ha/bin/Netlsnr"
 static keylist SupportedActions = { VRTS_GetInstanceName,
        VRTS_GetRunningServices, "tnsadmin.vfd" }
 static str ArgList[] = { Owner, Home, TnsAdmin, Listener,
        EnvFile, MonScript, LsnrPwd, Encoding }
 static int IMF{} = { Mode=3, MonitorFreq=5, RegisterRetryLimit=3 }
 static str IMFRegList[] = { Home, Owner, Listener }
 str Owner
 str Home
 str TnsAdmin
 str Listener = "LISTENER"
 str EnvFile
 str MonScript = "./bin/Netlsnr/LsnrTest.pl"
 str LsnrPwd
 str Encoding
 static boolean IntentionalOffline = 0
)
```

Attribute definition for the Netlsnr agent

Review the description of the Netlsnr agent attributes. The agent attributes are classified as required, optional, and internal.

Table A-8 lists the required attributes for Netlsnr agent. You must assign values to the required attributes.

Table A-8 Required attributes for Netlsnr agent

Required attributes	Type and dimension	Definition
Owner	string-scalar	The Oracle user who has privileges to start or stop the listener process. The agent also supports LDAP users as Oracle user.
Home	string-scalar	The $ORACLE_HOME path to Oracle binaries and configuration files. For example, you could specify the path as /opt/ora_home. Do not append a slash (/) at the end of the path.

Table A-9 lists the optional attributes for Netlsnr agent. You can configure the optional attributes if necessary.

Table A-9 Optional attributes for Netlsnr agent

Optional attributes	Type and dimension	Definition
TnsAdmin	string-scalar	The $TNS_ADMIN path to directory in which the Listener configuration file resides (listener.ora). Default is /var/opt/oracle.
Listener	string-scalar	Name of Listener. The name for Listener is considered case-insensitive by the Netlsnr agent and the Oracle database server. Default is LISTENER. **Note:** If a grid infrastructure is set up, make sure that you do not use the default name for the database listener. Otherwise, you may encounter various issues, because the grid infrastructure starts a listener with its default name, which is LISTENER.
LsnrPwd	string-scalar	The Veritas InfoScale products encrypted password used to stop and monitor the listener. This password is set in the Listener configuration file. Encrypt passwords only when entering them using the command-line. Passwords must be encrypted using the Veritas InfoScale products Encrypt utility.
EnvFile	string-scalar	Specifies the full path name of the file that is sourced by the entry point scripts. This file contains the environment variables set by the user for the Oracle listener environment such as LD_LIBRARY_PATH and so on. The syntax for the contents of the file depends on the login shell of Owner. This file must readable by Owner. The file must not contain any prompts for user input.
MonScript	string-scalar	Pathname to the script provided for detail monitoring. By default, the detail monitoring is enabled to monitor the listener process. **Note:** If the value of the attribute MonScript is set to an empty string, the agent disables detail monitoring. The pathname to the supplied detail monitoring script is /opt/VRTSagents/ha/bin/Netlsnr/LsnrTest.pl. MonScript also accepts a pathname relative to /opt/VRTSagents/ha. A relative pathname should start with "./", as in the path ./bin/Netlsnr/LsnrTest.pl.

Table A-10 lists the internal attribute for Netlsnr agent. This attribute is for internal use only. recommends not to modify the value of this attribute.

Table A-10 Internal attributes for Netlsnr agent

Optional Attributes	Type and Dimension	Definition
AgentDirectory	static-string	Specifies the location of binaries, scripts, and other files related to the Netlsnr agent. Default is /opt/VRTSagents/ha/bin/Netlsnr.

Appendix B

Sample configuration files for clustered deployments

This appendix includes the following topics:

- About sample configuration files
- Sample configuration file for Storage Foundation for High Availability
- Sample configuration file for Storage Foundation for Clustered File System HA
- Sample configuration file for Storage Foundation for Oracle RAC

About sample configuration files

The sample configuration files illustrate several deployment scenarios:

- Storage Foundation for High Availability
- Storage Foundation for Clustered File System HA
- Storage Foundation for Oracle RAC

You may use the sample files as a guideline for setting up your environment.

Sample configuration files with the Oracle agent resources is located in the following directory:

- For SF HA: /etc/VRTSagents/ha/conf/Oracle
- For SFCFSHA: /etc/VRTSagents/ha/conf/Oracle
- For SF Oracle RAC: /etc/VRTSvcs/conf/sample_rac

Sample configuration file for Storage Foundation for High Availability

```
include "OracleASMTypes.cf"
include "types.cf"
include "Db2udbTypes.cf"
include "OracleTypes.cf"
include "SybaseTypes.cf"

cluster cluster1 (
        UserNames = { admin = anoGniNkoJooMwoInl }
        Administrators = { admin }
        )

system system1 (
        )

system system2 (
        )

group Oracle_Group (
        SystemList = { system1 = 0, system2 = 1 }
        AutoStartList = { system1 }
        )

        DiskGroup DG-orabindg (
                DiskGroup = qorasrc
                PanicSystemOnDGLoss = 1
                )

        DiskGroup DG-oradatadg (
                DiskGroup = dbdata_dg
                PanicSystemOnDGLoss = 1
                )

        IP Ora_IP (
                Device = NIC0
                Address = "10.182.187.108"
                NetMask = "255.255.248.0"
                )

        Mount ora_arch1_mnt (
```

```
        MountPoint = "/oraarch"
        BlockDevice = "/dev/vx/dsk/dbdata_dg/oraarchvol"
        FSType = vxfs
        FsckOpt = "-n"
        )

Mount ora_bin_mnt (
        MountPoint = "/opt/oracle/orahome"
        BlockDevice = "/dev/vx/dsk/qorasrc/orasrc10g"
        FSType = vxfs
        FsckOpt = "-n"
        )

Mount ora_data1_mnt (
        MountPoint = "/oradata"
        BlockDevice = "/dev/vx/dsk/dbdata_dg/oradatavol"
        FSType = vxfs
        FsckOpt = "-n"
        )

NIC Ora_NIC (
        Device = NIC0
        )

Oracle Ora_Oracle (
        Sid = ora10g
        Owner = oracle
        Home = "/opt/oracle/orahome"
        Pfile = "/opt/oracle/orahome/dbs/initora10g.ora"
        AgentDebug = 1
        )

Ora_IP requires Ora_NIC
Ora_Oracle requires ora_arch1_mnt
Ora_Oracle requires ora_bin_mnt
Ora_Oracle requires ora_data1_mnt
ora_arch1_mnt requires DG-oradatadg
ora_bin_mnt requires DG-orabindg
ora_data1_mnt requires DG-oradatadg


// resource dependency tree
//
```

```
//      group Oracle_Group
//      {
//      IP Ora_IP
//          {
//          NIC Ora_NIC
//          }
//      Oracle Ora_Oracle
//          {
//          Mount ora_arch1_mnt
//              {
//              DiskGroup DG-oradatadg
//              }
//          Mount ora_bin_mnt
//              {
//              DiskGroup DG-orabindg
//              }
//          Mount ora_data1_mnt
//              {
//              DiskGroup DG-oradatadg
//              }
//          }
//      }
```

Sample configuration file for Storage Foundation for Clustered File System HA

```
include "OracleASMTypes.cf"
include "types.cf"
include "CFSTypes.cf"
include "CVMTypes.cf"
include "Db2udbTypes.cf"
include "OracleTypes.cf"
include "SybaseTypes.cf"

cluster cluster1 (
        UserNames = { admin = HopHojOlpKppNxpJom }
        Administrators = { admin }
        HacliUserLevel = COMMANDROOT
        )

system system1 (
        )
```

```
system system2 (
        )

group Oracle_Group (
        SystemList = { system1 = 0, system2 = 1 }
        AutoStartList = { system1 }

        )

        IP Ora_IP (
                Device = NIC0
                Address = "10.200.117.243"
                NetMask = "255.255.255.0"
                )

        NIC Ora_NIC (
                Device = NIC0
                )

        Oracle Ora_Oracle (
                Sid = ora11g
                Owner = oracle
                Home = "/opt/oracle/orahome"
                Pfile = "/opt/oracle/orahome/dbs/initora11g.ora"
                AgentDebug = 1
                )

        Ora_IP requires Ora_NIC
        Ora_Oracle requires Ora_IP


        // resource dependency tree
        //
        //      group Oracle_Group
        //      {
        //      Oracle Ora_Oracle
        //          {
        //          IP Ora_IP
        //              {
        //              NIC Ora_NIC
        //              }
        //          }
```

```
        //      }


group cvm (
        SystemList = { system1 = 0, system2 = 1 }
        AutoFailOver = 0
        Parallel = 1
        AutoStartList = { system1, system2 }
        )

        CFSMount oraarch_mnt (
                Critical = 0
                MountPoint = "/oraarch"
                BlockDevice = "/dev/vx/dsk/dbdata_dg/oraarchvol"
                )

        CFSMount orabin_mnt (
                Critical = 0
                MountPoint = "/ora11"
                BlockDevice = "/dev/vx/dsk/ora11bin/ora11vol"
                )

        CFSMount oradata2_mnt (
                Critical = 0
                MountPoint = "/oradata2"
                BlockDevice = "/dev/vx/dsk/oradatadg2/oradatavol2"
                )


        CFSMount dbdata_mnt (
                Critical = 0
                MountPoint = "/oradata"
                BlockDevice = "/dev/vx/dsk/dbdata_dg/oradatavol"
                )

        CFSMount oraredo_mnt (
                Critical = 0
                MountPoint = "/oraredo"
                BlockDevice = "/dev/vx/dsk/oradatadg2/oraredovol"
                )

        CFSfsckd vxfsckd (
                )
```

```
CVMCluster cvm_clus (
        CVMClustName = cluster1
        CVMNodeId = { system1 = 0, system2 = 1 }
        CVMTransport = gab
        CVMTimeout = 200
        )


CVMVolDg orabin_voldg (
        CVMDiskGroup = orabindg
        CVMVolume = { oracle_volume }
        CVMActivation = sw
        )

CVMVolDg oradata2_voldg (
        CVMDiskGroup = oradatadg2
        CVMVolume = { oradatavol2, oraredovol }
        CVMActivation = sw
        )

CVMVolDg dbdata_voldg (
        CVMDiskGroup = dbdata_dg
        CVMVolume = { oradatavol, oraarchvol }
        CVMActivation = sw
        )

CVMVxconfigd cvm_vxconfigd (
        Critical = 0
        CVMVxconfigdArgs = { syslog }
        )

cvm_clus requires cvm_vxconfigd
oraarch_mnt requires dbdata_voldg
oraarch_mnt requires vxfsckd
orabin_mnt requires vxfsckd
orabin_voldg requires cvm_clus
oradata2_mnt requires oradata2_voldg
oradata2_mnt requires vxfsckd
oradata2_voldg requires cvm_clus
dbdata_mnt requires dbdata_voldg
dbdata_mnt requires vxfsckd
dbdata_voldg requires cvm_clus
```

```
oraredo_mnt requires oradata2_voldg
oraredo_mnt requires vxfsckd
vxfsckd requires cvm_clus


// resource dependency tree
//
//      group cvm
//      {
//      CFSMount oraarch_mnt
//          {
//          CVMVolDg dbdata_voldg
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
//          CFSfsckd vxfsckd
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
//          }
//      CFSMount orabin_mnt
//          {
//          CFSfsckd vxfsckd
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
//          }
//      CVMVolDg orabin_voldg
//          {
//          CVMCluster cvm_clus
//              {
//              CVMVxconfigd cvm_vxconfigd
//              }
//          }
```

```
//      CFSMount oradata2_mnt
//          {
//          CVMVolDg oradata2_voldg
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
//          }
//      CVMVolDg orabin_voldg
//          {
//          CVMCluster cvm_clus
//              {
//              CVMVxconfigd cvm_vxconfigd
//              }
//          }
//      CFSMount oradata2_mnt
//          {
//          CVMVolDg oradata2_voldg
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
//          CFSfsckd vxfsckd
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
//          }
//      CFSMount dbdata_mnt
//          {
//          CVMVolDg dbdata_voldg
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
```

```
//          CFSfsckd vxfsckd
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
//          }
//      CFSMount oraredo_mnt
//          {
//          CVMVolDg oradata2_voldg
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
//          CFSfsckd vxfsckd
//              {
//              CVMCluster cvm_clus
//                  {
//                  CVMVxconfigd cvm_vxconfigd
//                  }
//              }
//          }
//      }
```

Sample configuration file for Storage Foundation for Oracle RAC

```
include "OracleASMTypes.cf"
include "types.cf"
include "CFSTypes.cf"
include "CRSResource.cf"
include "CVMTypes.cf"
include "MultiPrivNIC.cf"
include "OracleTypes.cf"
include "PrivNIC.cf"

cluster cluster1 (
        UserNames = { admin = hqrJqlQnrMrrPzrLqo,
                 "root@sxsvm02" = 0,
```

```
                 "root@sxsvm03" = 0 }
        Administrators = { admin, "root@sxsvm02",
                 "root@sxsvm03" }
        SecureClus = 1
        UseFence = SCSI3
        HacliUserLevel = COMMANDROOT
        )

system system1 (
        )

system system2 (
        )

group VxSS (
        SystemList = { system1 = 0, system2 = 1 }
        Parallel = 1
        AutoStartList = { system1, system2 }
        OnlineRetryLimit = 3
        OnlineRetryInterval = 120
        )

        Phantom phantom_vxss (
                )

        ProcessOnOnly vxatd (
                IgnoreArgs = 1
                PathName = "/opt/VRTSat/bin/vxatd"
                )



        // resource dependency tree
        //
        //      group VxSS
        //      {
        //      Phantom phantom_vxss
        //      ProcessOnOnly vxatd
        //      }


group cvm (
        SystemList = { system1 = 0, system2 = 1 }
```

```
AutoFailOver = 0
Parallel = 1
AutoStartList = { system1, system2 }
)

Application cssd (
        Critical = 0
        StartProgram = "/opt/VRTSvcs/rac/bin/cssd-online"
        StopProgram = "/opt/VRTSvcs/rac/bin/cssd-offline"
        CleanProgram = "/opt/VRTSvcs/rac/bin/cssd-clean"
        MonitorProgram = "/opt/VRTSvcs/rac/bin/cssd-monitor"
        )

CFSMount ocrmnt (
        Critical = 0
        MountPoint = "/ocr_vote"
        BlockDevice = "/dev/vx/dsk/ocrdg/ocrvol"
        )

CFSfsckd vxfsckd (
        )

CVMCluster cvm_clus (
        CVMClustName = sfrac_rp2
        CVMNodeId = { system1 = 0, system2 = 1 }
        CVMTransport = gab
        CVMTimeout = 200
        )

CVMVolDg ocrvoldg (
        Critical = 0
        CVMDiskGroup = ocrdg
        CVMVolume = { ocrvol }
        CVMActivation = sw
        )

CVMVxconfigd cvm_vxconfigd (
        Critical = 0
        CVMVxconfigdArgs = { syslog }
        )

PrivNIC ora_priv (
        Critical = 0
```

```
        Device @system1 = { NIC1 = 0, NIC2 = 1 }
        Device @system2 = { NIC1 = 0, NIC2 = 1 }
        Address @system1 = "192.168.12.1"
        Address @system2 = "192.168.12.2"
        NetMask = "255.255.240.0"
        )

cssd requires ocrmnt
cssd requires ora_priv
cvm_clus requires cvm_vxconfigd
ocrmnt requires ocrvoldg
ocrmnt requires vxfsckd
ocrvoldg requires cvm_clus
vxfsckd requires cvm_clus


// resource dependency tree
//
//      group cvm
//      {
//      Application cssd
//          {
//          CFSMount ocrmnt
//              {
//              CVMVolDg ocrvoldg
//                  {
//                  CVMCluster cvm_clus
//                      {
//                      CVMVxconfigd cvm_vxconfigd
//                      }
//                  }
//              CFSfsckd vxfsckd
//                  {
//                  CVMCluster cvm_clus
//                      {
//                      CVMVxconfigd cvm_vxconfigd
//                      }
//                  }
//              }
//          PrivNIC ora_priv
//          }
//      }
```

```
group ora_db (
        SystemList = { system1 = 0, system2 = 1 }
        AutoStart = 0
        AutoFailOver = 0
        Parallel = 1
        AutoStartList = { system1, system2 }
        )

        CFSMount archive_mnt (
                Critical = 0
                MountPoint = "/oraarchive"
                BlockDevice = "/dev/vx/dsk/oradg/archivevol"
                )

        CFSMount ora_data_mnt (
                Critical = 0
                MountPoint = "/oradata"
                BlockDevice = "/dev/vx/dsk/oradg/oradatavol"
                )

        CVMVolDg ora_data_voldg (
                Critical = 0
                CVMDiskGroup = oradg
                CVMVolume = { oradatavol }
                CVMActivation = sw
                )

        Oracle ORACLE (
                Sid @system1 = orarp1
                Sid @system2 = orarp2
                Owner = oracle
                Home = "/oracle/app/orahome"
                Pfile @system1 = "/oradata/orarp/spfileorarp.ora"
                Pfile @system2 = "/oradata/orarp/spfileorarp.ora"
                StartUpOpt = SRVCTLSTART
                ShutDownOpt = SRVCTLSTOP
                )

        requires group cvm online local firm
        ORACLE requires archive_mnt
        ORACLE requires ora_data_mnt
        archive_mnt requires ora_data_voldg
```

```
ora_data_mnt requires ora_data_voldg


// resource dependency tree
//
//      group ora_db
//      {
//      Oracle ORACLE
//          {
//          CFSMount archive_mnt
//              {
//              CVMVolDg ora_data_voldg
//              }
//          CFSMount ora_data_mnt
//              {
//              CVMVolDg ora_data_voldg
//              }
//          }
//      }
```

Appendix C

Database FlashSnap status information

This appendix includes the following topics:

- About Database FlashSnap status information
- Database FlashSnap Snapshot status information from the CLI

About Database FlashSnap status information

Veritas Database FlashSnap functionality provides the following information for the various snapplan stages and snapshot procedures:

- Snapshot status information
- Snapshot database status information

Database FlashSnap Snapshot status information from the CLI

To view snapshot status information from the command line interface (CLI), use the `dbed_vmchecksnap` command with the `-o list` option to list all available snapplans for a specified database. Snapshot status information is displayed in the command output under the column heading `SNAP_STATUS`.

Note: The snapshot status and snapshot database status information may also appear in error messages.

Snapshot status information from the CLI

Table C-1 shows detailed information about each snapshot status (`SNAP_STATUS`) value.

Table C-1 Snapshot status information from the CLI

SNAP_STATUS	Completed operations	Allowed operations
`init_full`	■ `dbed_vmchecksnap -o validate` (successful) ■ `dbed_vmsnap -o resync` (successful)	`dbed_vmsnap -o snapshot`
`init_db`	`dbed_vmchecksnap -o validate -f` *`snapplan`* (failed)	Ensure that your storage configuration has been set up correctly.
`snapshot_start`	`dbed_vmsnap -o snapshot` (failed)	Contact your system administrator for help. Use Veritas Volume Manager commands to resynchronize the snapshot volumes, and use `dbed_vmsnap -o snapshot -F` to force snapshot creation.
`snapshot_end`	■ `dbed_vmsnap -o snapshot` (successful)	■ `dbed_vmsnap -o resync` ■ `dbed_vmclonedb -o mount|mountdb|recoverdb`

`snapshot_vol_start`
`snapshot_vol_end`
`resync_dg_start`
`resync_dg_end`
	`dbed_vmsnap -o snapshot` (failed)	Re-run `dbed_vmsnap -o snapshot`

`resync_vol_start`
`resync_vol_end`
`snapshot_dg_start`
`snapshot_dg_end`
	`dbed_vmsnap -o resync` (failed)	Re-run `dbed_vmsnap -o resync`

Table C-1 Snapshot status information from the CLI *(continued)*

SNAP_STATUS	Completed operations	Allowed operations
`resync_start`	`dbed_vmsnap -o resync` (failed)	Contact your system administrator for help. Use Veritas Volume Manager commands to resynchronize the snapshot volumes, and use `dbed_vmsnap -o snapshot -F` to force snapshot creation.
`mount start`	`dbed_vmclonedb -o mount` (failed)	`dbed_vmclonedb -o -umount`
`mount_end`	`dbed_vmclonedb -o mount` (successful)	`dbed_vmclonedb -o umount`
`restartdb_start`	`dbed_vmclonedb -o restartdb` (failed)	■ `dbed_vmclonedb -o umount` ■ Start the snapshot database manually.
`restartdb_end`	`dbed_vmclonedb -o restartdb` (successful)	`dbed_vmclonedb -o umount`
`mountdb_start`	`dbed_vmclonedb -o mountdb` (failed)	Recover the snapshot database manually, then run `dbed_vmclonedb -o update_status`
`mountdb_end`	`dbed_vmclonedb -o mountdb` (successful)	■ `dbed_vmclonedb -o update_status` ■ `dbed_vmclonedb -o umount`
`recoverdb_start`	`dbed_vmclonedb -o recoverdb` (failed)	■ Recover the snapshot database manually, then run `dbed_vmclonedb -o update_status` ■ `dbed_vmclonedb -o umount`
`recoverdb_end`	`dbed_vmclonedb -o recoverdb` (successful)	`dbed_vmclonedb -o umount`

Table C-1 Snapshot status information from the CLI *(continued)*

SNAP_STATUS	Completed operations	Allowed operations
`umount_start`	`dbed_vmclonedb -o umount` (failed)	Verify that your file system(s) are not busy and retry the command.
`umount_end`	`dbed_vmclonedb -o umount` (successful)	■ `dbed_vmclonedb -o mount` ■ `dbed_vmclonedb -o restartdb` ■ `dbed_vmsnap -o resync`

Snapshot database status information from the CLI

To view snapshot database status information from the command line, use the `dbed_vmchecksnap` command with the `-o list` option to list all available snapplans for a specified database. Snapshot database status information is displayed in the command output under the column heading `DB_STATUS`.

Table C-2 shows detailed information about each database status (`DB_STATUS`) value.

Table C-2 Snapshot database status information from the CLI

DB_STATUS	Completed operations
`init`	■ `dbed_vmchecksnap -o validate` (successful) ■ `dbed_vmsnap -o snapshot` (successful)
`database_recovered`	`dbed_vmclonedb -o recoverdb` (successful)

Appendix D

Using third party software to back up files

This appendix includes the following topics:

- About using third party software to back up files
- Using third party software to back up files

About using third party software to back up files

SF Enterprise products supports the use of third party software for backing up files. However, Quick I/O is not supported for SF Enterprise products.

Using third party software to back up files

If you are using third party backup software other than Veritas NetBackup, ensure that it can back up and restore VxFS extent attributes. This is important because restored Oracle Disk Manager (ODM) files rely on proper extent layouts for best performance.

Backing up and restoring Oracle Disk Manager files using Oracle RMAN

Oracle allocates Oracle Disk Manager files with contiguous extent layouts for good database performance. When you restore database files they are allocated using these extent attributes. If you are using Oracle RMAN's conventional backup method with any backup software, datafiles are also restored with the proper extent layouts.

If you are using RMAN's “proxy copy” backup method with a backup software other than Veritas NetBackup, the extent attributes may not be backed up. To ensure the

restored datafiles have proper extent layouts, preallocate the lost datafiles using the `odmmkfile` command. This command preallocates contiguous space for files prior to restoring them.

For example, to preallocate an Oracle datafile with size 100M, assuming the Oracle database block size is 8K, use the `odmmkfile` command and enter:

```
# /opt/VRTS/bin/odmmkfile -h 8k -s 100m filename
```

For additional information about the `odmmkfile` command, see the `odmmkfile(1)` manual page.